PhD-FSTM-2022-132

Faculty of Science, Technology and Medicine

DISSERTATION

Defence held on 17/10/2022 in Luxembourg

to obtain the degree of

DOCTEUR DE L'UNIVERSITÉ DU LUXEMBOURG
EN SCIENCES DE L'INGÉNIEUR

by

Paul Kremer

Born on 31 December 1990 in Luxembourg (Luxembourg)

Design and Control of a Flying Gripper and Manipulator for the Assisted Disposal of Improvised Explosive Devices

Dissertation defence committee:

Dr. Holger Voos (Dissertation Supervisor)
Professor, Université du Luxembourg

Dr. Jean-Régis Hadji-Minaglou (Chairman)
Professor, Université du Luxembourg

Dr. Mohamed Darouach
Professor, Université de Lorraine

Dr. Jose Luis Sanchez Lopez (Vice-Chairman)
Research Scientist, Université du Luxembourg

Dr. Seyed Amin Sajadi Alamdari
Research Scientist, Goodyear Innovation Technology

UNIVERSITY OF LUXEMBOURG

FACULTY OF SCIENCE, TECHNOLOGY AND MEDICINE

INTERDISCIPLINARY CENTRE FOR SECURITY, RELIABILITY AND TRUST (SNT)
AUTOMATION AND ROBOTICS RESEARCH GROUP

DOCTORAL THESIS

DESIGN AND CONTROL OF A FLYING GRIPPER AND MANIPULATOR FOR THE ASSISTED DISPOSAL OF IMPROVISED EXPLOSIVE DEVICES

Author Paul Kremer

Dissertation Supervisors Dr. Holger Voos
Dr. Jean-Régis Hadji-Minaglou
Dr. Mohamed Darouach

Declaration of Authorship

I, Paul Kremer, declare that the Ph.D. thesis entitled ”Design and Control of a Flying Gripper and Manipulator for the Assisted Disposal of Improvised Explosive Devices” and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Paul Kremer

Luxembourg, February 10, 2023

Abstract

Improvised Explosive Devices (IEDs) are an ever-growing worldwide threat. The disposal of IEDs is typically performed by experts of the police or the armed forces with the help of specialized ground Ordnance Disposal Robots (ODRs). Unlike aerial robots, those ODRs have poor mobility, and their deployment in complex environments can be challenging or even impossible. Endowed with manipulation capabilities, aerial robots are capable of performing complex manipulation tasks akin to ground robots. This thesis leverages the manipulation skills and the high mobility of aerial robots to perform aerial disposal of IEDs. Being, in essence, an aerial manipulation task, this work presents numerous contributions to the broader field of aerial manipulation. This thesis presents the mechatronic concept of an aerial ODR and a high-level view of the fundamental building blocks developed throughout this thesis. Starting with the system dynamics, a new hybrid modeling approach for aerial manipulators (AMs) is proposed that provides the closed-form dynamics of any given open-chain AM. Next, a highly integrated, lightweight Universal Gripper (called TRIGGER) customized for aerial manipulation is introduced to improve grasping performance in unstructured environments. The gripper (attached to a multicopter) is tested under laboratory conditions by performing a pick-and-release task. Finally, an autonomous grasping solution is presented alongside its control architecture featuring computer vision and trajectory optimization. To conclude, the grasping concept is validated in a simulated IED disposal scenario.

"There's no learning without trying lots of ideas and failing lots of times."

- Jonathan Ive

Acknowledgements

The end of a long journey. Ultimately, the experience is not just marked by scientific work but, first and foremost, by the people you meet along the way. I had the chance to work in a relatively harmonic research group with a patient supervisor and supportive colleagues, which, as I have learned, cannot be taken for granted.

Therefore, I would like to thank my supervisor, Prof. Holger Voos, for his guidance and for allowing me to conduct my research without stringent bounds. My sincere gratitude to Raja Pandi Perumal, Claudio Cimarelli, Manuel Castillo, and Phillip Ludwig, who accompanied me for many years of the journey and supported me not only with good scientific advice but also with moral support. Also, a big thanks to Hamed Rahimi Nohooji and Jose Luis Sanchez-Lopez for supporting me with valuable advice and constructive feedback during the last steps of the Ph.D.

Lastly, I thank my family for their continuous support throughout these years.

Paul Kremer

Index

1 Introduction **1**

1.1 Improvised Explosive Devices 1

1.2 IED Disposal and Aerial Disposal Concept 2

1.3 Problem Statement and Research Goals 4

1.4 Outline 6

1.5 Publications 6

I Background **7**

2 State of the Art in Aerial Manipulation **8**

2.1 Multirotors 8

2.2 Aerial Manipulators 10

2.3 Grippers in Aerial Manipulation 12

2.4 Control Approaches 13

3 Fundamentals **15**

3.1 Quaternion Fundamentals 15

3.2 Lagrangian Mechanics 18

3.3 Non-linear control 19

  3.3.1 Sliding Mode Control 20

3.3.2 Model Predictive Control 22

II Contributions 24

4 Mechatronic Concept of an Aerial Ordnance Disposal Robot 25

4.1 Introduction 25

4.2 Aerial Manipulator 26

4.2.1 Concept 1: Belt Drive 27

4.2.2 Concept 2: Direct Drive 28

4.2.3 Discussion 28

4.3 Gripper and Modular Interface 29

4.4 Sensing Concept 29

4.5 Automation and Building Blocks 30

5 Hybrid Modeling of Aerial Manipulator Dynamics 35

5.1 Introduction 36

5.1.1 Related Work 37

5.1.2 Contributions 38

5.1.3 Structure 39

5.2 Kinematics 40

5.2.1 UAV Body Kinematics 40

5.2.2 Kinematics of the Manipulator 43

5.2.3 Kinematics of the Manipulator attached to the flying Base 47

5.3 System Dynamics 50

5.3.1 State Space and System Jacobian 51

5.3.2 Equations of Motion 52

5.3.3 Force Mapping 55

5.3.4 Propulsion Forces 57

5.4 Simulation and Constraints 59

5.5 Validation 62

5.5.1 Methodology 62
5.5.2 Results 64
5.6 Control 67
5.6.1 Computed Torque Control 68
5.6.2 Tests and Methodology 70
5.6.3 Results 72
5.7 Conclusion 75

6 TRIGGER: A Lightweight Universal Jamming Gripper for Aerial Grasping 77
6.1 Introduction 78
6.2 Concept and System Architecture 81
6.2.1 Concept 82
6.2.2 Pneumatics and Mechanics 83
6.2.3 Material Selection 87
6.2.4 Fabrication 87
6.2.5 Electronics and Firmware 90
6.2.6 Grasping Procedure 92
6.3 Experiments 95
6.3.1 Experimental Setup 97
6.3.2 Minimum Activation Force 99
6.3.3 Deadener (Additive) 100
6.4 Modeling 102
6.5 Aerial Application 108
6.6 Discussion 110
6.7 Conclusion 112

7 Autonomous Aerial Grasping 113
7.1 Introduction 113
7.2 Object Detection Pipeline 116

7.3 Trajectory Optimization with Soft Perception, Tracking, and Grasping Constraints . . . 121
7.3.1 Discrete Time Formulation . . . 123
7.3.2 Cost Function: Perception . . . 124
7.3.3 Cost Function: Tracking . . . 129
7.3.4 Cost Function: Grasping . . . 130
7.3.5 Cost Function: Repulsion . . . 132
7.3.6 Numerical Validation . . . 133
7.4 Force Control . . . 140
7.4.1 Problem Formulation . . . 141
7.4.2 Force Tracking . . . 142
7.5 Autonomous Grasping in Simulation . . . 146
7.5.1 Methodology and System Architecture . . . 146
7.6 Conclusion . . . 150

8 Conclusion and Future Work **153**

Bibliography **156**

List of Figures

1.1 IED disguised as a tin 2
1.2 Bomb disposal robot 3
1.3 Visual abstract, IED disposal 4

2.1 Types of multirotors 9
2.2 Common types of aerial manipulators 11
2.3 Common types of grippers in aerial manipulation 14

4.1 Concept of a belt-driven 4-DOF robotic arm 32
4.2 Design of a compact robotic joint 33
4.3 Robotic 4-DOF arm concept based on robotic joints 33
4.4 Modular gripper interface 34
4.5 Building blocks 34

5.1 Aerial manipulator with coordinate systems 36
5.2 Kinematic tree 43
5.3 Kinematics of a 2 DOF manipulator 45
5.4 Propulsion system 58
5.5 Physical properties of the Lagrange-model and the corresponding configuration of the quadcopter with a 1 DOF manipulator 62
5.6 Results: Joint positions, quaternion error 63

5.7 Validation methodology 65
5.8 Validation results 66
5.9 Singularity in attitude representation 67
5.10 Blockdiagram of the dynamics framework 68
5.11 Physical properties and configuration of the controlled quadcopter equipped with a 2-DOF manipulator 73
5.12 Results: Simulation with computed torque controller 74

6.1 General aerial grasping concept 80
6.2 Complete assembly of the grasping system 83
6.3 Pneumatic system 85
6.4 Section view of the main gripping module 86
6.5 Fabrication process of the membrane 88
6.6 Mass breakdown of the gripper 89
6.7 Custom controller board 90
6.8 Gripper automaton 92
6.9 Sensors and control topology 93
6.10 TRIGGER grasping a variety of objects 93
6.11 Grasping procedure 96
6.12 Experimental jig 98
6.13 Activation and holding force 101
6.14 Holding force in relation to the nominal activation force (low fill ratio) 101
6.15 Holding force in relation to the nominal activation force (high fill ratio) . . . 102
6.16 Holding force for various percentages of deadener 103
6.17 Membrane model 104
6.18 Elastic force (inflated membrane) 105
6.19 Lumped elastic force (stiff membrane) 105
6.20 Contact force model 107
6.21 Aerial grasping application 109

7.1 Autonomous grasping coordinate frames 115
7.2 Object detection pipeline 117
7.3 VKCV histogram 120
7.4 AM identification 123
7.5 Perception constraint 125
7.6 Logistic functions: sigmoid, pulse, tanh 126
7.7 Perception constraints 128
7.8 Tracking cost 130
7.9 Grasping cost 131
7.10 Repulsion cost 133
7.11 Validation: Scenario 1 (3D) 137
7.12 Validation: Scenario 2 (3D) 138
7.13 Validation: Scenario 3 (3D) 139
7.14 UG force controller 140
7.15 UG system model 141
7.16 Force control results 145
7.17 Gazebo setup 146
7.18 System architecture 147
7.19 IED scenario in Gazebo 149
7.20 Simulated: Path taken by UAV 150
7.21 Simulation: Velocities 151
7.22 Simulation: Activation force 152
7.23 Simulation: Positions 152

List of Tables

5.1 Notation and frequently used symbols 41
5.2 Computational performance scaling 75
5.3 Computational cost (Euler vs quaternion) 75

6.1 Comparison of robotic grippers 79
6.2 Comparison of filler materials 88

7.1 Identified system parameters 123
7.2 MPC gains and constants 135
7.3 SMC gains and constants 144

Acronyms

AM Aerial Manipulator. 5, 6, 10, 11, 13, 19, 30, 31, 35–40, 44, 64, 67, 68, 70–73, 75, 76, 113–116, 119, 121, 122, 124, 129–131, 133, 140, 141, 146–148, 150, 153–155

CAS Computer Algebra System. 23

CPU Central Processing Unit. 73, 115, 117, 118, 120

DOF Degrees of Freedom. 11, 26, 27, 36, 37, 40, 48, 53, 62, 67, 73, 74, 113, 114

EOD Explosive Ordnance Disposal. 2, 4

EPS Expanded Polystyrene. 87, 89

ESC Electronic Speed Controller. 59, 145

FDM Fused Deposition Modeling. 82, 87, 88

FoV Field of View. 124, 127, 136–138, 148

GPS Global Positioning System. 9

GPU Graphics Processing Unit. 113, 115, 117–120

HMI Human Machine Interface. 31, 114, 155

IED Improvised Explosive Device. 1–6, 10, 26, 29, 113–116, 146, 148, 150, 153–155

IMU Inertial Measurement Unit. 9

LIDAR Light Detection and Ranging. 9

MEMS Microelectromechanical System. 84

MPC Model Predictive Control. 13, 19, 20, 22, 23, 115, 117, 120, 122, 126, 134–136, 144, 147–151, 154

NMPC Non-linear Model Predictive Control. 117, 124

OCP Optimal Control Problem. 5, 124, 129, 132, 135

ODR Ordnance Disposal Robot. 5, 6, 25, 31, 153

PANOC Proximal Averaged Newton-type Method for Optimal Control. 124, 135

PWM Pulse width Modulation. 59, 145

ROS Robot Operating System. 117, 147, 148

SLAM Simultaneous Localization and Mapping. 148, 151

SMC Sliding Mode Control. 19–21, 142, 149, 150, 154

SQP Sequential Quadratic Programming. 124

TRIGGER lighTweight univeRsal jammInG Gripper for aErial gRasping. 6, 79–82, 85, 99, 102, 112, 153–155

UART Universal Asynchronous Receiver-Transmitter. 82, 90

UAV Unmanned Aerial Vehicle. 4, 6, 8–13, 19, 23, 26, 27, 36, 38, 40, 43, 47, 72, 73, 78, 81, 82, 85, 90, 100, 108–111, 114, 116, 117, 121, 122, 128–132, 134–136, 140, 142–146, 148, 149, 153, 155

UG Universal Gripper. 78–84, 92, 95, 102, 103, 107–112, 116, 121, 140, 141, 143, 144, 146–148, 150, 154

URDF Unified Robotics Description Format. 39, 44, 62, 70, 75

Chapter 1

Introduction

This thesis presents several contributions to the field of aerial manipulation. More precisely, a modeling approach for aerial manipulators, a universal soft gripper, and an automation concept for the aerial disposal of Improvised Explosive Devices (IEDs). This chapter introduces the general problem associated with IEDs and the disposal thereof (Section 1.1). Section 1.2 presents the motivation and the bigger concept that envisions the disposal of IEDs with aerial robots instead of the traditional ground robots. The problem statement and research goals are given in Section 1.3. The outline of this thesis is presented in Section 1.4.

1.1 Improvised Explosive Devices

The work and its contributions are related to the bigger context of the disposal of Improvised Explosive Devices (IEDs). IEDs are generally defined as homemade bombs fabricated by criminals, terrorists, and other malicious entities deployed in rural areas, urban centers, and conflict regions. Given their homemade and improvised nature, they can be made from virtually any container, such as tin cans, buckets, canisters, and thin-walled tubes [1, 2, 3]. They can thus have virtually any size, shape, and weight. Furthermore, IEDs can have a broad range of trigger mechanisms such as thermal switches, pressure plates, remote triggers, and motion detectors [4]. A complete categorization of IEDs depending on their operational mode is given in [5]. A typical example of an IED is depicted in Fig. 1.1 where the explosive

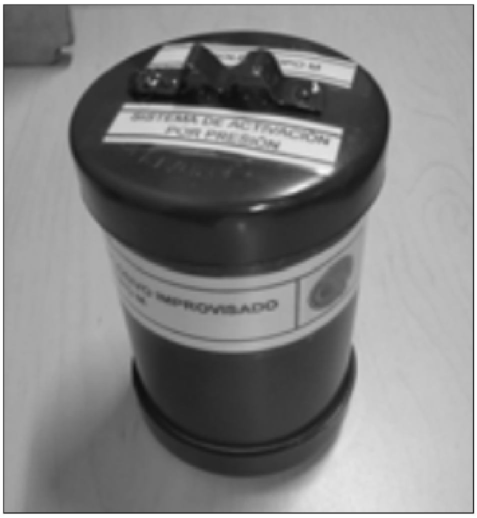


Figure 1.1: An IED disguised as a tin can with a pressure-sensitive trigger [1].

is hidden inside a commodity item (tin can) that gets triggered by a very simple pressure-sensitive trigger mounted on top. This disguise as seemingly harmless commodity items makes them particularly effective against civilians.

According to the United Nations [6], IEDs kill thousands of civilians around the world each year with a reported number of 65 400 casualties in 2014 alone. Incidents have been reported from 66 countries [6] with Syria, Yemen, Afghanistan, Somalia, and Uganda currently being the most affected countries [7]. Consequently, the detection and disposal of IEDs pose significant challenges to both the specialized forces of the military and the police.

1.2 IED Disposal and Aerial Disposal Concept

The disposal of an IED is a complex task and the exact procedure is dependent on many external factors, but typically it is performed in two phases; reconnaissance and defusing resp. disposal.

In the reconnaissance phase, a potentially dangerous object is identified and localized, its hazardous potential and nature are assessed, and the course of further action is decided. On the ground, the experts of the armed forces and the police rely on specialized sensor-packed robots [8], e.g., the Explosive Ordnance Disposal (EOD) robot *PackBot® 510* and the *MK8*

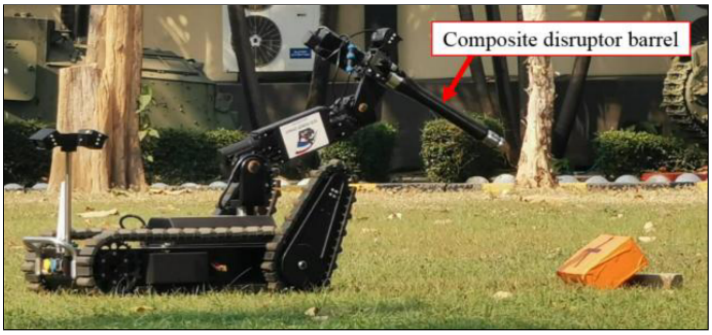


Figure 1.2: The bomb disposal robot *DYNA-T* developed at the King Mongkut's University of Technology [9].

PLUS II equipped with a heavy manipulator.

It should be noted that not every IED can be defused and also that the defusing process can vary greatly depending on the type of IED at hand; the weight and size, type of explosive, the location (lonely street vs. busy commercial center) and the time of the day [4] play a major role regarding the disposal strategy. In some cases, the safest way to proceed is to shoot the IED either with a ballistic projectile or, e.g., a high-speed water-jet [9] as shown in Fig. 1.2 in hope to destroy the internal circuits and consequently neutralize the IED.

Manual disposal of IEDs is, despite protective gear and years of training, an extremely dangerous task and is thus only performed if absolutely necessary. Therefore, the disposal or defusing of IEDs is often conducted with specialized ground robots equipped with a robotic arm, a grasping device, and multiple cameras. Those robots have very poor maneuverability, can be extremely heavy (500 kg), and require a significant logistical effort to deploy. Furthermore, the degree of automation is rather low, and they are thus often referred to as 'drivable manipulators' where the operator has to explicitly control every degree of freedom of the robot, which puts a high mental load on the operator. From a safe distance, a trained human operator teleoperates the robot, directing it to the IED and, if the situation permits, either grabs the device and transports it to a safe zone for disposal, or attempts to directly defuse the IED using the robot's manipulation capabilities. Approaching the IEDs with the robot can also be very challenging, especially in rough terrain and complex urban settings (e.g., stairs and station platforms). Furthermore, approaching the explosive on the ground

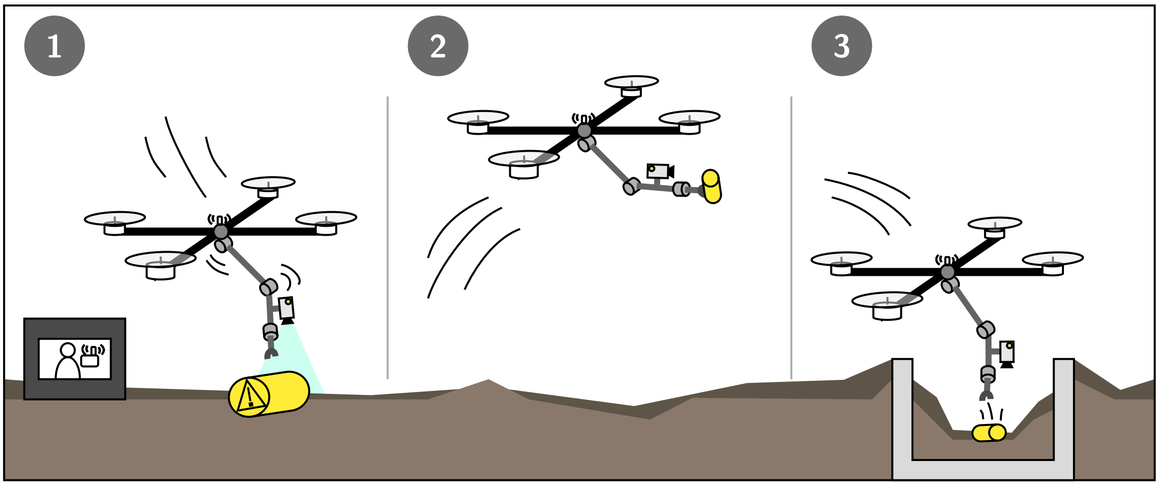


Figure 1.3: IED disposal concept with a flying aerial manipulator. ① Approaching the IED with the AM, ② Transport of the IED to the disposal location, ③ Disposing the IED at the safe location. The operator commands and supervises the AM from a safe distance.

can already trigger an explosion.

Contrary to ground robots, aerial robots have excellent maneuverability and recently started to get increasingly competitive when it comes to physical interaction with their surroundings. The study presented in [10] determined speed, operating range, and lift capability as some of the most important aspects of an EOD robot. Thus the idea of using resp. developing an aerial robot for the disposal of IEDs. The basic IED disposal concept with the help of a flying robot is illustrated in Fig. 1.3, where an aerial robot picks up the explosive with the help of its sensor and manipulation system in order to transport it to a safe disposal location nearby (e.g., a blast-proof containment).

1.3 Problem Statement and Research Goals

The field of aerial manipulation is particularly vast. It combines both the challenges associated with UAVs and classical robotics and furthermore presents a unique set of challenges originating from the interplay of the two systems. That field has many open research questions related to dynamic modeling, control, visual servoing, trajectory optimization, obstacle avoidance, and autonomous operation, but also the design and development of airframes, manipulators, and grippers adapted to the particularities of aerial robotics resp. aerial manipulation.

The general context of this thesis is defined by the big picture of IED disposal with AMs (Fig. 1.3). Since this is largely an aerial manipulation task, the objective of this thesis is to advance and contribute to the field of aerial manipulation in general, keeping the particularities of IEDs in mind. The goal is thus to leverage the high degree of mobility and autonomy of aerial robots to overcome these major shortcomings inherent to traditional ground ODRs.

This study encompasses different contributions to various disciplines within the field of aerial manipulation. More precisely:

1. Dynamics modeling: The development of a general-purpose, singularity-free dynamics framework for aerial manipulators. The proposed framework is preceded by a rigorous and general mathematical development, which is validated in simulation. The framework yields a dynamic model based on a high-level description of an AM. That model can then be used for model-based control.

2. Design & manufacturing: The design and development of a novel, lightweight, energy-efficient, highly integrated universal gripper for aerial manipulation. This development was followed by extensive experimental testing that resulted in the development of the first simulation model thereof.

3. Automation & control: The introduction of an autonomous grasping approach for the developed gripper that matches its characteristics and particularities. Within this context, an onboard fast GPU accelerated object detection pipeline is introduced, followed by trajectory optimization based on solving an Optimal Control Problem (OCP). The grasping process is then fully automated with the sensor data provided by the different subsystems of our model AM.

1.4 Outline

The state of the art in aerial manipulation is presented in Chapter 2 alongside a brief introduction related to UAVs. Chapter 3 introduces the fundamentals of quaternion attitude representation and non-linear control algorithms. In Chapter 4, the mechatronics concept of an aerial ODR is outlined alongside its main building blocks. The dynamics model of the general AM is presented in Chapter 5. TRIGGER, a lightweight universal gripper for IED disposal tasks is developed in Chapter 6. Finally, the control aspects, the object detection pipeline, and the mission control is laid out in Chapter 7. The work is concluded in Chapter 8 whilst pointing out directions for future work.

1.5 Publications

The contributions regarding Chapter 5 ”Hybrid Modeling of Aerial Manipulator Dynamics” and Chapter 6 ”TRIGGER: A Lightweight Universal Jamming Gripper for Aerial Grasping” of this work have been published as the two journal articles:

1. Paul Kremer, Jose Luis Sanchez-Lopez, and Holger Voos. “A Hybrid Modelling Approach for Aerial Manipulators”. In: *Journal of Intelligent and Robotic Systems: Theory and Applications* 105.4 (Aug. 2022), p. 74. ISSN: 15730409. DOI: `10.1007/s10846-022-01640-1`. arXiv: `2206.08644`. URL: `https://link.springer.com/10.1007/s10846-022-01640-1`

2. Paul Kremer et al. “TRIGGER: A Lightweight Universal Jamming Gripper for Aerial Grasping”. In: (Aug. 2022). arXiv: `2208.10768`. URL: `http://arxiv.org/abs/2208.10768` (resubmitted to IEEE Access after minor changes)

The work in Chapter 6 was also partially presented to the soft aerial robotics community during the ”Soft Aerial Robotics Workshop” (SoRo) on April 4, 2022, in Edinburgh, UK. The content of Chapter 7 titled ”Autonomous Aerial Grasping” remains to be published.

Part I

Background

Chapter 2

State of the Art in Aerial Manipulation

2.1 Multirotors

Unmanned Aerial Vehicles (UAVs) are flying robots that are either (semi-)autonomous or teleoperated by a human operator. According to [13], UAVs can be categorized by their relative weight (heavier resp. lighter than air) and their flight mode. While fixed wing Unmanned Aerial Vehicles (UAVs) primarily exploit the aerodynamics of their wings to stay air born, multirotor Unmanned Aerial Vehicles (UAVs) exclusively rely on their propulsion system to stay airborne. The former have the advantage of a long operation range, while the latter have the distinct advantage of being able to hover in midair and perform vertical take-off and landing maneuvers. The hovering ability is especially interesting in the light of aerial manipulation, where the UAV has to stand still while performing a given task.

Especially the sales of multirotors (commonly referred to as ’drones’ or Unmanned Aerial Vehicles (UAVs)) have skyrocketed over the past few of years. Typically equipped with high-quality cameras, they have gained immense popularity with cineastes, photographers (hobbyists and professionals alike) and researchers. Due to their low cost, ease of use, high maneuverability and also their ability to stay in place (hover), they quickly found new applications in the civil and military domains. Those applications include forest fire detection [14], border surveillance [15], spraying of pesticides in agriculture [16] and emergency trans-

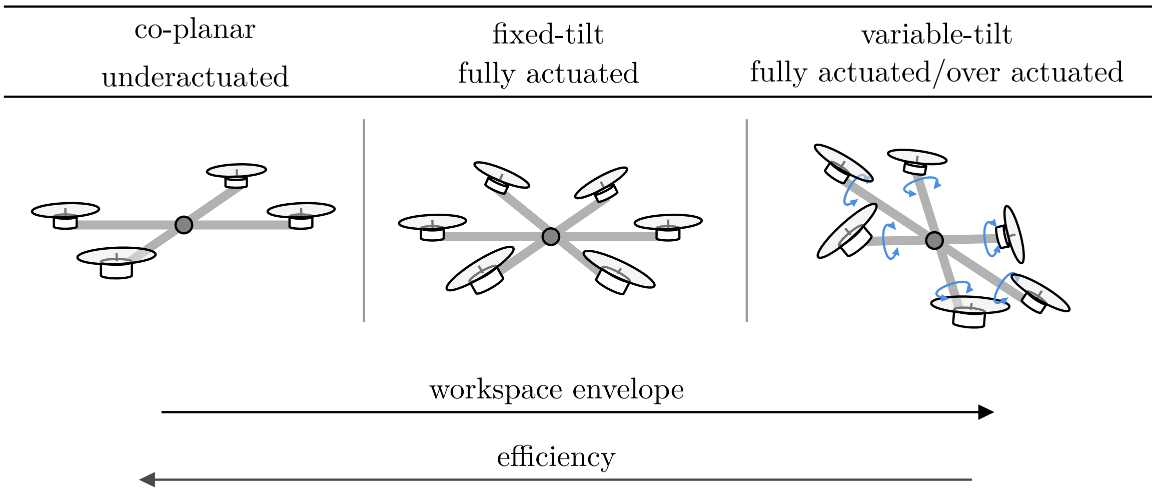


Figure 2.1: The evolution of multirotors from initially underactuated simple quadrotors towards fully actuated or even over-actuated variable-tilt robots.

portation of medical goods [17].

Contrary to fixed-wing aerial robots, multirotors exhibit particularly fast dynamics and cannot realistically be piloted just by a human operator. They generally require an extensive set of sensors such as an Inertial Measurement Unit (IMU), and a barometric pressure sensor which are coupled to an autopilot that provides significant assistance to the human pilot [18] ranging from basic self-leveling over to altitude resp. position control. With the help of a Global Positioning System (GPS) receiver, the autopilots gain the ability to perform fully autonomous flights where the UAV follows a user-defined flight path. In addition to that, ultrasonic and optical sensors (cameras and LIDARs) can be added to give the robot some awareness of its surroundings [19].

The co-planar multirotor is by far the most widespread UAV configuration to date. It is characterized by the fact that all of its propellers generate thrust along the same axis, which makes this configuration relatively efficient in terms of energy consumption, but comes with the drawback of having a coupling between the translational and rotational degrees of freedom. This particular type of robot hence cannot reposition itself without also tilting in one direction. To overcome this limitation, two new types of multirotors emerged: the *fixed-tilt* and the *variable-tilt* multirotors [20] (Fig. 2.1). Both of those types allow completely disjoint control of the UAVs's attitude and position, which consequently greatly expands the workspace and increases the positional accuracy at the expense of reduced energy efficiency. Due to their holonomic nature, fully actuated UAVs could become the go-to platform for aerial manipulators.

Many control algorithms were developed over the years [21], however, the predominant approach is still the model-free cascading PID controller, which delivers adequate performance for many applications. However, UAVs requiring rapid and aggressive maneuvers and - especially - AMs greatly benefit from control schemes based on the dynamic model of the system [13], [22].

2.2 Aerial Manipulators

Traditionally, all assignments performed by UAVs used to be centered around sensing and transportation tasks. These tasks did not require the UAVs to have physical contact with their surroundings. In fact, in this context, all physical interaction with the environment was completely undesired. Consequently, the role of UAVs was of a purely passive nature.

This changed with the emergence of Aerial Manipulators (AMs) [23]. AMs are highly integrated flying robotic systems that typically consist of a UAV equipped with one or multiple robotic arms and/or some form of grasping device (gripper) that allows them to perform various jobs that require manipulation skills. The fundamental idea is to employ those systems as autonomous or semi-autonomous robotic workers that perform, e.g., tasks such as maintenance and inspection of industrial oil & gas plants [24] as well as power-lines [25], aerial transportation [26], bridge inspection [27] and search and rescue missions [28]. Recently, public opinion has been increasingly changing in favor of aerial robots performing commodity tasks, e.g., package delivery [29]. As such, aerial IED disposal could become just one of many tasks performed by those emerging platforms.

The major challenges associated with this field of research are modeling (dynamic model), control (coupled [30] or decoupled [31]), perception resp. the sensing concept and the overall mechanical design of the system [32], [33]. The pervasive configurations of AMs are shown in

kinematic	claw	tether	serial	parallel
DOF	4+0	4+1	4+N	4+3 / 4+6

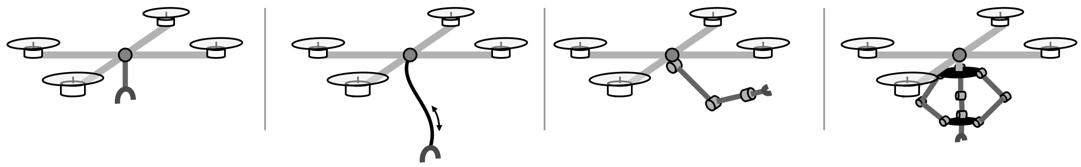


Figure 2.2: Common types of aerial manipulators range from simple claws to multi-DOF serial/parallel link manipulators.

Fig. 2.2. The most basic configuration is represented by the end-effector (the gripper) directly attached to the drone's body. This configuration is also often referred to as 'claw' or 'hook' where the grasping device is mounted either top, bottom or laterally on the drone's airframe [34]. It does not provide any additional DOF, nor does it extend the reach of the flying robot. As such, it represents the most limited AM configuration, with only 4 exploitable DOF (3 translational + yaw) in total. Despite being very basic, this 'claw' configuration has been used with great success [35, 36, 37]. There are, however, cases where the 'claw' configuration reaches its limits. Being placed underneath the UAV's body, they only have limited reach, and as such the bounding box of the carrying platform becomes a limiting factor for many manipulation tasks. This led to the emergence of AMs equipped with a rope (slung load) and serial resp. parallel manipulators that are known from classical robotics. The desirable properties of those manipulators nevertheless differ from their industrial counterparts. The authors of [33] describe the most desirable properties of manipulators within an aerial manipulation context as lightness, mechanical compliance, dexterity and reach. The rationale behind those properties is directly linked to the nature of their carrying platforms, i.e., the multicopter. A lightweight manipulator requires less thrust and thus extends the platform's flight time. Furthermore, it minimizes the dynamic impact on the platform. Compliance is the manipulator's ability to soften (absorb) hard socks associated with physical interaction, which greatly reduces the risk of destabilizing the platform under physical contact. Such a passively compliant robotic arm was developed, e.g., in [38] by using an antagonistic spring mechanism. Compliance is a very desirable property for applications with physical contact.

With almost all platforms in aerial manipulation being underactuated, reach, resp. the size of the workspace is considered an important factor. Due to the spinning propellers, the multirotor has to stay at a safe distance from any obstacles. Therefore, the manipulator has to bridge the gap between the UAV and the targeted object. Furthermore, some objects are easier to grasp from a different angle other than the top. As such, manipulators expand the workspace of aerial robots and add a great deal of flexibility with regard to the grasping direction. The longest reach is obtained with the 'rope' or 'wire' configuration, where the grasping device is suspended at a great distance from the aerial platform. This effectively reduces problems associated with ground effects by keeping the UAV at a sufficient distance from the terrain. That, however, creates new problems as the payload is now prone to oscillate. A list of aerial manipulators developed by several research groups and a comparison in terms of their properties can be found in [39], [40].

2.3 Grippers in Aerial Manipulation

Payloads come in many different geometries, materials, sizes and weights. For that reason, various grippers employing vastly different grasping principles were developed with specific requirements in mind, in an effort to achieve optimal grasping performance for a particular type of workpiece. As an example, ferromagnetic objects are best lifted with a magnetic gripper. The same gripper is, however, ineffective for non-ferrous objects. As a rule of thumb, there is no gripper that is truly universal and performs well over a wide range of payloads. Optimizing the grasper for versatility thus means compromising on other properties (e.g., holding force or reliability) [41]. A complete classification of grippers and grasping mechanisms can be found in [42].

Aerial grasping has seen substantial developments over many years, starting from the early rigid parallel jaw grippers [43, 44] originating from classical robotics towards soft grippers known from soft robotics [45]. Soft grippers are becoming increasingly popular in aerial grasping due to their favorable properties and passive mechanical intelligence built into their soft structures. Being built from soft, compliant elements comes with two distinct advantages;

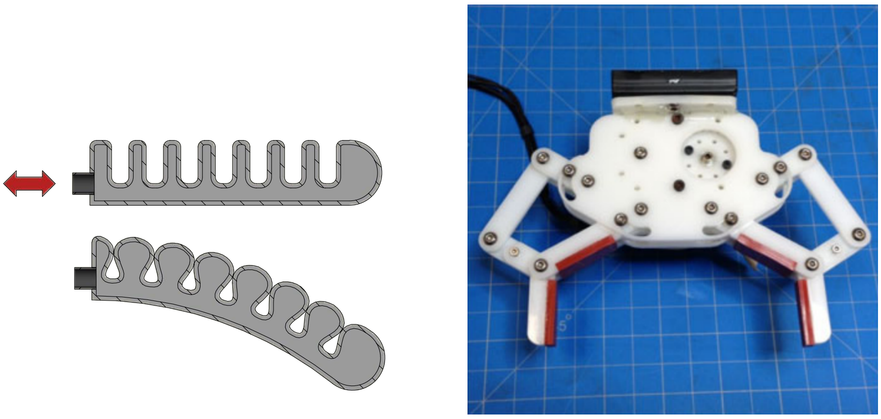
First, it makes them passively compliant and thus safer to use on UAVs. *Second*, possessing virtually infinite degrees of freedom, they tend to be more versatile, i.e., less specialized and thus compatible with a broader range of payloads. A *Fin Ray* gripper was recently shown in [37], and grasping using flexible or semi-flexible multi-finger grippers was introduced in [46, 35], and more specifically in [47] where a dynamic grasping approach was developed. Lastly, there is some work towards avian perching (grasping inspired by birds) [48] using bi-stable grippers [49]. Some of the more common types of grippers are shown in Fig. 2.3.

To some extent, the aforementioned desirable characteristics of manipulators also apply to the end-effectors themselves. A lightweight construction, passive compliance, and dexterity are equally important aspects of aerial grippers. In fact, a compliant gripper can add some degree of compliance to an otherwise rigid system. The current trend in aerial grasping is thus to move away from classic rigid jaw grippers to soft, compliant grippers.

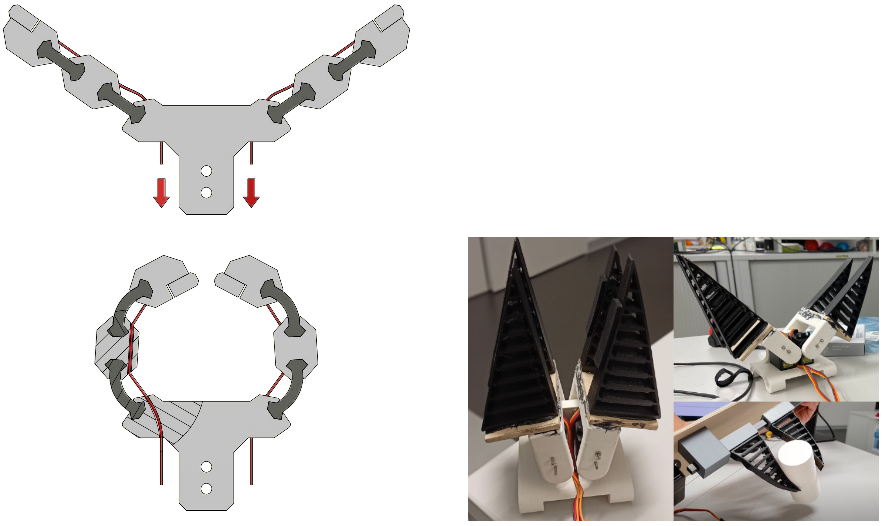
2.4 Control Approaches

Several surveys cover the control models and control methods taken by various research groups, e.g., [39], [40]. For AMs, there are two fundamental modeling approaches; *independent* and *unified.* In the independent approach, the UAV and the manipulator are regarded as two independent subsystems, where the dynamic impact of the manipulator is regarded as a mere disturbance on the UAV (and vice-versa). An example of such a system can be found in [51]. Those subsystems then also have completely separate controllers. In the unified approach, the UAV and the manipulator are seen as a single dynamic system described by a unified dynamic model coupled into a single controller that handles both the robotic arm and the UAV. Therein, the dynamic model of the system describes the changing center of mass and inertia and the reaction forces from the actuation of the joints, which, in the independent approach, are simply external disturbances. That unified approach was shown, e.g., in [52] where the dynamics model is used by a non-linear MPC.

The pervasive control methods for AMs are (adaptive) sliding mode control [43], non-linear model predictive control [52], LQR control [53], backstepping control [44] and various



(a) Pneumatically inflatable finger (b) Rigid jaw gripper [50]



(c) Tendon driven semi-rigid finger (d) Fin Ray gripper [37]

Figure 2.3: Common types of grippers in aerial manipulation

PID-based control schemes [54]. Non-linear, model-based controllers are among the most popular choices. Furthermore, since robustness towards external disturbances (e.g., wind gusts) and model uncertainties is of great importance, many approaches use some form of adaptive behavior [43] or try to estimate the disturbance using a disturbance observer design [55].

Chapter 3

Fundamentals

3.1 Quaternion Fundamentals

Any three-parameter set representing orientations is subject to singularities [56]. The most prominent set of such parameters are the widely used Euler angles (roll, pitch and yaw).

This section briefly introduces quaternions (also referred to as Euler parameters) with the focus on unit quaternions, which play a fundamental role in representing *singularity-free* orientations in three-dimensional space using a set of four parameters. Its most relevant aspects are summarized here. The reader is referred to [57, 58, 59] and [60] for further details.

A quaternion is defined by a set of four coefficients $q_w, q_x, q_y, q_z \in \mathbb{R}$, and three symbols i, j, k,

$$\mathbf{q} = q_w + q_x i + q_y j + q_z k \in \mathbb{H}, \tag{3.1}$$

which can be put in the more convenient form of

$$\mathbf{q} = (q_w, \bar{\mathbf{q}}_v) \in \mathbb{H}, \tag{3.2}$$

where $q_w \in \mathbb{R}$ is the scalar part and $\bar{\mathbf{q}}_v = (q_i, q_j, q_k) \in \mathbb{R}^3$ is the vector part containing the three imaginary coefficients.

The quaternion group $\mathbb{H}$ is endowed with the non-commutative quaternion product defined as

$$\mathbf{q} \otimes \mathbf{p} = \left(q_w p_w - \bar{\mathbf{q}}_v^\intercal \cdot \bar{\mathbf{p}}_v, q_w \bar{\mathbf{p}}_v + p_w \bar{\mathbf{q}}_v + \bar{\mathbf{q}}_v \times \bar{\mathbf{p}}_v\right). \tag{3.3}$$

The norm of a quaternion is defined as

$$\|\mathbf{q}\| = \sqrt{q_w^2 + \bar{\mathbf{q}}_v^\intercal \bar{\mathbf{q}}_v}. \tag{3.4}$$

In the context of mechanics, quaternions describe both a (uniform) scaling and a rotation [58]. A quaternion $\mathbf{q}$ with $\|\mathbf{q}\| = 1$, element of S^3, is called a *unit quaternion*, describing a 4-dimensional unit-sphere, and, contrary to a general quaternion in $\mathbb{H}$, solemnly describe the orientation of a rigid body. S^3 is a double cover of $SO(3)$ meaning that $\mathbf{q}$ and $-\mathbf{q}$ characterize the same orientation.

The inverse of a quaternion is defined as

$$\mathbf{q}^{-1} = \frac{1}{\|\mathbf{q}\|^2} \left(q_w, -\bar{\mathbf{q}}_v\right). \tag{3.5}$$

The conjugate of a quaternion is obtained by negating its vector part

$$\mathbf{q}^* = \left(q_w, -\bar{\mathbf{q}}_v\right), \tag{3.6}$$

and is equal to the inverse in case $\|\mathbf{q}\| = 1$.

The quaternion identity rotation is defined as

$$\mathbf{1} = \mathbf{q} \otimes \mathbf{q}^* = \left(1, \bar{\mathbf{0}}\right), \mathbf{q} \in S^3. \tag{3.7}$$

Unit quaternions can directly be obtained from axis-angle notation (the equivalent of the Euler notation in quaternion space) s.t.:

$$e^{\bar{\mathbf{u}}\theta} = \left(\cos\frac{\theta}{2}, \bar{\mathbf{u}}\sin\frac{\theta}{2}\right), \tag{3.8}$$

where $\bar{\mathbf{u}}$ represents the (normalized) axis of rotation and θ the angle of rotation.

A quaternion is called pure if its scalar part is zero:

$$\mathbf{q} = (0, \bar{\mathbf{q}}_v) \in \mathbb{H}_p \cong \mathbb{R}^4. \tag{3.9}$$

A vector $\bar{\mathbf{u}} \in \mathbb{R}^3$ defined in the body frame can be rotated into the world frame by $\mathbf{q} \in S^3$ using the double quaternion product:

$$\left(0, \bar{\mathbf{u}}'\right) = \mathbf{q} \otimes (0, \bar{\mathbf{u}}) \otimes \mathbf{q}^* \in \mathbb{H}_p. \tag{3.10}$$

The vectors $\bar{\mathbf{u}}$ and $\bar{\mathbf{u}}'$ have the same magnitude if and only if $\|\mathbf{q}\| = 1$. Otherwise, the length of $\bar{\mathbf{u}}$ is scaled by the norm of $\mathbf{q}$.

Composition of two quaternions $\mathbf{q}_1, \mathbf{q}_2 \in S^3$ is similar to the composition of rotation matrices:

$$\mathbf{q}_{12} = \mathbf{q}_1 \otimes \mathbf{q}_2. \tag{3.11}$$

The quaternion product is linear, and as such, it can be expressed as a matrix-vector product:

$$\begin{aligned}
\mathbf{q}_{12} &= \mathbf{q}_1 \otimes \mathbf{q}_2 & (3.12)\\
&= [\mathbf{q_1}]_L \, \mathbf{q}_2 & (3.13)\\
&= [\mathbf{q_2}]_R \, \mathbf{q}_1, & (3.14)
\end{aligned}$$

with the operators

$$[\mathbf{q}]_L = q_w \mathbf{1}_4 + \begin{bmatrix} 0 & -\bar{\mathbf{q}}_v^\intercal \\ \bar{\mathbf{q}}_v & [\bar{\mathbf{q}}_v]_\times \end{bmatrix} \tag{3.15}$$

and

$$[\mathbf{q}]_R = q_w \mathbf{1}_4 + \begin{bmatrix} 0 & -\bar{\mathbf{q}}_v^\intercal \\ \bar{\mathbf{q}}_v & -[\bar{\mathbf{q}}_v]_\times \end{bmatrix}, \tag{3.16}$$

wherein $[.]_\times : \mathbb{R}^3 \rightarrow \mathbb{R}^{3\times3}$ is the cross product operator

$$[\bar{\mathbf{a}}]_\times = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} . \tag{3.17}$$

Applying (3.15) and (3.16) to (3.10) yields the rotation of a vector in matrix notation:

$$\begin{bmatrix} 0 \\ \bar{\mathbf{u}}' \end{bmatrix} = [\mathbf{q}]_L \, [\mathbf{q}^*]_R \begin{bmatrix} 0 \\ \bar{\mathbf{u}} \end{bmatrix} \tag{3.18}$$

The matrix equivalent of (3.8) is given by the Rodrigues rotation formula $R(\theta, \bar{\mathbf{u}}) : \mathbb{R} \times \mathbb{R}^3 \rightarrow$ *SO(3)*, which represents a rotation of an angle θ around an arbitrary axis $\bar{\mathbf{u}}$ and is define as

$$R(\theta, \bar{\mathbf{u}}) = \mathbf{1}_3 + \sin\theta \, [\bar{\mathbf{u}}]_\times + (1 - \cos\theta) \, [\bar{\mathbf{u}}]_\times^2 . \tag{3.19}$$

Within this work, the quaternion is always assumed to be of unit length, i.e., respecting the constraint $\|\mathbf{q}\| = 1$.

3.2 Lagrangian Mechanics

Given a mechanical system with n independent coordinates $\bar{\mathbf{x}} \in \mathbb{R}^n$ (degrees of freedom), Lagrangian mechanics (based on the principle of least action), states that the path taken by the system is such that it minimizes the system's action functional [61].

$$S = \int_{t_1}^{t_2} (E_{kin} - E_{pot}) \, dt = \int_{t_1}^{t_2} \mathcal{L}\left(\bar{\mathbf{x}}, \dot{\bar{\mathbf{x}}}\right) dt \tag{3.20}$$

The trajectory taken by the system is thus given by the stationary points (i.e., first-order derivative of zero) of the system's action functional given by

$$\delta S = 0, \tag{3.21}$$

which gives birth to the well-known Euler-Lagrange formula [62]

$$\frac{d}{dt}\frac{\mathcal{L}}{\dot{\bar{\mathbf{x}}}} - \frac{\delta\mathcal{L}}{\delta\bar{\mathbf{x}}} = 0. \tag{3.22}$$

Alternatively, the Lagrange equations can be derived from d'Alembert's principle and the principle of virtual work [63]. D'Alembert's principle states that the trajectory of a body is such that the virtual work associated with all constraint forces $\bar{\mathbf{f}}_c$ vanishes at every instant, i.e., $\bar{\mathbf{f}}_c \cdot \delta\bar{\mathbf{r}} = 0$ ($\delta\bar{\mathbf{r}}$ being the virtual displacement) since the constraint forces are always perpendicular to the body's trajectory. Following d'Alembert's principle and the principle of virtual work, the following form of the Lagrange equations can be obtained containing external forces $\bar{\mathbf{f}}_e$ along the generalized coordinates:

$$\frac{d}{dt}\frac{\mathcal{L}}{\dot{\bar{\mathbf{x}}}} - \frac{\delta\mathcal{L}}{\delta\bar{\mathbf{x}}} = \bar{\mathbf{f}}_e. \tag{3.23}$$

However, in the case of non-independent coordinates, resp. in the case of constraints described by $g_j(\bar{\mathbf{x}}) = 0$, the constraint forces no longer vanish and the Euler-Lagrange equations take the following form as constraint forces arise (method of Lagrangian multipliers):

$$\frac{d}{dt}\frac{\mathcal{L}}{\dot{x}_i} - \frac{\delta\mathcal{L}}{\delta x_i} = \underbrace{\sum_j \lambda_j \frac{\delta g_j}{\delta x_i}}_{f_{c,i}}. \tag{3.24}$$

Combining (3.23) and (3.24) yields

$$\frac{d}{dt}\frac{\mathcal{L}}{\dot{\bar{\mathbf{x}}}} - \frac{\delta\mathcal{L}}{\delta\bar{\mathbf{x}}} = \bar{\mathbf{f}}_e + \bar{\mathbf{f}}_c. \tag{3.25}$$

3.3 Non-linear control

Even though state-of-the-art autopilots such as PX4 use a simple cascading PID control scheme, non-linear, model-based control strategies are at the heart of many more advanced control solutions for both UAVs and AMs alike (see, e.g., [21] and [40]). Within this work, two control algorithms are used: Sliding Mode Control (SMC) and Model Predictive Control

(MPC).

3.3.1 Sliding Mode Control

The SMC is a robust control technique that is particularly interesting for non-linear systems with parameter uncertainty and bounded disturbances. Recent developments, technical advances and practical issues concerning SMC are summarized in [64].

Let $\dot{x}$ be a general non-linear system defined by

$$\dot{x} = f(x) + g(x)\,u(t) + \vartheta(t) \tag{3.26}$$

where $f(\cdot)$ and $g(\cdot)$ are non-linear smooth functions with $|\vartheta| \leq D$ being an unknown, bounded disturbance. The sliding surface is defined as

$$s = \dot{e} - ce, \tag{3.27a}$$

$$\dot{s} = \ddot{e} - c\dot{e}, \tag{3.27b}$$

where $c > 0$ is a design parameter and $e = x_r - x$ the tracking error. The goal is to keep the closed loop system on the surface defined by $s = 0$ with the help of the control input $u(t)$. The reduced dynamics of the closed-loop system on the sliding surface thus become

$$\dot{e} = ce, \tag{3.28}$$

which reduces system (3.26) to a first-order LTI system with an exponentially stable origin [65]. Assuming a constant rate reaching-law

$$\dot{s} = -\eta \operatorname{sign}(s), \; \eta > 0, \tag{3.29}$$

the input can be defined as

$$u = \frac{1}{g(x)}\left(-\eta \operatorname{sign}(s) - \ddot{x}_r + f(x) + c\dot{e}\right). \tag{3.30}$$

By defining the Lyapunov function

$$V = \frac{1}{2}s^2, \tag{3.31}$$

and its time derivative

$$\dot{V} = s\dot{s}, \tag{3.32}$$

and replacing (3.27b) and (3.30) in (3.32) it follows

$$\begin{aligned}\dot{V} &= s\left(-\eta \cdot \operatorname{sign}(s) - \vartheta - D \cdot \operatorname{sign}(s)\right)\\ &= -\hat{\eta}|s| - s\vartheta < 0, \text{ if } \hat{\eta} > |D|,\end{aligned} \tag{3.33}$$

which, according to the reachability condition [66], ensures finite time stability of s under the bounded disturbance ϑ.

A common problem of sliding mode control is a phenomenon called *chattering* which occurs when the system is confined to the sliding surface and $u(t)$ switches rapidly between the two extremes caused by the discontinuous signum function. The chattering effect is highly undesirable for mechanical systems as it may cause vibrations and even damage to the actuators. Chattering can effectively be mitigated or reduced in several ways; (i) by defining a boundary layer (saturation control) [66], (ii) by using a higher order SMC [67] (e.g., the super twisting controller), (iii) by using adaptive switching gains [68], (iv) or by applying a low-pass filter to the switching function (equivalent control) [69]. A straightforward method to mitigate chattering is to define a boundary layer in the neighborhood of $s = 0$ where the discontinuous signum function is approximated by the continuous saturation function

$$\operatorname{sat}(x) = \begin{cases} \frac{1}{\epsilon}x, & |x| \leq \epsilon \\ \operatorname{sign}(x), & |x| > \epsilon \end{cases} \tag{3.34}$$

that has a linear regime between ±ϵ [66]. This quasi-sliding mode approach comes at the cost of a steady state error and, thus, a general loss of accuracy [69].

As an alternative, the low-pass filter (lpf) method [69] can be used that applies the first-order low-pass filter

$$\dot{z} = \frac{-z + \mathrm{sign(s)}}{\tau} \tag{3.35}$$

to the switching function. That method does not completely eliminate the high-frequency switching noise but reduces it to an acceptable level. The time constant τ has to be chosen to be sufficiently small but larger than the sampling time of the discrete filter. Unlike the saturation function, it does not cause a steady-state error, but it comes at the cost of introducing a time delay into the closed loop system which may cause instability.

3.3.2 Model Predictive Control

Model Predictive Control (MPC) is a model-based control technique for multidimensional systems under physical and/or user-defined constraints. The model of the underlying system is used to predict the system's evolution over a finite moving horizon given a certain control sequence $\bar{\mathbf{u}}$. The sequence of control inputs is chosen such that a cost functional is minimized under a set of constraints. The relative ease with which constraints can be included in the formulation is one of the biggest strengths of MPC, alongside the ability to fully exploit any system knowledge given in form of (non-linear) ordinary differential equations [70].

According to [70] an optimal control problem can generally be stated as

$$\underset{\bar{\mathbf{x}},\bar{\mathbf{u}},T}{\mathrm{Minimize}} \int_0^T J\left(\bar{\mathbf{x}}(t), \bar{\mathbf{u}}(t)\right) dt + E(\bar{\mathbf{x}}(T)) \tag{3.36a}$$

subject to:

$$\bar{\mathbf{x}}(0) = \bar{\mathbf{x}}_0, \tag{3.36b}$$

$$\dot{\bar{\mathbf{x}}}(t) = f\left(\bar{\mathbf{x}}(t), \bar{\mathbf{u}}(t)\right), \tag{3.36c}$$

$$H_t\left(\bar{\mathbf{x}}(t), \bar{\mathbf{u}}(t)\right) \geq 0, \tag{3.36d}$$

$$H_T\left(\bar{\mathbf{x}}(T)\right) = 0, \tag{3.36e}$$

$$t \in [0, T] \tag{3.36f}$$

where $f(\cdot)$ represents the non-linear system dynamics, $H_T(\cdot)$ the terminal constraints, $H_t(\cdot)$ the path constraints, $\bar{\mathbf{x}}_0$ the initial value, and $J(\cdot)$ and $E(\cdot)$ are cost functionals. The problem in (3.36) generally does not possess a closed-form solution. However, a solution can be obtained by discretizing the problem first and then approximating it numerically. To date, the three most widely used approaches to solve optimal control problems are *single shooting*, *multiple shooting* and *collocation*, which are all *direct methods* for solving non-linear programming (NLP) problems of finite dimensions [70].

Typically, finding the optimal control sequence $\bar{\mathbf{u}}(t)$ is computationally very expensive and requires solving a (non-linear) optimization problem, e.g., by sequential quadratic programming (Newton's method with constraints). Therefore, MPC was historically mainly applied to slow processes, e.g., in chemical plants. However, in recent years, driven by advances in microprocessors and the development of new, fast non-linear optimization algorithms, MPC and, more specifically, non-linear MPC became increasingly viable for agile, higher dimensional systems [71] with fast dynamics (such as UAVs). Some authors have demonstrated that it can be applied in the innermost control loops of UAVs [72]. Alongside the algorithmic advances, new user-friendly MPC frameworks have surfaced, such as *ACADO* [73], *ACADOS* [74] and *OpEn* [75], which leverage the CasADi [76] middleware for its symbolic CAS and automatic differentiation capabilities.

Part II

Contributions

Chapter 4

Mechatronic Concept of an Aerial Ordnance Disposal Robot

4.1 Introduction

An ODR typically consists of an (often track-driven) robotic platform (see also Fig. 1.2) that is equipped with a (serial link) robotic manipulator, an end-effector (gripper), and several cameras and other sensors that provide the required feedback to the operator [77]. Those platforms possess poor mobility that originates not only from mechanical shortcomings (heavy weight, poor ground clearance) but also from the platform's complete lack of autonomous behavior. The operator is solely responsible for navigating the robot through complex environments, mentally processing the sensor data, and dealing with the shortcomings of the robotic platform. On the other hand, aerial robots generally possess at least some degree of situational awareness and autonomy [78] that helps them and the operator navigate complex environments safely.

The concept of the developed aerial ODR is based on the established ground-based ODR, taking the limitations and particularities of an aerial platform into account. The different building blocks of the solution are presented and synthesized into a holistic concept. In particular, the presented mechatronic concept addresses the following concerns: (i) the design resp. concept of a serial link manipulator that is up to the task of IED disposal, (ii) the development of a gripper that can grasp a variety of shapes, (iii) the concept and development of a control system that provides the necessary degree of autonomy for the disposal task at hand.

The following core assumptions are made throughout this work: (i) the project only aims to relocate the explosive to a safe location (e.g., a blast-proof container), which also means that it has, in advance, been identified as an IED that is safe to move, (ii) the explosive weights less than the payload capability of the aerial platform, (iii) the item can clearly be recognized and reached by the manipulation system (e.g., it is not stashed under a pile of other objects).

To date, to the best of the author's knowledge, only a couple of publications exist dealing with the topic of aerial ordnance disposal. The authors of [79] aim to grab explosives using a serial link manipulator with a flexible *Fin Ray* gripper that can grasp objects with various shapes. In [80], the authors chose a different approach and aimed to purposefully trigger the explosive by dropping a small robot from a UAV. Lastly, an aerial manipulator with a winch device using the *YOLO* CNN neural network for object recognition was shown in [81].

4.2 Aerial Manipulator

The aerial manipulator is the combination of an UAV and a robotics manipulator. To no surprise, various concepts stemming from classical robotics have been carried over to aerial robotics. Several designs have thus been developed around different kinematic types such as simple 'claw' grippers, delta manipulators, and highly complex multiple DOFs serial link manipulators. Each kinematic type can be realized using different actuation concepts. As such, a serial link manipulator can be conceived with joints that are directly powered by servo motors or other means, e.g., tendon driven via belts, ropes [82], shape memory alloy wires [83]), or even with hydraulic fluids. The tendon-driven concepts have the advantage that the energy source is located at some distance from the joints, and therefore the moving mass is kept to a minimum, a property that is especially interesting for aerial robots where the moving center of mass causes huge disturbances. Nevertheless, tendon-driven concepts also have their downsides. For one, they generally do not scale well; as an example, a belt-driven robotic arm needs $n = \left(n_{DOF}(n_{DOF} + 1)\right)/2$ pulleys that transfer the mechanical energy from the motor to the joints. This means, that a 4-DOF robot needs 10 pulleys, and 15 for a 5-DOF robot. As such, this concept does not scale well in terms of size, weight and complexity. Therefore direct drives are preferred, as can be seen on most state-of-the-art industrial robots. That concept envisions that each robotic joint is not only driven by its own motor but is also fitted with the sensors (e.g., angular encodes and torque sensors), logic, and signal processing that is a requisite to create an independently working robotic joint. Those modules are also often referred to as servo motors, e.g., the *Dynamixel* lineup used in [44]. The outstanding feature of these joints is that they can be combined (chained) into a multi-DOF robotic arm with relative ease. Their big downside is that their mass is distributed along the manipulator rather than being located at the robot's base.

Next, two manipulator concepts based on those considerations are presented based on the assumption that the payload has a mass of 2 kg or less with a range of 0.5 m intended to be mounted on a large hexacopter that has a maximum payload of 6 kg.

4.2.1 Concept 1: Belt Drive

The concept of the belt-driven robotic arm is based on the idea of positioning the heavy parts (i.e., the motors) as close as possible to the center of the base of the UAV. Since each link does not have to carry the weight of all subsequent links and robotic joints, the motors can also be smaller and lighter. Furthermore, it means that the heaviest part will be the base of the robot, which does not move with the arm; thus, the robot's movement only has minimal impact on the center of gravity of the whole system.

A design proposal of the aforementioned concept in form of a 4-DOF serial link belt-driven manipulator is shown in Fig. 4.1. The heavy motors equipped with high-reduction planetary gear sets are located at the base of the manipulator together with the pneumatic pump powering the gripper. The structural parts are made from carbon-fiber-reinforced plastic due to its outstanding stiffness-to-weight ratio. As such, the whole assembly weighs only 3.4 kg where the motors sill account for the bulk of the mass with a total of 1 kg. The wrist joint combines both joints 3 and 4 in a very compact assembly that works similarly to a car's differential; turning both pulleys in the same direction and at the same time actuates the third joint only. Asymmetric actuation will move both joint 3 and 4. A hollow shaft that reaches through the hub permits the electric cables and pneumatic lines to pass through to the quick-exchange mechanism and the tool center point. This design envisioned incremental position sensors at the motor shafts and thus requires zeroing the joint position before use.

4.2.2 Concept 2: Direct Drive

Direct-drive robotic joints are at the core of modern robotics. Endowed with high reduction gearboxes with virtually no backlash, such as harmonic drives (also called strain wave gears) or cycloidal gears, driven by brushless DC motors, and being equipped with force/torque and angular position sensors, they form an excellent combination for high performance, yet compact robotic joints with outstanding torque-to-weight ratios. By design, the robotic joints are meant to be daisy-chained and are generally entirely enclosed and are thus (within reason) resistant to dirt and water ingress, and consequently more reliable, which can also be seen as a net benefit compared to the belt-driven concept.

An example of such a complex robotic joint is shown in [84] and similar actuators can be found on robots from *DLR*, *Kinova* or *HEBI*. A design proposal of such an actuator built from off-the-shelf components is presented in Fig. 4.2, which shows a 90 W BLDC motor, a harming drive, and an absolute magnetic off-axis sensor stuffed into a compact housing producing 12 N m of torque. A slip ring allows the electric wires to pass through the center of the assembly, permitting the joint to rotate indefinitely.

4.2.3 Discussion

Both concepts have inherent strengths and weaknesses, as discussed in previous sections. The decision on which concept to opt for is thus a question of what compromises one is willing to accept. Herein, the decision was made to follow concept 2 "direct drive" with commercial off-the-shelf robotic joints from *HEBI Robotics* arranged in the configuration depicted in

Fig. 4.3, based on the fact that they offer a feature-set adequate for the IED disposal task (high torque, relatively lightweight, absolute position encoders, force sensors), and the high reliability of industrial actuators. To keep the scope manageable, the focus of this work is on the dynamics (see Chapter 5) of such an aerial system equipped with a robotic arm as shown in Fig. 4.3.

4.3 Gripper and Modular Interface

The gripper (end-effector) is the tool attached to the manipulator. It establishes contact with the targeted payloads and is responsible for a secure grasp. As a rule of thumb, highly specialized grippers only work on a limited subset of objects (e.g., a particular shape or material). In contrast, more universal grippers are less sensitive to those variations but generally provide a less secure grasp. In the context of IEDs a broad range of shapes and materials is part of the problem. Therefore, a quick-exchange interface for a wide range of grippers is proposed such that they can be swapped in the field with minimal effort (Fig. 4.4). The interface passes through power (e.g., electrical or pneumatic) and information from potential sensors placed in the different grippers. Within this work, a lightweight pneumatic gripper for aerial robots is introduced in Chapter 6 that implements a simple form of that interface (pneumatics only) combined with a toolless quick-change mechanism.

4.4 Sensing Concept

The eye-in-hand and eye-to-hand camera configurations are well-known from classical robotics. The eye-in-hand configuration places the camera close to (resp., integrates it into) the end effector of the robot. The camera is thus subjected to the same movements as the gripper and guarantees that the field of view of the camera is not obstructed by the manipulator resp. the gripper itself. Contrary to that, the eye-to-hand configuration places the camera in a fixed place in the robot's workspace, giving it a broader field of view of the robot and its environment. However, that comes with the risk of the robot obstructing the camera's field of view. In practice, both concepts can complement each other [85].

In aerial robotics, eye-to-hand configurations are typically only realizable in laboratory resp. indoor-conditions. Therefore, the predominant sensor concept is the eye-in-hand configuration [86, 87], which is also envisioned to be used in this work (see Fig. 4.3 and also Fig. 7.17). Nevertheless, sensing in aerial robotics is generally more complex, and the environment they are operating is more dynamic. Therefore, many different sensors (e.g., ultrasonic, GPS, IMU, and various types of optical sensors) are required such that the aerial manipulator can gain awareness of itself, its target and also its surroundings. The vision system has two roles to fulfill; *First*, it has to detect the object reliably (e.g., via classification), *second* it has to be able to determine the optimal position for the gripper to grasp [87]. Furthermore, the processing of the camera frames has to be done in real-time and with limited resources, as any information gained from the images is potentially invalidated (outdated) by the movement of the robot.

Within this work, the payload is assumed to be localizable with relatively simple means, that is, by color. As the developed gripper is relatively indifferent with respect to the grasping location, the step of determining the precise grasping location is omitted.

4.5 Automation and Building Blocks

For aerial systems with manipulators, the grasping procedure is often separated into discrete phases such as the *approaching* phase and the *grasping* phase [87]. In the approaching phase, the AM approaches the target whilst keeping the manipulator in a fixed position (e.g., such that it minimizes the inertia of the system). Only during the grasping phase is the manipulator activated and guided towards the target using the complete sensor suite onboard the AM. This procedure also stays valid for simpler systems with static claws; however, especially in the domain of soft aerial grasping, this is no longer necessary. The dynamic approach taken there is more akin to how birds perform perching in nature [47]. The approach to choose is nevertheless guided by the problem and equipment at hand. Especially for a safety-critical task such as ordnance disposal, a slower but more reliable approach is preferable. It should also be noted that within this context, the automation concept should be more supportive (i.e., take some of the mental load of the operator) than proactive. Contrary to ground-based ODR, the task cannot be achieved without *some* degree of automation, even with an experienced pilot in charge. Therefore, the bigger concept also requires a Human Machine Interface (HMI) that allows the operator to plan, monitor, and correct the AM during operation. The development of such an HMI is, however, out of scope for this work.

The individual parts (building blocks) of the developed solution tailored to the problem stated in Section 1.2 are shown in Fig. 4.5 and described in detail in the subsequent chapters. The development of such a system is highly multidisciplinary and covers many engineering domains, such as control and software engineering as well as system engineering (mechatronics).

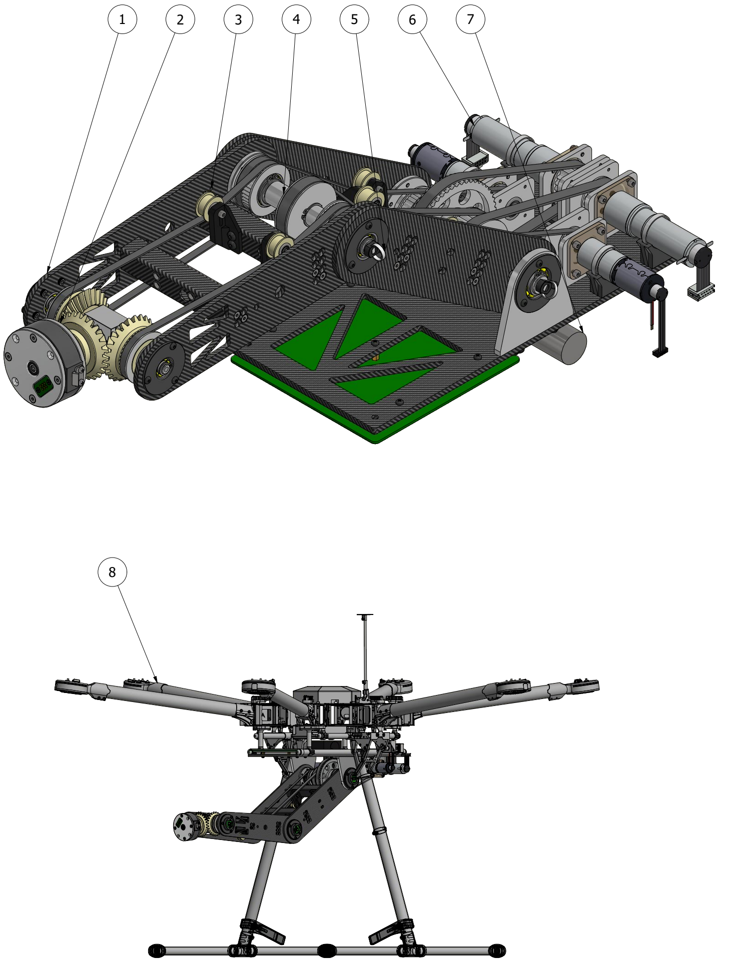


Figure 4.1: Concept of a belt-driven 4-DOF robotic arm. ① Carbon panels, ② Wrist (joint 3 & 4), ③ Belt tensioner, ④ Joint 2, ⑤ Cable clip, ⑥ DC motors, ⑦ Pneumatic pump, ⑧ DJI M600.

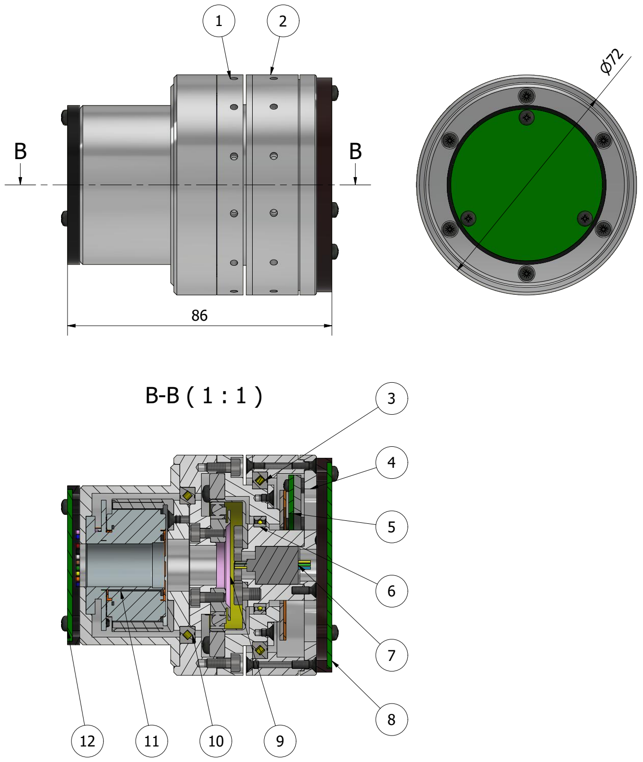


Figure 4.2: Design of a compact robotic joint. ① Input ring, ② Output ring, ③ Cross roller bearing, ④ Torque sensor, ⑤ Angular position sensor, ⑥ Ball bearing, ⑦ Slip ring, ⑧ Output PCB, ⑨ Harmonic drive, ⑩ Cross roller bearing, ⑪ BLDC motor, ⑫ Input PCB.

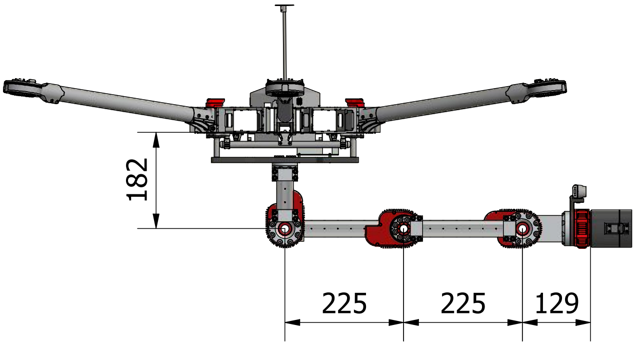


Figure 4.3: Robotic 4-DOF arm concept based on joints from *HEBI robotics*. The concept envisions a quick-change mechanism with a common interface and a ”camera in hand”.

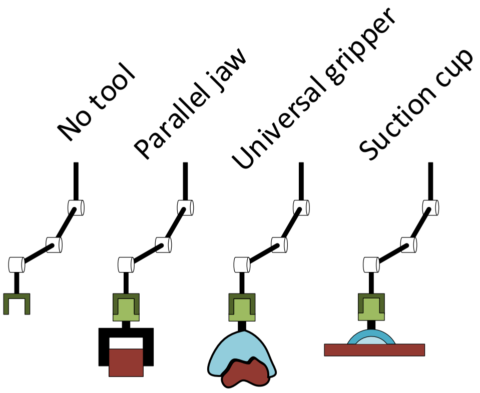


Figure 4.4: Multiple grippers can be attached to a common interface that provides power, which can be either pressurized air or electrical power and communication with potential sensors positioned in the respective grippers.

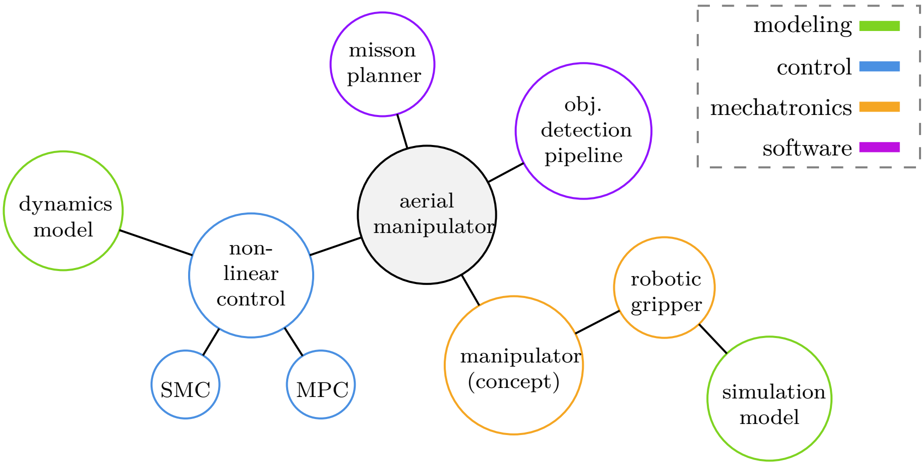


Figure 4.5: The building blocks developed throughout this work aimed toward the fulfillment of the IED disposal task.

Chapter 5

Hybrid Modeling of Aerial Manipulator Dynamics

AMs exhibit particularly challenging non-linear dynamics, where the UAV and its manipulator form a tightly coupled dynamic system. The mathematical model describing these dynamics is the core of many solutions developed in non-linear control. Traditionally, the formulation of the dynamics involves Euler angle parametrization in the Lagrangian framework or quaternion parametrization in the Newton-Euler framework. The former has the disadvantage of giving rise to singularities and the latter of being algorithmically complex. This chapter presents a hybrid solution, combining the benefits of both, namely a quaternion approach leveraging the Lagrangian framework, connecting the singularity-free parameterization with the algorithmic simplicity of the Lagrangian approach. Detailed insights are given into the kinematic modeling process and the formulation of the dynamics of a general AM. The obtained dynamic model is validated against a simulated AM in a real-time physics engine. A practical application of the obtained dynamic model is shown in the context of a computed torque feedback controller (feedback linearization), where its real-time capability is analyzed with increasingly complex AMs configurations. The content of this chapter was published in [11].

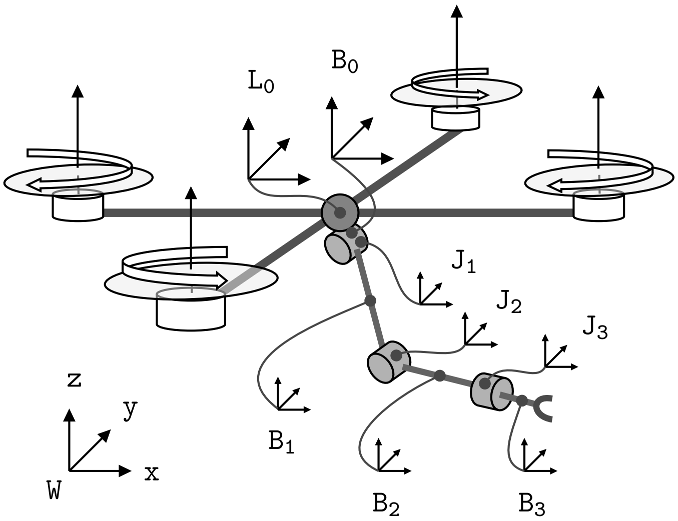


Figure 5.1: Illustration of a typical aerial manipulator; a quadcopter with a 3-DOF serial link manipulator. Indicated are the frames of reference for the joints $\mathtt{J}_i$, the inertial (body) frames $\mathtt{B}_i$ and forces of the propulsion devices for thrust $\bar{\mathbf{f}}_{thr}$ and drag $\bar{\mathbf{f}}_{drag}$.

5.1 Introduction

AMs are flying, autonomous, resp. semi-autonomous Unmanned Aerial Vehicles (UAVs) carrying one or multiple robotic arms to perform a given manipulation task. Some tasks in aerial manipulation can be performed with relatively simple static claws [88] with only minimal impact on the carrying platform. However, more complex tasks require the additional Degrees of Freedom (DOF) offered by serial link manipulators [89, 90]. A typical AM is shown in Fig. 5.1. Those manipulators are often relatively heavy and have a substantial dynamic impact on their carrying, generally, under-actuated flying platform. This impact stems from multiple sources, such as the reaction forces and torques related to the movement of the manipulator, the shifting center of gravity related to the changing physical configuration of the manipulator, and contact forces due to the direct interaction with the environment.

The mathematical model describing the tightly coupled dynamics of such systems is a key component, e.g., in deep reinforcement learning, where the neural network may learn how to control a dynamic system based on its accurate simulation [91], but also in traditional, non-linear control. The derivation of the kinematics and dynamics model is a well-studied subject in classical robotics, falling under the category of mobile robotics [92], where the floating base is modeled via a 6-DOF joint. In aerial manipulation, however, the parametrization of rotational DOF often employs Euler angles [93, 43, 89, 94], which gives birth to singularities, or complex dual number formulations [95], which can be hard to implement.

As tasks in aerial manipulation grow in complexity, Euler angles (resp. any three-angle representation of orientations) become a limiting factor as they inherently introduce singularities and consequently reduce the space of operation of the AM. Therefore, we propose a hybrid solution to dynamic modeling. By combining unit quaternion parameterization with the simplicity of the Lagrangian framework, a singularity-free, dynamic model is obtained without resorting to the complexity of dual number formulations resp. the algorithmic complexity of Newton-based methods. The approach shown in this work is kept completely general with the goal to cover most AM configurations. The different steps in our approach are described in detail to facilitate practical implementations.

5.1.1 Related Work

The dynamics of a multi-body system are commonly derived using either Newton or energy-based methods. The recursive Newton-Euler algorithm (RNEA) and the composite-rigid-body algorithm (CRBA) [92] are very popular choices of Newton-based methods. The Euler-Lagrange equations are conceptually much more straightforward and thus easier to implement. However, Newton-based methods often yield more compact solutions [96] at the expense of being algorithmically more complex.

It is common practice to either combine quaternion attitude representation with one of the Newton-based methods or Euler attitude representation with the simpler Lagrangian formulation. The authors of [43, 94] derived the coupled model of an AM with a 2-DOF manipulator using Euler angles and Lagrangian mechanics. The same approach was chosen for a dual 5-DOF arm configuration in [89] and a 3-DOF manipulator in [97]. A dual-quaternion approach employing the Euler-Lagrange formalism has been chosen in [98]. In [99], the authors chose to derive the dynamics of three 2-DOF manipulators using the Newton-Euler method. The authors of [100] followed the same approach, with a focus on inverse kinematics with holonomic constraints. In [93], the dynamics were derived using the Newton-Euler method and Euler angles. In [101], the authors presented an approach within the Lagrangian framework (employing Euler angles) that handles constraints applied to the end-effector by introducing an additional constraint term to the dynamics equation. In this work, the coupled dynamics, using singularity-free quaternion representation, are derived in detail using the simpler Euler-Lagrange equations, thus offering an appealing alternative (resp. middle ground) to the current state of the art.

The state of the art with regard to model-based control techniques can be found in [21]. Some highlights include the robust, adaptive sliding mode controller introduced in [43], the variable parameter integral backstepping controller in [93, 44], the passivity-based controller in [102], the disturbance observer-based control in [100], the hybrid position and force controller respecting constraint conditions presented in [101], and the model reference adaptive control in [103]. The authors of [104] specifically addressed the problem of model uncertainty and unknown disturbances by employing a nonlinear disturbance observer in conjunction with a robust prescribed performance controller. Those control schemes have the dynamic model of the AM at their core while being robust towards model imperfections. This work focuses primarily on modeling the dynamics of AMs. Therefore, only a basic PD controller with a bias correction term gets introduced throughout this work because it does not compensate (thus hide) model imperfections.

5.1.2 Contributions

The main contribution of this chapter is an easy-to-implement yet powerful general-purpose dynamics framework for aerial manipulation that yields a singularity-free, closed-form dynamic model of an AM. Although we do not target tilting propeller resp. fully actuated UAV configurations in this work, the results presented herein are general enough to apply to those configurations as well (assuming that the motion of the tilting mechanism has a negligible impact on the location of the center of gravity).

The contributions of this work are thus stated as follows: 1. Providing a hybrid approach with regard to the dynamic modeling of AMs, by combining quaternion parametrization and

Lagrangian mechanics. This is contrary to what most authors do in this field, namely using either Euler angle parameterization in the Lagrangian framework or quaternion parameterization in the Newton-Euler framework. The approach presented here leads to a completely generic, singularity-free, coupled dynamic model. Implementing our approach requires significantly less coding than Newton-based methods, thanks to the algorithmic simplicity of the Lagrangian framework. The bulk of the work consists in implementing the simple kinematic equations, contrary to the Newton-Euler algorithm, which also involves expressing the respective force and torque equations in a recursive manner [92]. 2. The derivation of the kinematics resp. the dynamic model often comes up short in many publications, where the focus is often on control rather than the modeling. This perceived lack of modeling methodology is addressed herein, showing all the required steps and thus providing insights into the normally hidden process, facilitating the entry into the field of aerial manipulation. 3. The introduction of a *complete* general-purpose dynamics framework for aerial manipulators based on a (widely used) URDF [105] description of an AM. By using the Lagrangian framework and by refraining from using dual quaternions (dual numbers), in combination with the mapping of all quaternion operations to matrix-vector products, the results presented here are particularly well suited to be implemented with common computer algebra systems (CAS). 4. Two applications of the model are shown; *first*, in a simulation resp. validation context where the obtained model is validated against an identical reference model simulated in a real-time physics engine. And *second*, as part of an adaptation of a computed torque controller adopted from classical robotics to work in an aerial manipulation context. Performance figures concerning the algebraic complexity and the real-time capability are provided for two typical scenarios, namely onboard resp. off-board control using increasingly complex AM. We also compare the complexity of the resulting dynamics equations obtained from different methods.

5.1.3 Structure

This chapter is organized as follows. In Section 5.2.1, we start with the kinematics of a single body in the inertial frame, followed by the kinematic equations of an open kinematic chain manipulator with regard to its own base frame in Section 5.2.2. Then, by combining those results (Section 5.2.3), the base of the robot is turned into a floating platform, providing additional 6-DOF, which results in the complete kinematic equations of the manipulating system in the world frame. The kinematic equations are then fed into the Lagrangian framework (Section 5.3) in order to obtain the dynamics equations. It follows the non-trivial mapping between body forces and generalized forces resp. propulsion forces and body forces in Section 5.3.4. Finally, in Section 5.4, a general-purpose holonomic constraint solver is applied to the unconstrained system to impose the quaternion unit constraint via constraint forces. In Section 5.5, the obtained dynamic model is validated in simulation against a real-time physics engine. As an application, a stabilizing controller is introduced in Section 5.6 that uses the developed dynamic model to cancel the nonlinearities present in the AM. We proceed by presenting performance figures regarding the algebraic complexity and the real-time capability of the model within the control context in Section 5.6.3. This work is then summarized and concluded in Section 5.7. The nomenclature is as stated in Table 5.1. To simplify the notation, arguments in variables or functions are dropped whenever possible and if no confusion is likely to occur.

5.2 Kinematics

The following section derives the forward kinematics of the combined (coupled) system formed by a UAV and its manipulator(s). Commonly, the additional DOF of the UAV are accounted for by introducing a 6-DOF joint between the world frame and the base link of the manipulator. In this work, however, it is more convenient to *first* describe the kinematics of each of the two subsystems (drone and manipulator) separately. And *then* to merge both systems as the last step.

5.2.1 UAV Body Kinematics

Given a UAV, represented by base link $\mathtt{L}_0$ and the body $\mathtt{B}_0$ in the kinematic chain as depicted in Fig. 5.2. Its orientation is described by the unit quaternion $\mathbf{q} = (q_w, \bar{\mathbf{q}}_v) \in S^3$, and its

Symbol	Definition
k	A Scalar k
$\bar{\mathbf{p}}$	Vector $\bar{\mathbf{p}}$
$\bar{\mathbf{p}}_{[i]}$	Component i in $\bar{\mathbf{p}} = (p_x, p_y, p_z, p_w)$, depending on the size of the vector
$\bar{\mathbf{p}}_{[ij]}$	Swizzled components i and j of $\bar{\mathbf{p}}$
$\mathtt{W}$	Inertial frame
$\mathtt{B_i}$	Body frame i
$\mathtt{J_i}$	Joint frame i
$\mathtt{L_i}$	Link frame i
${}^{\mathtt{A}}\bar{\mathbf{n}}_{\mathtt{B}}$	A normal vector situated in $\mathtt{B}$ expressed in $\mathtt{A}$
${}^{\mathtt{A}}\bar{\mathbf{p}}^{\mathtt{C}}_{\mathtt{D}}$	A vector that spans from $\mathtt{C}$ to $\mathtt{D}$, in $\mathtt{A}$
${}^{\mathtt{A}}\bar{\boldsymbol{\omega}}^{\mathtt{C}}_{\mathtt{D}}$	Angular velocity of $\mathtt{D}$, relative to $\mathtt{C}$, in $\mathtt{A}$
${}^{\mathtt{A}}\bar{\mathbf{v}}^{\mathtt{C}}_{\mathtt{D}}$	Linear velocity of $\mathtt{D}$, relative to $\mathtt{C}$, in $\mathtt{A}$
$\mathbf{A}$	Matrix $\mathbf{A}$
${}^{\mathtt{W}}\mathbf{R}_{\mathtt{B}}$	Rotation matrix mapping from $\mathtt{B}$ to $\mathtt{W}$
${}^{\mathtt{W}}\mathbf{A}_{\mathtt{B}}$	Homogeneous transformation mapping from $\mathtt{B}$ to $\mathtt{W}$
$\mathbf{q}$	Quaternion
N_J	Number of joints
N_B	Number of bodies
N_L	Number of links
N_x	Number of coordinates

Table 5.1: Notation and frequently used symbols. The notation is kept throughout this thesis.

position in the world frame `W` is given by the vector

$$ {}^{\mathtt{W}}\bar{\mathbf{p}}^{\mathtt{W}}_{\mathtt{L_0}} = [x, y, z] \in \mathbb{R}^3. \tag{5.1} $$

The base link's angular velocity ${}^{\mathtt{W}}\bar{\boldsymbol{\omega}}^{\mathtt{W}}_{\mathtt{L_0}} \in \mathbb{R}^3$, expressed in `W`, relative to `W`, is tied to the quaternion $\mathbf{q}$ via

$$ \left(0, {}^{\mathtt{W}}\bar{\boldsymbol{\omega}}^{\mathtt{W}}_{\mathtt{L_0}}\right) = 2\dot{\mathbf{q}} \otimes \mathbf{q}^*. \tag{5.2} $$

Which can be written in matrix form using (3.16)

$$ {}^{\mathtt{W}}\boldsymbol{\omega}^{\mathtt{W}}_{\mathtt{L_0}} = 2\left[\mathbf{q}\right]_R^{\intercal} \dot{\mathbf{q}}. \tag{5.3} $$

Similarly, the base link's angular velocity expressed in `L₀`, relative to `W` can be written as

$$ \left(0, {}^{\mathtt{L_0}}\bar{\boldsymbol{\omega}}^{\mathtt{W}}_{\mathtt{L_0}}\right) = 2\mathbf{q}^* \otimes \dot{\mathbf{q}}, \tag{5.4} $$

respectively with the help of (3.15)

$$ {}^{\mathtt{L_0}}\boldsymbol{\omega}^{\mathtt{W}}_{\mathtt{L_0}} = 2\left[\mathbf{q}\right]_L^{\intercal} \dot{\mathbf{q}}. \tag{5.5} $$

Since ${}^{\mathtt{W}}\boldsymbol{\omega}^{\mathtt{W}}_{\mathtt{L_0}}$ and ${}^{\mathtt{L_0}}\boldsymbol{\omega}^{\mathtt{W}}_{\mathtt{L_0}}$ are pure quaternions, their vector part can directly be obtained by dropping the first row from (5.3) resp. (5.5) yielding the two orthogonal mapping matrices **E** and **G** defined by

$$ {}^{\mathtt{W}}\bar{\boldsymbol{\omega}}^{\mathtt{W}}_{\mathtt{L_0}} = 2 \underbrace{\begin{bmatrix} -\bar{\mathbf{q}}_v & q_w \mathbf{1}_3 + \left[\bar{\mathbf{q}}_v\right]_\times \end{bmatrix}}_{\mathbf{E} \in \mathbb{R}^{3\times 4}} \dot{\mathbf{q}} \tag{5.6} $$

resp.

$$ {}^{\mathtt{L_0}}\bar{\boldsymbol{\omega}}^{\mathtt{W}}_{\mathtt{L_0}} = 2 \underbrace{\begin{bmatrix} -\bar{\mathbf{q}}_v & q_w \mathbf{1}_3 - \left[\bar{\mathbf{q}}_v\right]_\times \end{bmatrix}}_{\mathbf{G} \in \mathbb{R}^{3\times 4}} \dot{\mathbf{q}}. \tag{5.7} $$

Solving (5.7) for $\dot{\mathbf{q}}$ and replacing it into (5.6) reveals the relation between ${}^{\mathtt{W}}\boldsymbol{\omega}^{\mathtt{W}}_{\mathtt{L_0}}$ and ${}^{\mathtt{L_0}}\boldsymbol{\omega}^{\mathtt{W}}_{\mathtt{L_0}}$:

$$
{}^{\mathtt{W}}\bar{\boldsymbol{\omega}}^{\mathtt{W}}_{\mathtt{L_0}} = \underbrace{\mathbf{E}\mathbf{G}^{\intercal}}_{{}^{\mathtt{W}}\mathbf{R}_{\mathtt{L_0}}} {}^{\mathtt{L_0}}\bar{\boldsymbol{\omega}}^{\mathtt{W}}_{\mathtt{L_0}}, \tag{5.8}
$$

where ${}^{\mathtt{W}}\mathbf{R}_{\mathtt{L_0}} \in SO\,(3)$ is the orthogonal rotation matrix associated with $\mathbf{q}$.

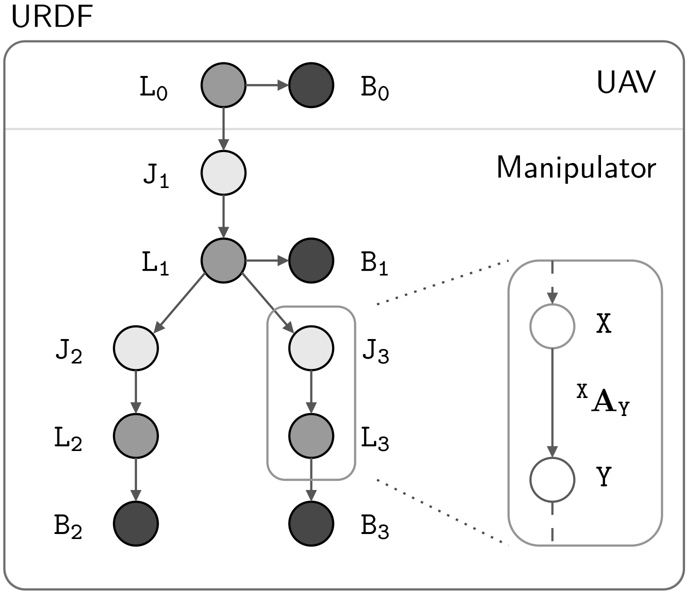


Figure 5.2: Example of a URDF-compliant kinematic tree of a branched kinematic chain. The link and joint frames are annotated as $\mathtt{L}_i$ resp. $\mathtt{J}_i$. The body frame $\mathtt{B_i}$ has an associated mass and moment of inertia. It marks the center of mass of its parent link $\mathtt{J}_i$. $\mathtt{L_0}$ marks the base link of the robot (namely the UAV). The edges between the nodes (see detail) represent coordinate transformations between two frames denoted by ${}^{\mathtt{X}}\mathbf{A}_{\mathtt{Y}}$.

5.2.2 Kinematics of the Manipulator

In the following section, the forward kinematics of an open-chain manipulator is developed. Although the base link of the manipulator is represented by the UAV, for the development in this section, the base link $\mathtt{L_0}$ is regarded as static and all quantities of motion are expressed in and with regard to $\mathtt{L_0}$ (Fig. 5.2). The development largely follows the procedures found in classical robotics literature [106, 107, 92] and [108] with some changes to the kinematic tree that are required to interface with the Universal Robot Description Format specifications seamlessly.

A generic high-level description of a robotic arm can be stated in form of a URDF, providing all required kinematic and physical properties of a robotic manipulator. Its conventions concerning reference frames, links, joints, and parents have been adopted here to facilitate the translation from an AM given as a URDF file to its system dynamics.

The URDF structure is standardized and describes an (open) kinematic chain formed by various types of nodes. Nodes are defined as either joints $\mathtt{J_i}$, links $\mathtt{L_i}$ or bodies $\mathtt{B_i}$ (Fig. 5.2), and are related to each other by homogeneous transformations such as

$$ {}^{\mathtt{X}}\mathbf{A}_{\mathtt{Y}} = \begin{bmatrix} {}^{\mathtt{X}}\mathbf{R}_{\mathtt{Y}} & {}^{\mathtt{X}}\bar{\mathbf{p}}_{\mathtt{Y}}^{\mathtt{X}} \\ \bar{\mathbf{0}} & 1 \end{bmatrix} \in SE(3). \tag{5.9} $$

The key aspects of the URDF are summarized as follows (refer to [105] for further details):

- The kinematic chain starts with the base link $\mathtt{L_0}$.
- Each link can be the parent of multiple joints, thus creating branched structures.
- Each joint is the parent of the single link.
- Each link has an associated (rigid) body. The body frame $\mathtt{B_i}$ defines the center of mass (CM) of the body having the mass m_i and the moment of inertia $\mathbf{\Phi}_i$.
- The transformation (edge) between two nodes is defined by a homogeneous transformation ${}^{\mathtt{X}}\mathbf{A}_{\mathtt{Y}}$.

Those physical properties m_i, $\mathbf{\Phi}_i$ and ${}^{\mathtt{X}}\mathbf{A}_{\mathtt{Y}}$ are best obtained directly from CAD data.

Let there be a manipulator with $J = \{\mathtt{J_1}, \mathtt{J_2}, \dots, \mathtt{J_N}\}$ joints, $L = \{\mathtt{L_0}, \mathtt{L_1}, \dots, \mathtt{L_N}\}$ links and $B = \{\mathtt{L_0}, \mathtt{L_1}, \dots, \mathtt{L_N}\}$ bodies forming a kinematic tree of nodes e.g. as shown in Fig. 5.2. To traverse the kinematic tree, let there be a function $p(i)$ that retrieves an ordered list of all parents of node i starting at node i and ending with the base link $\mathtt{L_0}$.

The transformation of each node i in the tree with regard to $\mathtt{L_0}$ is given by the product

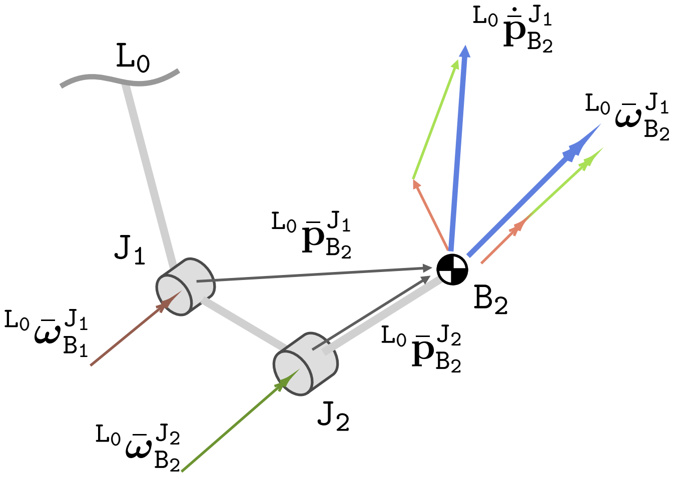

$\mathtt{L_0}$
${}^{\mathtt{L_0}}\dot{\bar{\mathbf{p}}}^{\mathtt{J_1}}_{\mathtt{B_2}}$
${}^{\mathtt{L_0}}\bar{\boldsymbol{\omega}}^{\mathtt{J_1}}_{\mathtt{B_2}}$
$\mathtt{J_1}$
${}^{\mathtt{L_0}}\bar{\mathbf{p}}^{\mathtt{J_1}}_{\mathtt{B_2}}$
$\mathtt{B_2}$
${}^{\mathtt{L_0}}\bar{\boldsymbol{\omega}}^{\mathtt{J_1}}_{\mathtt{B_1}}$
${}^{\mathtt{L_0}}\bar{\mathbf{p}}^{\mathtt{J_2}}_{\mathtt{B_2}}$
$\mathtt{J_2}$
${}^{\mathtt{L_0}}\bar{\boldsymbol{\omega}}^{\mathtt{J_2}}_{\mathtt{B_2}}$

Figure 5.3: Kinematics of a 2 DOF manipulator showing the angular and tangential velocity of the body frame $\mathtt{B_2}$ due to the movement of the joints $\mathtt{J_1}$ and $\mathtt{J_2}$.

of each local transformation $\mathbf{A}$ between the node i and $\mathtt{L_0}$

$$ {}^{\mathtt{L_0}}\mathbf{A}_{\mathtt{i}} = \prod_{\mathbf{A}\in p(i)} \mathbf{A}. \tag{5.10} $$

Generally, ${}^{\mathtt{L_i}}\mathbf{A}_{\mathtt{B_i}}$, ${}^{\mathtt{L_i}}\mathbf{A}_{\mathtt{J_i}}$ are constants defined by the physical configuration of the robot. On the other hand, ${}^{\mathtt{J_i}}\mathbf{A}_{\mathtt{L_i}}$ is a function of the joint position. For revolute joints, ${}^{\mathtt{J_i}}\mathbf{A}_{\mathtt{L_i}}\left(\theta_i\right)$ designates a rotation around the local joint axis ${}^{\mathtt{J_i}}\bar{\mathbf{n}}_{\mathtt{J_i}} \in \mathbb{R}^3, \|\bar{\mathbf{n}}\| = 1$ (e.g., the local z-axis), with an angle θ_i, which corresponds to (3.19) and can be encapsulated in the homogeneous transformation

$$ {}^{\mathtt{J_i}}\mathbf{A}_{\mathtt{L_i}}\left(\theta_i\right) = \begin{bmatrix} R\left(\theta_i, {}^{\mathtt{J_i}}\bar{\mathbf{n}}_{\mathtt{J_i}}\right) & \bar{\mathbf{0}} \\ \bar{\mathbf{0}} & 1 \end{bmatrix}. \tag{5.11} $$

Although the focus is on revolute joints, a completely analogous procedure can be followed for prismatic joints.

The position of each node with respect to $\mathtt{L_0}$ is obtained by the relation

$$ \begin{bmatrix} {}^{\mathtt{L_0}}\bar{\mathbf{p}}^{\mathtt{L_0}}_{\mathtt{i}} \\ 1 \end{bmatrix} = {}^{\mathtt{L_0}}\mathbf{A}_{\mathtt{i}} \begin{bmatrix} \bar{\mathbf{0}} \\ 1 \end{bmatrix}. \tag{5.12} $$

The angular velocity of $\mathtt{B_i}$ relative to its parent joint $\mathtt{J_i}$ and expressed in $\mathtt{J_i}$ is denoted as ${}^{\mathtt{J_i}}\bar{\boldsymbol{\omega}}^{\mathtt{J_i}}_{\mathtt{B_i}}$ (Fig. 5.3), which can be expressed in terms of its rotation axis and angular velocity

$$ {}^{\mathtt{J_i}}\bar{\boldsymbol{\omega}}^{\mathtt{J_i}}_{\mathtt{B_i}} = {}^{\mathtt{J_i}}\bar{\mathbf{n}}_{\mathtt{J_i}}\dot{\theta}_{\mathtt{J_i}}. \tag{5.13} $$

It is clear that the body frame $\mathtt{B_i}$, and the link $\mathtt{L_i}$ must have the same angular velocity since ${}^{\mathtt{B_i}}\mathbf{A}_{\mathtt{L_i}}$ is constant, therefore ${}^{\mathtt{J_i}}\bar{\boldsymbol{\omega}}^{\mathtt{J_i}}_{\mathtt{L_i}} = {}^{\mathtt{J_i}}\bar{\boldsymbol{\omega}}^{\mathtt{J_i}}_{\mathtt{B_i}}$.

By multiplying with the rotational part ${}^{\mathtt{L_0}}\mathbf{R}_{\mathtt{J_i}}$ of ${}^{\mathtt{L_0}}\mathbf{A}_{\mathtt{J_i}}$, the angular velocity is obtained with respect to $\mathtt{L_0}$:

$$ {}^{\mathtt{L_0}}\bar{\boldsymbol{\omega}}^{\mathtt{J_i}}_{\mathtt{B_i}} = {}^{\mathtt{L_0}}\mathbf{R}_{\mathtt{J_i}}\,{}^{\mathtt{J_i}}\bar{\boldsymbol{\omega}}^{\mathtt{J_i}}_{\mathtt{B_i}}. \tag{5.14} $$

Using (5.13), it follows

$$ {}^{\mathtt{L_0}}\bar{\boldsymbol{\omega}}^{\mathtt{J_i}}_{\mathtt{B_i}} = {}^{\mathtt{L_0}}\mathbf{R}_{\mathtt{J_i}}\,{}^{\mathtt{J_i}}\bar{\mathbf{n}}_{\mathtt{J_i}}\dot{\theta}_{\mathtt{J_i}}. \tag{5.15} $$

With all angular velocities expressed in the same frame of reference, the total angular velocity (relative to the base link) of each body is obtained by the sum of the angular velocities (5.15) over all of its ancestors:

$$ {}^{\mathtt{L_0}}\bar{\boldsymbol{\omega}}^{\mathtt{L_0}}_{\mathtt{B_i}} = \sum_{\mathtt{J_j}\in p(B_i)} {}^{\mathtt{L_0}}\bar{\mathbf{n}}_{\mathtt{J_j}}\dot{\theta}_{\mathtt{J_j}}. \tag{5.16} $$

Rewriting (5.16) as matrix-vector product then yields the Jacobian for rotation ${}^{\mathtt{L_0}}\mathbf{J}_{\omega,\mathtt{i}}$ for the i-th body in the chain. The k-th element of that matrix is given by

$$ {}^{\mathtt{L_0}}\mathbf{J}_{\omega,\mathtt{i},\mathtt{k}} = \begin{cases} {}^{\mathtt{L_0}}\bar{\mathbf{n}}_{\mathtt{J_k}}, & \text{if } \mathtt{J_k} \in p\left(B_i\right) \\ \bar{\mathbf{0}} & \text{otherwise} \end{cases}. \tag{5.17} $$

Noncontributing joints are thus accounted for by a $\bar{\mathbf{0}}$ contribution.

The linear, sectional velocity of the body ${}^{\mathtt{L_0}}\dot{\bar{\mathbf{p}}}^{\mathtt{J_i}}_{\mathtt{B_i}}$ due to the rotational movement ${}^{\mathtt{J_j}}\mathbf{A}_{\mathtt{B_i}}$ of one of its ancestral joints $\mathtt{J}_j \in p(\mathtt{B}_i)$ is given by the tangential velocity

$$ {}^{\mathtt{L_0}}\dot{\bar{\mathbf{p}}}^{\mathtt{J_j}}_{\mathtt{B_i}} = {}^{\mathtt{L_0}}\bar{\boldsymbol{\omega}}^{\mathtt{J_j}}_{\mathtt{B_i}} \times \underbrace{\left({}^{\mathtt{L_0}}\bar{\mathbf{p}}^{\mathtt{L_0}}_{\mathtt{J_j}} - {}^{\mathtt{L_0}}\bar{\mathbf{p}}^{\mathtt{L_0}}_{\mathtt{B_i}}\right)}_{{}^{\mathtt{L_0}}\bar{\mathbf{p}}^{\mathtt{J_j}}_{\mathtt{B_i}}}, \tag{5.18} $$

where ${}^{\mathtt{L_0}}\bar{\mathbf{p}}_{\mathtt{B_i}}^{\mathtt{J_j}}$ defines the lever formed between the frames $\mathtt{J_j}$ and $\mathtt{B_i}$ (see Fig. 5.3). By using identity (5.14) and defining ${}^{\mathtt{L_0}}\bar{\mathbf{r}}_{\mathtt{B_i}}^{\mathtt{J_j}} = {}^{\mathtt{L_0}}\bar{\mathbf{n}}_j \times {}^{\mathtt{L_0}}\bar{\mathbf{p}}_{\mathtt{B_i}}^{\mathtt{J_j}}$, (5.18) can be written as

$$
{}^{\mathtt{L_0}}\dot{\bar{\mathbf{p}}}_{\mathtt{B_i}}^{\mathtt{J_j}} = {}^{\mathtt{L_0}}\bar{\mathbf{r}}_{\mathtt{B_i}}^{\mathtt{J_j}}\dot{\theta}_{\mathtt{J_j}}. \tag{5.19}
$$

With all of the sectional velocities expressed with regard to $\mathtt{L_0}$, adding them yields the total linear velocity of $\mathtt{B_i}$

$$
{}^{\mathtt{L_0}}\dot{\bar{\mathbf{p}}}_{\mathtt{B_i}}^{\mathtt{L_0}} = \sum_{\mathtt{J_j}\in p(B_i)} {}^{\mathtt{L_0}}\bar{\mathbf{r}}_{\mathtt{B_i}}^{\mathtt{J_j}}\dot{\theta}_{\mathtt{J_j}}, \tag{5.20}
$$

This sum can also be written as ${}^{\mathtt{L_0}}\dot{\bar{\mathbf{p}}}_{\mathtt{B_i}}^{\mathtt{L_0}} = {}^{\mathtt{L_0}}\mathbf{J}_{\mathtt{t,i}}\bar{\theta}$, where ${}^{\mathtt{L_0}}\mathbf{J}_{\mathtt{t,i}}$ is the Jacobian matrix for translation. The k-th element of that matrix is given by

$$
{}^{\mathtt{L_0}}\mathbf{J}_{\mathtt{t,i,k}} = \begin{cases} {}^{\mathtt{L_0}}\bar{\mathbf{r}}_{\mathtt{B_i}}^{\mathtt{J_k}}, & \text{if } \mathtt{J_k} \in p\left(B_i\right) \\ \bar{\mathbf{0}} & \text{otherwise} \end{cases}. \tag{5.21}
$$

Joints that are not an ancestor of $\mathtt{B_i}$ do not add to its velocity, and their contribution is thus $\bar{\mathbf{0}}$.

Eq. (5.19) and (5.16) can thus be expressed in terms of the Jacobian matrices (5.17), (5.21). The total linear velocity is given by

$$
{}^{\mathtt{L_0}}\dot{\bar{\mathbf{p}}}_{\mathtt{B_i}}^{\mathtt{L_0}} = {}^{\mathtt{L_0}}\mathbf{J}_{\mathtt{t,i}}\dot{\bar{\theta}}, \tag{5.22}
$$

with the total angular velocity ${}^{\mathtt{L_0}}\bar{\boldsymbol{\omega}}_{\mathtt{B_i}}$ of body i being

$$
{}^{\mathtt{L_0}}\bar{\boldsymbol{\omega}}_{\mathtt{B_i}}^{\mathtt{L_0}} = {}^{\mathtt{L_0}}\mathbf{J}_{\omega,\mathtt{i}}\dot{\bar{\theta}}, \tag{5.23}
$$

where $\bar{\theta} = \left[\theta_{\mathtt{J_1}}, \dots, \theta_{\mathtt{J_N}}\right] \in \mathbb{R}^{N_J}$ is the vector of all joint positions.

5.2.3 Kinematics of the Manipulator attached to the flying Base

In the previous section, the kinematics of the manipulator was derived with regard to $\mathtt{L_0}$. Attaching the manipulator to a floating base (the UAV here) endows the system with an additional 6 DOF. To account for those 6 DOF, it suffices to describe the forward kinematics of the manipulator with regard to the inertial frame $\mathtt{W}$, knowing that the floating base is located at a position $\bar{\mathbf{p}}$ from the origin and that its orientation with regard to $\mathtt{W}$ is given by the unit quaternion $\mathbf{q}$, essentially inserting a virtual 6-DOF joint in between $\mathtt{W}$ and $\mathtt{L_0}$ that is parametrized by $\bar{\mathbf{p}}$ and $\mathbf{q}$.

Looking at the system from $\mathtt{W}$, it is clear that the position ${}^{\mathtt{W}}\bar{\mathbf{p}}^{\mathtt{L_0}}_{\mathtt{B_i}}$ of each body of the manipulator in $\mathtt{W}$ with regard to $\mathtt{L_0}$ is superimposed by the position ${}^{\mathtt{W}}\bar{\mathbf{p}}^{\mathtt{W}}_{\mathtt{L_0}}$ of the floating base, resulting in the absolute link position

$$ {}^{\mathtt{W}}\bar{\mathbf{p}}^{\mathtt{W}}_{\mathtt{B_i}} = {}^{\mathtt{W}}\bar{\mathbf{p}}^{\mathtt{W}}_{\mathtt{L_0}} + {}^{\mathtt{W}}\bar{\mathbf{p}}^{\mathtt{L_0}}_{\mathtt{B_i}}. \tag{5.24} $$

Each link's position in $\mathtt{W}$ is obtained by rotating it by $\mathbf{q}$, such as:

$$ \left(0, {}^{\mathtt{W}}\bar{\mathbf{p}}^{\mathtt{L_0}}_{\mathtt{B_i}}\right) = \mathbf{q} \otimes \left(0, {}^{\mathtt{L_0}}\bar{\mathbf{p}}^{\mathtt{L_0}}_{\mathtt{B_i}}\right) \otimes \mathbf{q}^*. \tag{5.25} $$

The link velocity is then obtained from the time derivative of its position vector (5.25)

$$ {}^{\mathtt{W}}\dot{\bar{\mathbf{p}}}^{\mathtt{W}}_{\mathtt{B_i}} = \frac{d}{dt}\left({}^{\mathtt{W}}\bar{\mathbf{p}}^{\mathtt{W}}_{\mathtt{B_i}}\right). \tag{5.26} $$

Inserting (5.25) in (5.26) yields

$$ \begin{aligned} \left(0, {}^{\mathtt{W}}\dot{\bar{\mathbf{p}}}^{\mathtt{W}}_{\mathtt{B_i}}\right) = {} & \left(0, {}^{\mathtt{W}}\dot{\bar{\mathbf{p}}}^{\mathtt{W}}_{\mathtt{L_0}}\right) \\ & + \mathbf{q} \otimes \left(0, {}^{\mathtt{L_0}}\dot{\bar{\mathbf{p}}}^{\mathtt{L_0}}_{\mathtt{B_i}}\right) \otimes \mathbf{q}^* \\ & + \dot{\mathbf{q}} \otimes \left(0, {}^{\mathtt{L_0}}\bar{\mathbf{p}}^{\mathtt{L_0}}_{\mathtt{B_i}}\right) \otimes \mathbf{q}^* \\ & + \mathbf{q} \otimes \left(0, {}^{\mathtt{L_0}}\bar{\mathbf{p}}^{\mathtt{L_0}}_{\mathtt{B_i}}\right) \otimes \dot{\mathbf{q}}^*. \end{aligned} \tag{5.27} $$

By multiplying the left resp. right side of the last two terms in (5.27) by $\mathbf{1} = \mathbf{q} \otimes \mathbf{q}^*$ the following expression is obtained:

$$
\begin{aligned}
\left(0, {}^{\mathtt{W}}\dot{\bar{\mathbf{p}}}^{\mathtt{W}}_{\mathtt{B_i}}\right) = {} & \left(0, {}^{\mathtt{W}}\dot{\bar{\mathbf{p}}}^{\mathtt{W}}_{\mathtt{L_0}}\right) \\
& + \mathbf{q} \otimes \left(0, {}^{\mathtt{L_0}}\dot{\bar{\mathbf{p}}}^{\mathtt{L_0}}_{\mathtt{B_i}}\right) \otimes \mathbf{q}^* \\
& + \mathbf{q} \otimes \left[\mathbf{q}^* \otimes \dot{\mathbf{q}} \otimes \left(0, {}^{\mathtt{L_0}}\bar{\mathbf{p}}^{\mathtt{L_0}}_{\mathtt{B_i}}\right)\right. \\
& + \left.\left(0, {}^{\mathtt{L_0}}\bar{\mathbf{p}}^{\mathtt{L_0}}_{\mathtt{B_i}}\right) \otimes \dot{\mathbf{q}}^* \otimes \mathbf{q}\right] \otimes \mathbf{q}^*.
\end{aligned}
\tag{5.28}
$$

Applying (5.4) yields

$$
\begin{aligned}
\left(0, {}^{\mathtt{W}}\dot{\bar{\mathbf{p}}}^{\mathtt{W}}_{\mathtt{B_i}}\right) = {} & \left(0, {}^{\mathtt{W}}\dot{\bar{\mathbf{p}}}^{\mathtt{W}}_{\mathtt{L_0}}\right) \\
& + \mathbf{q} \otimes \left(0, {}^{\mathtt{L_0}}\dot{\bar{\mathbf{p}}}^{\mathtt{L_0}}_{\mathtt{B_i}}\right) \otimes \mathbf{q}^* \\
& + \mathbf{q} \otimes \left[\left(0, \frac{1}{2}{}^{\mathtt{L_0}}\bar{\boldsymbol{\omega}}^{\mathtt{L_0}}_{\mathtt{L_0}}\right) \otimes \left(0, {}^{\mathtt{L_0}}\bar{\mathbf{p}}^{\mathtt{L_0}}_{\mathtt{B_i}}\right)\right. \\
& + \left.\left(0, {}^{\mathtt{L_0}}\bar{\mathbf{p}}^{\mathtt{L_0}}_{\mathtt{B_i}}\right) \otimes \left(0, -\frac{1}{2}{}^{\mathtt{L_0}}\bar{\boldsymbol{\omega}}^{\mathtt{L_0}}_{\mathtt{L_0}}\right)\right] \otimes \mathbf{q}^*.
\end{aligned}
\tag{5.29}
$$

Evaluating the two pure quaternion products using (3.3) then results in

$$
\begin{aligned}
\left(0, {}^{\mathtt{W}}\dot{\bar{\mathbf{p}}}^{\mathtt{W}}_{\mathtt{B_i}}\right) = {} & \left(0, {}^{\mathtt{W}}\dot{\bar{\mathbf{p}}}^{\mathtt{W}}_{\mathtt{L_0}}\right) \\
& + \mathbf{q} \otimes \left(0, {}^{\mathtt{L_0}}\dot{\bar{\mathbf{p}}}^{\mathtt{L_0}}_{\mathtt{B_i}}\right) \otimes \mathbf{q}^* \\
& + \mathbf{q} \otimes \frac{1}{2}\left[\left(-{}^{\mathtt{L_0}}\bar{\boldsymbol{\omega}}^{\mathtt{L_0}}_{\mathtt{L_0}}{}^{\mathtt{L_0}}\bar{\mathbf{p}}^{\mathtt{L_0}}_{\mathtt{B_i}}, {}^{\mathtt{L_0}}\bar{\boldsymbol{\omega}}^{\mathtt{L_0}}_{\mathtt{L_0}} \times {}^{\mathtt{L_0}}\bar{\mathbf{p}}^{\mathtt{L_0}}_{\mathtt{B_i}}\right)\right. \\
& + \left.\left({}^{\mathtt{L_0}}\bar{\boldsymbol{\omega}}^{\mathtt{L_0}}_{\mathtt{L_0}}{}^{\mathtt{L_0}}\bar{\mathbf{p}}^{\mathtt{L_0}}_{\mathtt{B_i}}, {}^{\mathtt{L_0}}\bar{\boldsymbol{\omega}}^{\mathtt{L_0}}_{\mathtt{L_0}} \times {}^{\mathtt{L_0}}\bar{\mathbf{p}}^{\mathtt{L_0}}_{\mathtt{B_i}}\right)\right] \otimes \mathbf{q}^*,
\end{aligned}
\tag{5.30}
$$

which further simplifies to

$$
\begin{aligned}
\left(0, {}^{\mathtt{W}}\dot{\bar{\mathbf{p}}}^{\mathtt{W}}_{\mathtt{B_i}}\right) = {} & \left(0, {}^{\mathtt{W}}\dot{\bar{\mathbf{p}}}^{\mathtt{W}}_{\mathtt{L_0}}\right) \\
& + \mathbf{q} \otimes \left(0, {}^{\mathtt{L_0}}\dot{\bar{\mathbf{p}}}^{\mathtt{L_0}}_{\mathtt{B_i}}\right) \otimes \mathbf{q}^* \\
& + \mathbf{q} \otimes \left[\left(0, {}^{\mathtt{L_0}}\bar{\boldsymbol{\omega}}^{\mathtt{L_0}}_{\mathtt{L_0}} \times {}^{\mathtt{L_0}}\bar{\mathbf{p}}^{\mathtt{L_0}}_{\mathtt{B_i}}\right)\right] \otimes \mathbf{q}^*.
\end{aligned}
\tag{5.31}
$$

Using (5.6), (5.7), (5.8) and (3.17), it can further be transformed into

$$\begin{aligned} {}^{\mathtt{W}}\dot{\bar{\mathbf{p}}}^{\mathtt{W}}_{\mathtt{B_i}} &= {}^{\mathtt{W}}\dot{\bar{\mathbf{p}}}^{\mathtt{W}}_{\mathtt{L_0}} \\ &+ {}^{\mathtt{W}}\mathbf{R}_{\mathtt{L_0}}\, {}^{\mathtt{L_0}}\dot{\bar{\mathbf{p}}}^{\mathtt{L_0}}_{\mathtt{B_i}} \\ &- {}^{\mathtt{W}}\mathbf{R}_{\mathtt{L_0}} \left[{}^{\mathtt{L_0}}\bar{\mathbf{p}}^{\mathtt{L_0}}_{\mathtt{B_i}}\right]_{\times} {}^{\mathtt{L_0}}\bar{\boldsymbol{\omega}}^{\mathtt{L_0}}_{\mathtt{L_0}}. \end{aligned} \tag{5.32}$$

Introducing (5.7) and (5.22) finally yields the complete equation of the linear velocity of $\mathtt{B_i}$ in the inertial frame

$$\begin{aligned} {}^{\mathtt{W}}\dot{\bar{\mathbf{p}}}^{\mathtt{W}}_{\mathtt{B_i}} &= {}^{\mathtt{W}}\dot{\bar{\mathbf{p}}}^{\mathtt{W}}_{\mathtt{L_0}} && \text{(base translation)} \\ &+ {}^{\mathtt{W}}\mathbf{R}_{\mathtt{L_0}}\, {}^{\mathtt{L_0}}\mathbf{J}_{\mathtt{t,i}}\dot{\bar{\theta}} && \text{(joint rotation)} \\ &- 2\,{}^{\mathtt{W}}\mathbf{R}_{\mathtt{L_0}} \left[{}^{\mathtt{L_0}}\bar{\mathbf{p}}^{\mathtt{L_0}}_{\mathtt{B_i}}\right]_{\times} \mathbf{G}\dot{\mathbf{q}} && \text{(base rotation)} \end{aligned} \tag{5.33}$$

showing the contributions of the moving base and the rotating links. In particular, the last term shows the component of the translational velocity due to the lever ${}^{\mathtt{L_0}}\bar{\mathbf{p}}^{\mathtt{L_0}}_{\mathtt{B,i}}$ and the angular velocity of the base.

The angular velocity of the bodies is simply the superposition of their own angular velocity and that of the base link in the inertial frame:

$$\left(0, {}^{\mathtt{W}}\bar{\boldsymbol{\omega}}^{\mathtt{W}}_{\mathtt{B_i}}\right) = \left(0, {}^{\mathtt{W}}\bar{\boldsymbol{\omega}}^{\mathtt{W}}_{\mathtt{L_0}}\right) + \mathbf{q} \otimes \left(0, {}^{\mathtt{L_0}}\bar{\boldsymbol{\omega}}^{\mathtt{L_0}}_{\mathtt{B_i}}\right) \otimes \mathbf{q}^{*}. \tag{5.34}$$

By inserting (5.3), (5.23) and writing the equation in matrix form then yields the expression for $\mathtt{B_i}$'s angular velocity in world frame as the 4-vector

$$\begin{bmatrix} 0 \\ {}^{\mathtt{W}}\bar{\boldsymbol{\omega}}^{\mathtt{W}}_{\mathtt{B_i}} \end{bmatrix} = 2\left[\mathbf{q}\right]^{\mathtt{T}}_{R}\dot{\mathbf{q}} + \begin{bmatrix} 0 \\ {}^{\mathtt{W}}\mathbf{R}_{\mathtt{L_0}}\, {}^{\mathtt{L_0}}\mathbf{J}_{\omega,\mathtt{i}} \end{bmatrix}\dot{\bar{\theta}}. \tag{5.35}$$

5.3 System Dynamics

In the following section, the direct dynamics are derived from the kinematic equations (5.33) and (5.35) by leveraging the Lagrangian framework. Furthermore, the relation between the body- and generalized forces is established from the principle of virtual work. Lastly, those body forces are defined in the context of an AM.

5.3.1 State Space and System Jacobian

The state space $\bar{\mathbf{z}}$ of the combined system is defined by the choice of the (generalized) coordinates, namely the orientation given by the quaternion $\mathbf{q}$, the position $\bar{\mathbf{p}}$ and the N_J joint angles collected in $\bar{\theta}$:

$$\bar{\mathbf{z}} = \begin{bmatrix} \bar{\mathbf{x}} & \dot{\bar{\mathbf{x}}} \end{bmatrix}, \tag{5.36}$$

with

$$\bar{\mathbf{x}} = \begin{bmatrix} \bar{\mathbf{p}} & \mathbf{q} & \bar{\theta} \end{bmatrix} \in \mathbb{R}^{7+N_J}. \tag{5.37}$$

The rotational and translational velocity of each particle of the system can be written in form of $\bar{\mathbf{v}} = {}^{\mathtt{W}}\mathbf{J}\left(\bar{\mathbf{x}}\right)\dot{\bar{\mathbf{x}}}$, where $\bar{\mathbf{v}}$ is the vector of angular and linear velocities of each body of the system:

$$\bar{\mathbf{v}} = \left[{}^{\mathtt{W}}\dot{\bar{\mathbf{p}}}^{\mathtt{W}}_{\mathtt{B_0}}, \ldots, {}^{\mathtt{W}}\dot{\bar{\mathbf{p}}}^{\mathtt{W}}_{\mathtt{B_{N_B-1}}}, \left(0, {}^{\mathtt{W}}\dot{\bar{\boldsymbol{\omega}}}^{\mathtt{W}}_{\mathtt{B_0}}\right), \ldots \left(0, {}^{\mathtt{W}}\dot{\bar{\boldsymbol{\omega}}}^{\mathtt{W}}_{\mathtt{B_{N_B-1}}}\right)\right]. \tag{5.38}$$

${}^{\mathtt{W}}\mathbf{J}$ is the system's Jacobian matrix which is composed of the individual components of (5.33) and (5.35):

$${}^{\mathtt{W}}\mathbf{J} = \begin{bmatrix} {}^{\mathtt{W}}\mathbf{J}_{\mathtt{t,0}} \\ \vdots \\ {}^{\mathtt{W}}\mathbf{J}_{\mathtt{t,N}} \\ {}^{\mathtt{W}}\mathbf{J}_{\omega,\mathtt{0}} \\ \vdots \\ {}^{\mathtt{W}}\mathbf{J}_{\mathtt{t,N}} \end{bmatrix} \in \mathbb{R}^{7N_B \times (7+N_J)}, \tag{5.39}$$

with the individual Jacobian matrices for translation being

$${}^{\mathtt{W}}\mathbf{J}_{\mathtt{t,i}} = \left[\begin{array}{c|c|c} \mathbf{1}_3 & -2{}^{\mathtt{W}}\mathbf{R}_{\mathtt{L_0}}\left[{}^{\mathtt{L_0}}\bar{\mathbf{p}}^{\mathtt{L_0}}_{\mathtt{B_i}}\right]_{\times}\mathbf{G} & {}^{\mathtt{W}}\mathbf{R}_{\mathtt{L_0}}{}^{\mathtt{L_0}}\mathbf{J}_{\mathtt{t,i}} \end{array}\right] \in \mathbb{R}^{3\times(7+N_J)}, \tag{5.40}$$

and for rotation being

$$ {}^{\mathtt{W}}\mathbf{J}_{\omega,\mathtt{i}} = \left[\begin{array}{c|c|c} \mathbf{0}_{4\times3} & 2\left[\mathbf{q}\right]_R^{\mathsf{T}} & \begin{bmatrix} 0 \\ {}^{\mathtt{W}}\mathbf{R}_{\mathtt{L}_0}{}^{\mathtt{L}_0}\mathbf{J}_{\omega,\mathtt{i}} \end{bmatrix} \end{array}\right] \in \mathbb{R}^{4\times(7+N_J)}. \tag{5.41} $$

5.3.2 Equations of Motion

Solving the Lagrangian equations is an algorithmically simple way of retrieving the dynamics equations. However, it comes with some requirements regarding the choice of coordinates. More precisely, the coordinates need to be independent and complete. It is clear that the chosen coordinates (5.37) are complete and can represent any configuration of the system. However, the requirement of independent coordinates is not met, e.g., fixing the coordinates q_x, q_y, q_z of $\mathbf{q}$ is also fully constraining q_w by the unity constraint on (3.4), such that $\|\mathbf{q}\| = 1$. In other words, the system has more coordinates than degrees of freedom. But, for the time being, we assume that all the coordinates are independent and then handle the resulting holonomic constraint forces later in Section 5.4.

In the Lagrangian framework, the system dynamics are obtained by solving the Euler-Lagrange equations [109]

$$ \frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{\bar{\mathbf{x}}}} - \frac{\partial \mathcal{L}}{\partial \bar{\mathbf{x}}} = \bar{\mathbf{f}}_x + \bar{\mathbf{f}}_c, \tag{5.42} $$

where $\mathcal{L} = \mathsf{E}_{\mathsf{kin}} - \mathsf{E}_{\mathsf{pot}}$ is the Lagrangian, $\bar{\mathbf{x}} \in \mathbb{R}^{N_x}$ the state space of the system, $\bar{\mathbf{f}}_x \in \mathbb{R}^{N_x}$ are the generalized forces resp. torques along the generalized coordinates, and $\bar{\mathbf{f}}_c \in \mathbb{R}^{N_x}$ mark the constraint forces, pulling the system towards the manifold defined by the holonomic constraints. Herein, the constraint forces arise by the unit quaternion coordinates not being independent.

In the case of mechanical systems, the solution of (5.42) takes particular form [92]:

$$ \mathbf{M}\left(\bar{\mathbf{x}}\right)\ddot{\bar{\mathbf{x}}} + \mathbf{C}\left(\bar{\mathbf{x}}, \dot{\bar{\mathbf{x}}}\right)\dot{\bar{\mathbf{x}}} + \bar{\mathbf{g}}\left(\bar{\mathbf{x}}\right) = \bar{\mathbf{f}}_x + \bar{\mathbf{f}}_c, \tag{5.43} $$

where $\mathbf{M} \in \mathbb{R}^{N_x \times N_x}$ is the mass matrix, $\mathbf{C} \in \mathbb{R}^{N_x \times N_x}$ the Coriolis matrix and $\bar{\mathbf{g}} \in \mathbb{R}^{N_x}$ the vector of gravitational terms. The total kinetic energy of the system is given by

$$\mathsf{E}_{\mathsf{kin}} = \frac{1}{2}\dot{\mathbf{x}}{}^{\mathtt{W}}\mathbf{J}^{\mathsf{T}}\mathbf{M}_L{}^{\mathtt{W}}\mathbf{J}\dot{\mathbf{x}} \tag{5.44}$$

with $\mathbf{M}_L$ being the generalized inertia matrix of the system represented by the block diagonal formed by the individual body masses m_i and the moments of inertia ${}^{\mathtt{W}}\mathbf{\Theta}_i$ in world frame [58, 110]. It is defined as

$$\begin{aligned}\mathbf{M}_L = \operatorname{blockdiag}\big(&\mathbf{1}_3 m_0, \ldots, \mathbf{1}_3 m_{N_B-1}, \\ &{}^{\mathtt{W}}\mathbf{\Theta}_0, \ldots, {}^{\mathtt{W}}\mathbf{\Theta}_{N_B-1}\big)\,,\end{aligned} \tag{5.45}$$

with ${}^{\mathtt{W}}\mathbf{\Theta}_i$ being linked to the classical inertia tensor ${}^{\mathtt{W}}\mathbf{\Phi}_i$ by

$${}^{\mathtt{W}}\mathbf{\Theta}_i = \begin{bmatrix} \nu & \bar{\mathbf{0}} \\ \bar{\mathbf{0}} & {}^{\mathtt{W}}\mathbf{\Phi}_i \end{bmatrix} \in \mathbb{R}^{4\times 4}. \tag{5.46}$$

Therein, ν is associated with the scaling DOF of a body that would arise without enforcing $\|\mathbf{q}\| = 1$. However, since $\mathbf{q}$ is assumed to be of unit length, the bodies of the system are rigid, and ν does not come into play. As such, ν can be defined as any positive number [110].

Solving (5.42) directly via calculating $\mathcal{L}$ becomes impractical resp. computationally expensive and thus time-consuming for larger systems. A much faster approach is the factorization into the individual system matrices (5.43) which has the additional benefit of not having to separate the individual components into their respective matrix. Once the expression for the system's mass matrix is obtained, the Coriolis matrix can be obtained from its Christoffel symbols. The gravitational terms are obtained from the potential energy, which is usually a relatively compact expression and thus inexpensive to compute.

By analyzing (5.43), (5.44), it can be seen that the only resulting term in $\ddot{\mathbf{x}}$, corresponding to the system's mass matrix $\mathbf{M}$ (symmetric, positive semi-definite [111]), is obtained from

$$\mathbf{M} = {}^{\mathtt{W}}\mathbf{J}^{\mathsf{T}}\mathbf{M}_L{}^{\mathtt{W}}\mathbf{J}, \tag{5.47}$$

The system's Coriolis matrix $\mathbf{C} \in \mathbb{R}^{N_x \times N_x}$ containing the centrifugal and centripetal terms is obtained by calculating the Christoffel symbols first kind from the mass matrix (5.47), [92]. Its individual components are given by

$$\mathbf{C}_{ij} = \frac{1}{2} \sum_{k=1}^{N_x} \left(\frac{\partial \mathbf{M}_{ij}}{\partial \bar{\mathbf{x}}_k} + \frac{\partial \mathbf{M}_{ik}}{\partial \bar{\mathbf{x}}_j} - \frac{\partial \mathbf{M}_{jk}}{\partial \bar{\mathbf{x}}_i} \right) \dot{\bar{\mathbf{x}}}_k. \tag{5.48}$$

The definition of the Coriolis matrix is not unique and different choices are possible (e.g., [112]). The choice herein has the property of $\dot{\mathbf{M}} - 2\mathbf{C}$ being skew-symmetric, a useful property for controller design [92].

Lastly, the system's gravity vector $\bar{\mathbf{g}}$ is obtained from the gradient of the potential energy

$$\mathsf{E}_{\mathsf{pot}} \left(\bar{\mathbf{x}} \right) = \sum_{i=0}^{N_B - 1} m_i \bar{\mathbf{g}}_0 {}^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{B_i}}^{\mathtt{W}}, \tag{5.49}$$

with $\bar{\mathbf{g}}_0$ being the gravitational acceleration. The vector $\bar{\mathbf{g}}$ contains the gravitational terms and thus accounts for the shifting center of gravity due to the motion of the manipulator. It is obtained from the gradient of the potential energy by calculating

$$\bar{\mathbf{g}} = \nabla \mathsf{E}_{\mathsf{pot}} = \frac{\partial \mathsf{E}_{\mathsf{pot}}}{\partial \bar{\mathbf{x}}}. \tag{5.50}$$

An alternative to those equations is presented in [101], where the system's mass matrix $\mathbf{M}$ and lumped Coriolis vector $\bar{\mathbf{h}}$ are obtained from the system's total kinetic energy by calculating

$$\mathbf{M} = \frac{\partial}{\partial \dot{\bar{\mathbf{x}}}} \left(\frac{\partial \mathsf{E}_{\mathsf{kin}}}{\partial \dot{\bar{\mathbf{x}}}} \right), \tag{5.51}$$

$$\bar{\mathbf{h}} = \frac{\partial}{\partial \bar{\mathbf{x}}} \left(\frac{\partial \mathsf{E}_{\mathsf{kin}}}{\partial \dot{\bar{\mathbf{x}}}} \right) \dot{\bar{\mathbf{x}}} - \frac{\partial \mathsf{E}_{\mathsf{kin}}}{\partial \bar{\mathbf{x}}}. \tag{5.52}$$

The vector $\bar{\mathbf{h}}$ is equal to $\mathbf{C}\dot{\bar{\mathbf{x}}}$.

5.3.3 Force Mapping

The term $\bar{\mathbf{f}}_x$ in (5.43) is given in terms of forces along the generalized coordinates. In many situations, it is very convenient to have a mapping from local body forces to generalized forces. This mapping also greatly facilitates the application of the forces and torques originating from the propulsion system of the AM.

By the choice of the generalized coordinates, the torques applied to the AM have to be specified along the four coordinates of the quaternion. The principle of virtual work is used to establish a relationship between the quaternion torques and the Cartesian torques along the body-fixed axis.

The principle of virtual work states that the work δW is equal to the force $\bar{\mathbf{f}}$ acting on a body along its virtual displacement $\delta\bar{\mathbf{s}}$:

$$\delta W = \bar{\mathbf{f}} \cdot \delta\bar{\mathbf{s}}. \tag{5.53}$$

In a completely analogous manner, the work performed by the torque ${}^{\mathtt{L_0}}\bar{\mathbf{f}}_{\mathtt{q}}$ acting on the rotation $\delta\bar{\varphi} = \bar{\mathbf{n}}\delta\varphi$ is defined as

$$\delta W_q = {}^{\mathtt{L_0}}\bar{\mathbf{f}}_{\mathtt{q}} \cdot \bar{\mathbf{n}}\delta\varphi. \tag{5.54}$$

Following the approach in [57] stating that for small changes in rotations (i.e., $\|\mathbf{q}_\delta\| = 1$), the following approximation holds:

$$\mathbf{q} + \delta\mathbf{q} \approx \mathbf{q} \otimes \mathbf{q}_\delta, \tag{5.55}$$

where $\delta\mathbf{q}$ is a small variation and $\mathbf{q}_\delta$ is a small change of rotation from $\mathbf{q}$. Left multiplying both sides of that relation with $\mathbf{q}^*$ yields:

$$(1, \bar{\mathbf{0}}) + \mathbf{q}^* \otimes \delta\mathbf{q} \approx \mathbf{q}_\delta. \tag{5.56}$$

Using the axis angle relation (3.8), $\mathbf{q}_\delta$, under consideration of small angles, can be approximated as

$$\mathbf{q}_\delta \approx \left(1, \frac{1}{2}\bar{\mathbf{n}}\delta\varphi\right). \tag{5.57}$$

Thus, (5.57) in (5.56) results in

$$2\left(\mathbf{q}^* \otimes \delta\mathbf{q}\right) \approx \left(0, \bar{\mathbf{n}}\delta\varphi\right). \tag{5.58}$$

Writing (5.54) as the dot product of two pure quaternions

$$\delta W_q \approx \left(0, {}^{\mathtt{Lo}}\bar{\mathbf{f}}_{\mathtt{q}}\right) \cdot \left(0, \bar{\mathbf{n}}\delta\varphi\right), \tag{5.59}$$

and by inserting (5.58) the relation

$$\delta W_q \approx \left(0, {}^{\mathtt{Lo}}\bar{\mathbf{f}}_{\mathtt{q}}\right) \cdot 2\left(\mathbf{q}^* \otimes \delta\mathbf{q}\right) \tag{5.60}$$

is obtained. Using (5.7) yields

$$\delta W_q \approx 2{}^{\mathtt{Lo}}\bar{\mathbf{f}}_{\mathtt{q}} \cdot \mathbf{G}\delta\mathbf{q}. \tag{5.61}$$

With the help of the dot product property $\bar{\mathbf{x}} \cdot \mathbf{A}\bar{\mathbf{y}} = \mathbf{A}^\intercal\bar{\mathbf{x}} \cdot \bar{\mathbf{y}}$, (5.61) becomes

$$\delta W_q \approx 2\mathbf{G}^\intercal{}^{\mathtt{Lo}}\bar{\mathbf{f}}_{\mathtt{q}} \cdot \delta\mathbf{q}, \tag{5.62}$$

wherein $2\mathbf{G}^\intercal$ can be identified as the mapping between Cartesian torques (in body frame) and the generalized quaternion torques. The forces along the body-fixed axis can be mapped to the generalized forces with

$${}^{\mathtt{W}}\bar{\mathbf{f}}_{\mathtt{xyz}} = {}^{\mathtt{W}}\mathbf{R}_{\mathtt{L_0}}{}^{\mathtt{Lo}}\bar{\mathbf{f}}_{\mathtt{xyz}}. \tag{5.63}$$

The mapping of joint torques to generalized joint forces is trivial:

$$^{\mathtt{W}}\bar{\mathbf{f}}_{\mathtt{J}} = \mathbf{1}_{N_J}{}^{\mathtt{L_0}}\bar{\mathbf{f}}_{\mathtt{J}}. \tag{5.64}$$

The results (5.62), (5.63) and (5.64) are summarized in the following block diagonal force mapping matrix $\mathbf{M}_F$, mapping from body forces to generalized forces $\bar{\mathbf{f}}_x$:

$$\mathbf{M}_F = \text{blockdiag}\left({}^{\mathtt{W}}\mathbf{R}_{\mathtt{L_0}}, 2\mathbf{G}^{\mathsf{T}}, \mathbf{1}_{N_J}\right), \tag{5.65}$$

such that the generalized forces in (5.43) can be written as

$$\bar{\mathbf{f}}_x = \mathbf{M}_F\bar{\mathbf{f}}_B, \tag{5.66}$$

with ${}^{\mathtt{L_0}}\bar{\mathbf{f}}_{\mathtt{B}}$ being the body forces composed by the forces ${}^{\mathtt{L_0}}\bar{\mathbf{f}}_{\mathtt{xyz}}$ and torques ${}^{\mathtt{L_0}}\bar{\mathbf{f}}_{\mathtt{q}}$ in resp. about the local x, y and z body axis and the joint torques ${}^{\mathtt{L_0}}\bar{\mathbf{f}}_{\mathtt{J}}$. Those forces typically originate from the propulsion system but can also include aerodynamic effects if so desired. The body force vector is thus defined by

$${}^{\mathtt{L_0}}\bar{\mathbf{f}}_{\mathtt{B}} = \begin{bmatrix} {}^{\mathtt{L_0}}\bar{\mathbf{f}}_{\mathtt{xyz}} & {}^{\mathtt{L_0}}\bar{\mathbf{f}}_{\mathtt{q}} & {}^{\mathtt{L_0}}\bar{\mathbf{f}}_{\mathtt{J}} \end{bmatrix}. \tag{5.67}$$

5.3.4 Propulsion Forces

The body forces (5.67) are defined as a function of the thrust of the AM's propulsion system. Given a UAV with N_M propulsion devices, then the forces from the propulsion system acting on the body of the AM are obtained from the following relations

$$\begin{aligned} {}^{\mathtt{L_0}}\bar{\mathbf{f}}_{\mathtt{xyz}} &= \sum_i^{N_M} {}^{\mathtt{L_0}}\bar{\mathbf{f}}_{thrust,i} && \text{(5.68a)} \\ {}^{\mathtt{L_0}}\bar{\mathbf{f}}_{\mathtt{q}} &= \sum_i^{N_M} {}^{\mathtt{L_0}}\bar{\mathbf{f}}_{rot,i} \\ &= \sum_i^{N_M} \left({}^{\mathtt{L_0}}\bar{\mathbf{r}}_{\mathtt{M,i}}^{\mathtt{L_0}} \times {}^{\mathtt{L_0}}\bar{\mathbf{f}}_{thrust,i} + {}^{\mathtt{L_0}}\bar{\mathbf{f}}_{drag,i}\right). && \text{(5.68b)} \end{aligned}$$

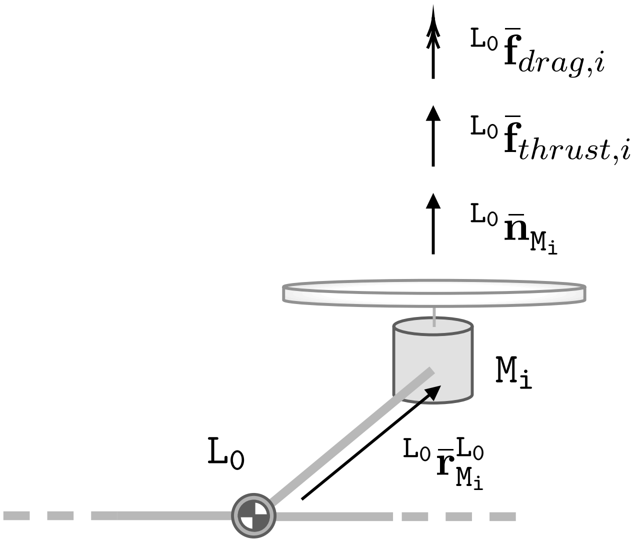

$^{\mathtt{L_0}}\bar{\mathbf{f}}_{drag,i}$
$^{\mathtt{L_0}}\bar{\mathbf{f}}_{thrust,i}$
$^{\mathtt{L_0}}\bar{\mathbf{n}}_{\mathtt{M_i}}$
$\mathtt{M_i}$
$\mathtt{L_0}$
$^{\mathtt{L_0}}\bar{\mathbf{r}}^{\mathtt{L_0}}_{\mathtt{M_i}}$

Figure 5.4: A segment of the propulsion system. The thrust force along the motor-spin normal and the associated drag of the propeller generate a net force acting on $\mathtt{L_0}$.

Therein $\bar{\mathbf{f}}_{thrust,i}$ designates the thrust force vector and $\bar{\mathbf{f}}_{drag,i}$ the drag vector (reaction torque). The vector $^{\mathtt{L_0}}\bar{\mathbf{r}}^{\mathtt{L_0}}_{\mathtt{M,i}}$ designates the lever formed between the UAV's body and the motor frame (see Fig. 5.4). The mapping matrix $\mathbf{M}_{mot}$, also commonly referred to as efficiency or allocation matrix (used by the mixer in many controllers), contains the contribution of each motor to each force resp. torque along each body axis. It is constant (unless the motors themselves can tilt) and defined by

$$\begin{aligned}\mathbf{M}_{mot} = \text{blockdiag}\Big(&\left[{}^{\mathtt{L_0}}\bar{\mathbf{f}}_{thrust,1}, \ldots, {}^{\mathtt{L_0}}\bar{\mathbf{f}}_{thrust,N_M}\right], \\ &\left[{}^{\mathtt{L_0}}\bar{\mathbf{f}}_{rot,1}, \ldots, {}^{\mathtt{L_0}}\bar{\mathbf{f}}_{rot,N_M}\right], \mathbf{1}_N\Big)\end{aligned} \tag{5.69}$$

and can be used to map body forces to motor forces.

Both quantities are a function of the angular velocity $\omega_{\mathtt{M_i}}$ of the propeller [113] as in (see Fig. 5.1)

$${}^{\mathtt{L_0}}\bar{\mathbf{f}}_{thrust,i} = {}^{\mathtt{L_0}}\bar{\mathbf{n}}_{\mathtt{M_i}} k_t \omega^2_{\mathtt{M_i}}, \tag{5.70a}$$

$${}^{\mathtt{L_0}}\bar{\mathbf{f}}_{drag,i} = {}^{\mathtt{L_0}}\bar{\mathbf{n}}_{\mathtt{M_i}} k_p s_{\mathtt{M_i}} \omega^2_{\mathtt{M_i}}, \tag{5.70b}$$

where ${}^{\mathtt{L0}}\bar{\mathbf{n}}_{\mathtt{M,i}}$ is the motor's normalized axis of rotation. The constants k_t and k_p for thrust and drag can be identified experimentally. Furthermore, since $\omega_{\mathtt{M_i}}$ is rarely known and only very few ESCs provide telemetry resp. accept inputs as RPM references, (5.70a), (5.70b), can also be written as a function of the flight controller's output PWM signal $u_{M,i} = [0, 1]$, which yields the roughly linear relationship [113], [114]:

$$
{}^{\mathtt{L0}}\bar{\mathbf{f}}_{thrust,i} \approx {}^{\mathtt{L0}}\bar{\mathbf{n}}_{\mathtt{M_i}} \hat{k}_t u_{\mathtt{M_i}}, \tag{5.71a}
$$

$$
{}^{\mathtt{L0}}\bar{\mathbf{f}}_{drag,i} \approx {}^{\mathtt{L0}}\bar{\mathbf{n}}_{\mathtt{M_i}} \hat{k}_p s_{\mathtt{M_i}} u_{\mathtt{M_i}}, \tag{5.71b}
$$

where $s_{M,i} = \{1, -1\}$, for CW resp. CCW, designates the spin direction of the motor. Notice that for tilting propeller configurations, the normal vector ${}^{\mathtt{L0}}\bar{\mathbf{n}}_{\mathtt{M_i}}$ would be a function of the tilt angles commanded to the tilting mechanism.

A real motor cannot reach its commanded velocity instantaneously; therefore, it is common practice to model the motor dynamics as a first-order system [114]:

$$
G(s) = \frac{K}{1 + Ts}, \tag{5.72}
$$

where K is the maximum RPM and T is the time constant of the motor-propeller system.

Robotic joints are generally more complex and have a non-negligible amount of friction and damping due to their internal mechanics. Their modeling highly depends on their internals and thus has to be seen on a case-by-case basis. For the sake of simplicity, the model in (5.72) is also used for the robotic joints throughout this work.

5.4 Simulation and Constraints

The numeric simulation of the AM is a very helpful and sometimes even critical step before deploying an AM in practice. Furthermore, there are applications in deep reinforcement learning where the model needs to be simulated such that the controller can be learned from the model dynamics [91]. The equations for numerical simulation that respect the holonomic constraints are derived herein.

The system dynamics equation (5.43) can be written in form of the differential algebraic equation (DAE) [96]

$$\begin{cases} \mathbf{M}\ddot{\bar{\mathbf{x}}} + \mathbf{C}\dot{\bar{\mathbf{x}}} + \bar{\mathbf{g}} - \bar{\mathbf{f}}_x - \lambda \mathbf{J}_{\bar{\phi}} = \bar{\mathbf{0}}, & (5.73) \\ \bar{\phi} = \bar{\mathbf{0}}, & (5.74) \end{cases}$$

where $\bar{\phi}$ are the holonomic constraints imposed on the system, $\mathbf{J}_{\bar{\phi}}$ designates the Jacobian matrix of $\bar{\phi}$ and λ is a Lagrange multiplier. The constraint vector contains (at least) the unity constraint originating from the unit quaternion:

$$\bar{\phi}(\bar{\mathbf{x}}) = [\|\mathbf{q}\| - 1]. \tag{5.75}$$

The system equation can be used for numerical simulation by solving (5.43) for $\ddot{\bar{\mathbf{x}}}$ and implementing it as first-order ODEs:

$$\dot{\bar{\mathbf{z}}} = \begin{cases} \dot{\bar{\mathbf{x}}} = \bar{\mathbf{v}}, & (5.76) \\ \dot{\bar{\mathbf{v}}} = \bar{\mathbf{a}}, & (5.77) \end{cases}$$

wherein $\bar{\mathbf{a}} = \ddot{\bar{\mathbf{x}}}$ marks the unconstrained acceleration:

$$\bar{\mathbf{a}} = \mathbf{M}^{-1}\left[\bar{\mathbf{f}}_x - \mathbf{C}\dot{\bar{\mathbf{x}}} - \bar{\mathbf{g}}\right]. \tag{5.78}$$

Naturally, such a system would not be constrained to the manifold defined by the constraint $\bar{\phi} = 0$, violating the unit length assumption of the quaternion, thus degrading the physical fidelity of the system up to a point where it is completely disjoint from reality. Therefore, it is mandatory to constrain the system such that the unit length assumption holds. Generally, this can be seen as a holonomic constraint problem, which is typically addressed by Lagrangian multipliers within the Lagrangian framework.

Herein, we opted for a general-purpose solution to this holonomic constraints problem as discussed in [109, 115] and [116], in the spirit that it provides additional value, e.g., in a reinforcement learning context, where it might be useful to temporarily stiffen the joints to speed up the training process. The formulation is, however, fairly heavy. The authors of

[96, 117] specifically addressed the problem of the quaternion unit constraint and provided a numerically faster method. However, the general idea behind those methods is the same. It consists of performing small corrective actions orthogonal to the manifolds defined by the constraints for the system's velocities and positions. The authors of [58] showed how to calculate the Lagrangian multiplier directly. However, this requires splitting the system into translational and rotational parts, which is unfeasible here due to the coupled nature of the system.

According to [116], the holonomic constraint is enforced onto the system by constraining the accelerations and velocities, which are obtained by the sum of the unconstrained acceleration $\bar{\mathbf{a}}$ resp. velocity $\bar{\mathbf{v}}$ and a corrective term. The system respecting the holonomic constraints is thus given by

$$\dot{\bar{\mathbf{z}}}_{\bar{\phi}} = \begin{cases} \dot{\bar{\mathbf{x}}} = \bar{\mathbf{v}} + \mathbf{M}^{-1/2}\mathbf{B}^{+}\left(\bar{\mathbf{b}}_q - \mathbf{A}\dot{\bar{\mathbf{v}}} - \bar{\phi}/\mathbf{dt}\right), & (5.79a) \\ \dot{\bar{\mathbf{v}}} = \bar{\mathbf{a}} + \mathbf{M}^{-1/2}\mathbf{B}^{+}\left(\bar{\mathbf{b}}_v - \mathbf{A}\bar{\mathbf{a}} - \dot{\bar{\phi}}/\mathbf{dt}\right), & (5.79b) \end{cases}$$

where $(.)^{+}$ designates the Moore-Penrose 'pseudo' inverse. The remaining terms are computed as

$$\mathbf{A} = \frac{\partial \bar{\phi}}{\partial \bar{\mathbf{x}}} \in \mathbb{R}^{N_\phi \times N_x}, \tag{5.80a}$$

$$\mathbf{B} = \mathbf{A}\mathbf{M}^{-1/2} \in \mathbb{R}^{N_\phi \times N_x}, \tag{5.80b}$$

$$\bar{\mathbf{b}}_q = -\frac{d}{dt}\bar{\phi} \in \mathbb{R}^{N_\phi}, \tag{5.80c}$$

$$\bar{\mathbf{b}}_v = -\dot{\bar{\mathbf{x}}}^{\intercal}\left(\frac{\partial^2 \bar{\phi}}{\partial \bar{\mathbf{x}}^2}\right)\dot{\bar{\mathbf{x}}} - 2\frac{\partial^2 \bar{\phi}}{\partial t \partial \bar{\mathbf{x}}}\dot{\bar{\mathbf{x}}} - \frac{d^2 \bar{\phi}}{dt^2} \in \mathbb{R}^{N_\phi}, \tag{5.80d}$$

where N_x is the number of states of the system and N_ϕ the number of constraints imposed onto the system.

Integrating this system using common numerical integration methods (e.g., RK4) results in a small error that occurs after each integration step, caused by the fact that S^3 is not closed for addition $\left(\mathbf{q} + \mathbf{q}_{\boldsymbol{\delta}} \notin S^3\right)$. The authors of [118] have shown how this error accumulates and alters the dynamics of the system. In practice, however, the error is relatively small under the condition that the time steps are sufficiently small, in which case (5.55) holds. Furthermore, it is not required to re-normalize the quaternion after each iteration since the constraint drives the unity norm error to zero.

5.5 Validation

In the following section, the physical fidelity of the constrained system model (5.79) is validated.

5.5.1 Methodology

Property	Value
UAV Configuration	Quadcopter
#joints & links	1
Joint Type	Revolute (y-axis)
Wheelbase	1.51 m
Mass (base)	6 kg
Inertia (base)	$(0.48, 0.48, 0.95)\,\mathrm{kg\,m^2}$
Mass (link 1)	1 kg
Inertia (link 1)	$(0.595, 3.824, 3.7)\,10^{-3}\,\mathrm{kg\,m^2}$
CM (link 1)	$(0.5, 0, 0)\,\mathrm{m}$
k_t	$2.165 \times 10^{-6}\,\mathrm{N\,min^{-2}}$
k_p	$5.865 \times 10^{-8}\,\mathrm{N\,m\,min^{-2}}$
P (prop)	$4500\,\mathrm{min^{-1}}$
P (joint)	16 N m
T (prop & joint)	0.2 s

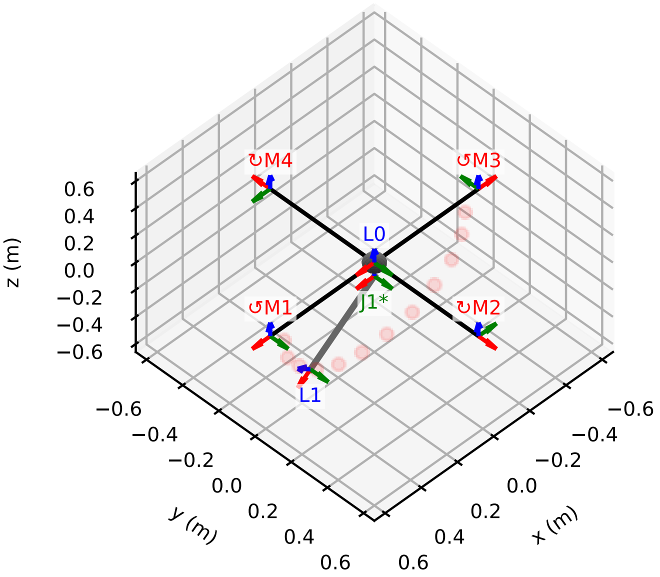


Figure 5.5: Physical properties of the Lagrange-model and the corresponding configuration of the quadcopter with a 1 DOF manipulator (rotating about its local y-axis) used for the validation test. The range of motion of the manipulator is indicated by the red dots. The present manipulator configuration corresponds to $\bar{\theta} = [45°]$

Correctness of the model is shown by comparing the evolution of a dynamics simulation of an AM with a 1-DOF manipulator (Fig. 5.5) using the constrained first order system (5.79) and comparing them with the simulation as performed by the *Bullet* physics engine [119] using the same URDF model in both cases. The *Bullet* simulation is therefore the reference model, called the *Bullet-model*. For clarity, the model corresponding to (5.79) is referred to as the *Lagrange-model*.

The simulation in both cases is performed at a fixed time step of 240 Hz. The simulation of the Lagrange-model uses the Euler forward (first-order) integration scheme, *Bullet* physics uses the semi-implicit Euler method. As far as possible, equal conditions have been established for both simulations (e.g., the default linear and angular damping in *Bullet* has been disabled for this test).

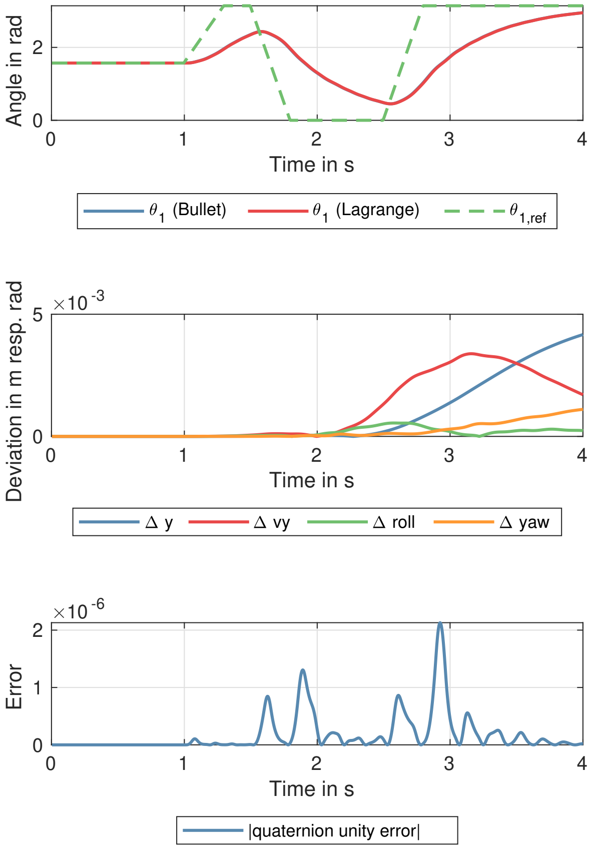


Figure 5.6: Joint positions (top) and the error on all remaining axis (center) of the AM. The absolute quaternion unity error $abs(\|q\| - 1)$ (bottom) is driven to zero by the holonomic constraint.

A short dynamics simulation of the modeled AM is carried out, during which positions, velocities, and propulsion forces are recorded. Given the inherently unstable nature of the AM, it is stabilized via a stabilizing (sliding mode) controller similar to [43] around the equilibrium point. During the simulation, the first (and only) joint θ_1 is driven towards the commanded reference position, following the pattern shown in Fig. 5.6 (top). The movement of that joint is thus acting as a disturbance onto the closed-loop system and stresses the various terms in (5.43). Caused by the accumulation of small errors at each simulation step, the two systems are prone to deviate from each other eventually. Furthermore, the same control input no longer stabilizes both systems. This limits the time interval over which the dynamics of both systems are comparable. Here, this interval was identified to be around 4 s.

The recorded forces applied to the Lagrangian-model via the propulsion devices are then copied to the Bullet-model, which is thus pseudo-open-loop. If both models are sufficiently close, the exact same forces should stabilize both models (within a reasonable time span). The joint torques, on the other hand, are not transferred due to numerical stability concerns, but since the joint follows the same trajectory, the generated disturbance is the same in both cases. Therefore, the Bullet-model tracks the joint velocity and position of the Lagrange-model via *Bullet's* built-in *joint position control* algorithm. The process is depicted in Fig. 5.7.

The hypothesis is thus if both systems have the same intrinsic dynamics, the time response of both systems is going to be the same, resp. very close. Nonetheless, the two systems eventually diverge over time due to the accumulation of minor errors, e.g., caused by numerical inaccuracies, single vs. double precision floating point format, fundamental architectural differences and different integration schemes. This is the reason for keeping the observation interval relatively short.

Lastly, by performing a 'backflip' maneuver during which the simulated AM momentarily points straight up resp. down, it can be verified that the proposed model remains consistent where the model using Euler angles [43] fails due to gimbal lock [120]. To that end, the aforementioned 1-link AM is commanded to flip about its local y-axis, thus pitching the platform by more than 90°.

5.5.2 Results

By following the commanded pattern shown in Fig. 5.6 (top), the first joint creates a disturbance in the system. The manipulator moves in the xz-plane (Fig. 5.5), with the reaction

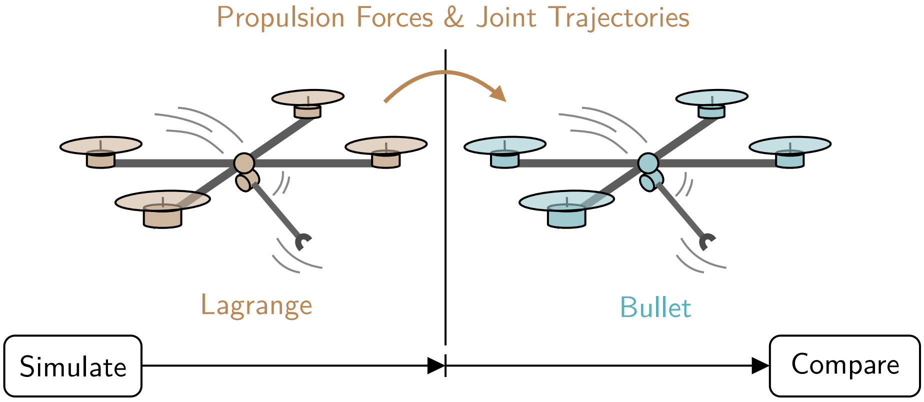
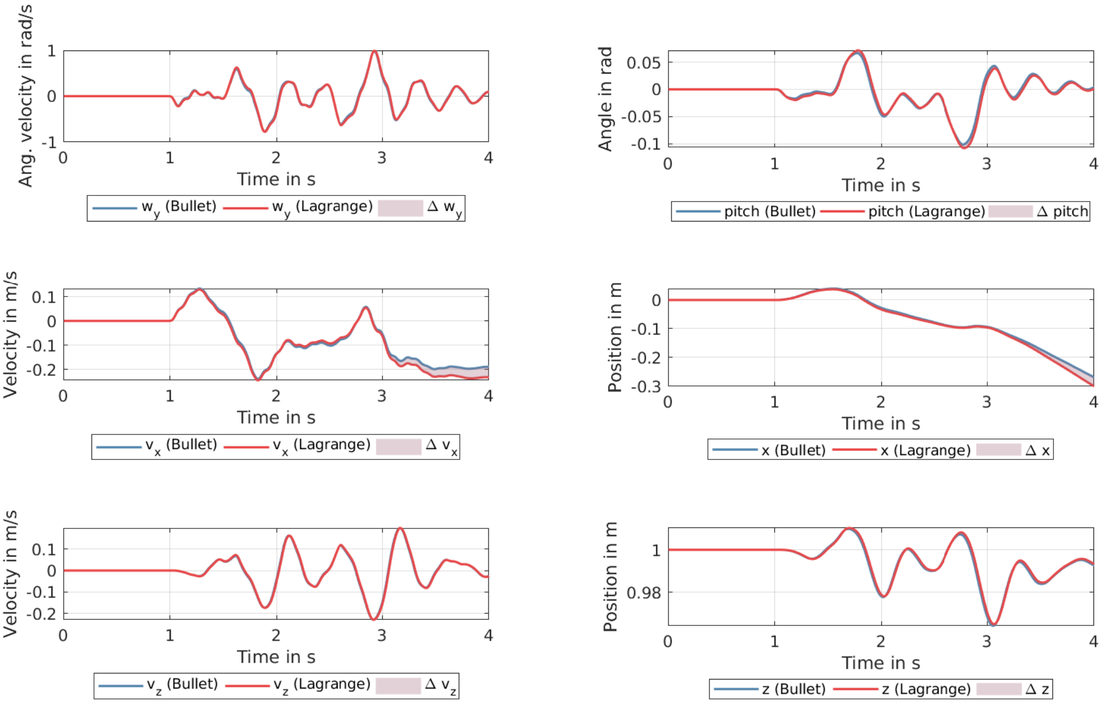


Figure 5.7: The validation scenario. The model simulated using the model developed inhere (left), transfers the propulsion forces and joint trajectories to the (physically) identical model simulated with *Bullet* (right). The two models are identical if they evolve dynamically close over a time t.

torque acting on the y-axis (pitch) and the Coriolis forces pushing the vehicle in x resp. z-direction. The resulting motion is shown in Fig. 5.8 and Fig. 5.6.

The resulting velocities along the main axis of the disturbance are shown in Fig. 5.8a. The reaction torque creates a net torque around the y-axis, causing the system to accelerate, and turn about the y-axis. The same applies to the velocities in X and Z caused by the Coriolis forces resulting from the circular motion of the moving mass (the link here). The corresponding positions are shown in Fig. 5.8b. Noticeably, the evolution of both models is extremely close; however, as can be seen in Fig. 5.8a (center), minor errors do sum up. Consequently, both systems deviate from each other eventually and the control input can no longer stabilize both systems. Therefore, the observation period is only 4 s and a comparison thus only makes sense over that short interval.

The yaw and roll position discrepancies are shown in Fig. 5.6. Since the manipulator passes, during its movement, straight through the center of mass (in the xz-plane), there should be no rotation about the roll axis, which is the case here as it measures close to zero. The same reasoning applies to the yaw axis, except that also the propeller drag comes into play here. In the y-direction, the AM exhibits a net-zero movement as the manipulator moves in the xz-plane.



(a) Angular velocity about the y-axis (top), velocity in y-direction (center), velocity in z-direction (bottom), for both the Bullet-model and the Lagrange-model as well as their mutual deviation.

(b) Angular position about the pitch axis (top), position in x-direction (center), position in z-direction (bottom), for both the Bullet-model and the Lagrange-model as well as their mutual deviation.

Figure 5.8: Validation results

The absolute quaternion unity error is shown in Fig. 5.6 (bottom). Since this is the fundamental assumption made at the start, it must deviate from unity as little as possible. The maximum error measured here was only 2.9×10^{-6} indicating that the holonomic constraint solver pushes the quaternion norm to stay close to unity. It can also be seen that re-normalizing the quaternion after each iteration is unnecessary.

It can thus be concluded that the developed model is true to the reference model as simulated by the *Bullet* real-time physics engine (Fig. 5.8).

Lastly, a backflip maneuver pitching the AM by more than 90° is shown in Fig. 5.9. The results show that the Euler-model fails close to the south pole (90° mark) as it enters

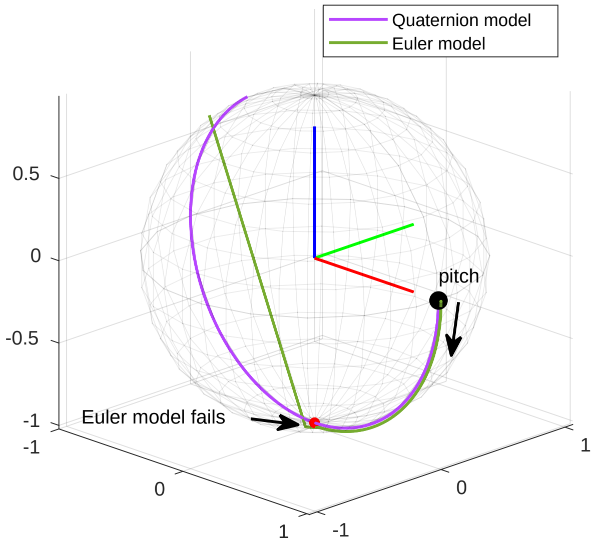


Figure 5.9: Quaternion parameterization avoids the singularities inherent to Euler parameterization. Here, the UAV is commanded to pitch (backflip), which drives the Euler-model into gimbal lock as it approaches a pitch angle of 90° close to the south pole, ultimately causing the motion to become unpredictable. The two trajectories represent the vehicle's x-axis plotted on a unit-sphere. A gimbal lock condition causes a sudden jump on the Euler trajectory (green).

a gimbal lock condition, where the movement becomes unpredictable as the transformation matrix tying Euler rates to angular velocities becomes rank deficient. Consequently, the system states become 'NaN' - terminating the simulation. Our quaternion model has no such limitations and remains consistent.

5.6 Control

This section briefly demonstrates an application of the developed model in a control context applied to an AM consisting of a quadcopter carrying a 2-DOF serial link manipulator. Furthermore, investigations are performed to analyze the real-time applicability of the control scheme with increasingly complex AM systems.

5.6.1 Computed Torque Control

The controller is largely based on the computed torque controller presented in [92] and depicted in Fig. 5.10. The controller makes use of the model for feedback linearization and is, at its core, a PD controller with bias terms (known disturbances). This very simple structure was deliberately chosen for the subsequent tests as it does not hide model imperfections, contrary to the various state-of-the-art robust control schemes, which are largely preferable for real-world applications. Nonetheless, they also make heavy use of a dynamics model such as presented in this work.

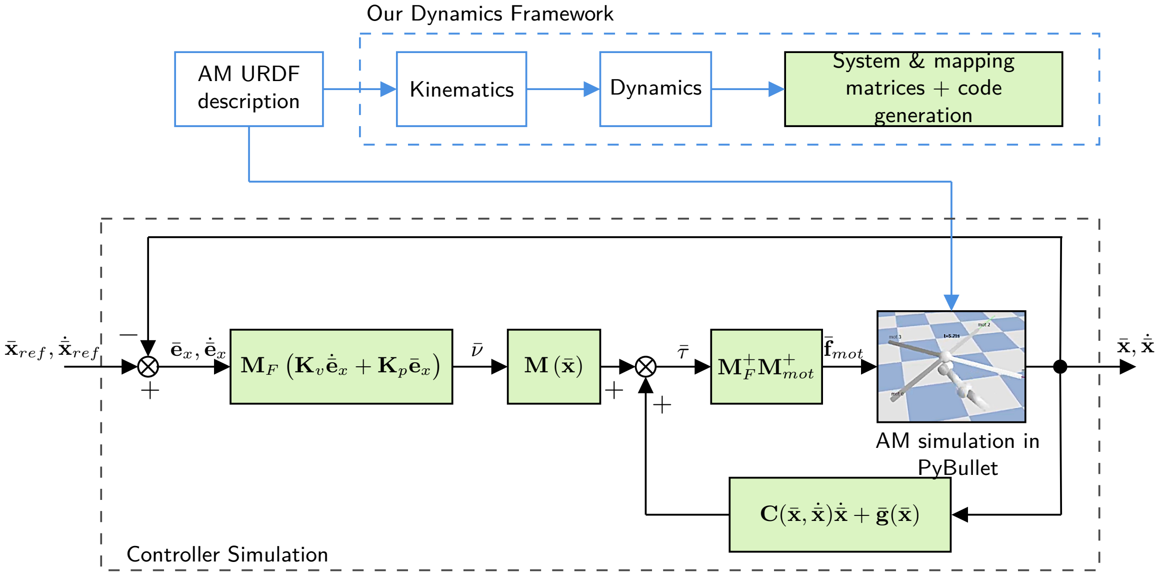

Our Dynamics Framework
AM URDF description
Kinematics
Dynamics
System & mapping matrices + code generation
$\bar{\mathbf{x}}_{ref}, \dot{\bar{\mathbf{x}}}_{ref}$
−
+
$\bar{\mathbf{e}}_x, \dot{\bar{\mathbf{e}}}_x$
$\mathbf{M}_F \left(\mathbf{K}_v \dot{\bar{\mathbf{e}}}_x + \mathbf{K}_p \bar{\mathbf{e}}_x\right)$
$\bar{\nu}$
$\mathbf{M}\left(\bar{\mathbf{x}}\right)$
+
$\bar{\tau}$
+
$\mathbf{M}_F^{+}\mathbf{M}_{mot}^{+}$
$\bar{\mathbf{f}}_{mot}$
AM simulation in PyBullet
$\bar{\mathbf{x}}, \dot{\bar{\mathbf{x}}}$
$\mathbf{C}(\bar{\mathbf{x}}, \dot{\bar{\mathbf{x}}})\dot{\bar{\mathbf{x}}} + \bar{\mathbf{g}}(\bar{\mathbf{x}})$
Controller Simulation

Figure 5.10: Block diagram of the computed torque controller in the context of our dynamics framework. The controller's output, mapped to motor forces, is fed into the PyBullet simulation where it is applied to the simulated model of the AM in terms of local forces $\bar{\mathbf{f}}_{mot}$ at their corresponding locations. The positions and velocities of the simulated model are then fed back into the controller. The system matrices of the AM are generated by our dynamics framework based on a URDF description which also serves as the blueprint for the model inside PyBullet.

Given the non-linear AM system

$$\mathbf{M}\ddot{\bar{\mathbf{x}}} + \mathbf{C}\dot{\bar{\mathbf{x}}} + \bar{\mathbf{g}} = \bar{\tau}, \tag{5.81}$$

by deliberately choosing the control input

$$\bar{\tau} = \mathbf{M}\bar{\nu} + \mathbf{C}\dot{\bar{\mathbf{x}}} + \bar{\mathbf{g}} \tag{5.82}$$

that cancels the nonlinearities via the bias term $\mathbf{C}\dot{\bar{\mathbf{x}}} + \bar{\mathbf{g}}$, the following system is obtained by inserting (5.82) in (5.81):

$$\ddot{\bar{\mathbf{x}}} = \bar{\nu}. \tag{5.83}$$

Using a PD-feedback law, the virtual control input $\bar{\nu}$ can be defined by

$$\bar{\nu} = \mathbf{M}_F\left(\ddot{\bar{\mathbf{x}}}_d + \mathbf{K}_v\dot{\bar{\mathbf{e}}}_x + \mathbf{K}_p\bar{\mathbf{e}}_x\right), \tag{5.84}$$

which, by inserting (5.84) in (5.83) yields the linear error dynamics

$$\mathbf{M}_F\left(\ddot{\bar{\mathbf{e}}}_x + \mathbf{K}_v\dot{\bar{\mathbf{e}}}_x + \mathbf{K}_p\bar{\mathbf{e}}_x\right) = \bar{\mathbf{0}}, \tag{5.85}$$

which is stable and guaranteed to converge by the right choice of the gain matrices $\mathbf{K}_v$ and $\mathbf{K}_p$ according to linear control theory. The mapping matrix $\mathbf{M}_F$ (5.65) maps the pseudo-forces of the controller into generalized coordinates.

The positional error vector $\bar{\mathbf{e}}$ is defined as

$$\bar{\mathbf{e}} = \begin{bmatrix} {}^{\mathtt{W}}\bar{\mathbf{p}}^{\mathtt{W}}_{\mathtt{L_0},ref} - {}^{\mathtt{W}}\bar{\mathbf{p}}^{\mathtt{W}}_{\mathtt{L_0}} \\ \bar{\mathbf{q}}_{v,err} \\ \bar{\theta}_{ref} - \bar{\theta} \end{bmatrix}, \tag{5.86}$$

where $\bar{\mathbf{q}}_{v,err}$ is the vector part of the error quaternion defined as $\mathbf{q}_{err} = \mathbf{q}^* \otimes \mathbf{q}_{ref}$.

The velocity error is given by

$$\dot{\bar{\mathbf{e}}} = \begin{bmatrix} {}^{\mathtt{W}}\dot{\bar{\mathbf{p}}}^{\mathtt{W}}_{\mathtt{L_0},ref} - {}^{\mathtt{W}}\dot{\bar{\mathbf{p}}}^{\mathtt{W}}_{\mathtt{L_0}} \\ {}^{\mathtt{B}}\bar{\boldsymbol{\omega}}^{\mathtt{B}}_{\mathtt{B},ref} - {}^{\mathtt{B}}\bar{\boldsymbol{\omega}}^{\mathtt{B}}_{\mathtt{B}} \\ \dot{\bar{\theta}}_{ref} - \dot{\bar{\theta}} \end{bmatrix}. \tag{5.87}$$

The acceleration error term $\ddot{\mathbf{e}}$ is not used as $\ddot{\mathbf{x}}$ is difficult to obtain in practice (in good quality), thus $\ddot{\mathbf{e}} = \bar{\mathbf{0}}$.

Lastly, the body forces to be applied by the controller are obtained by calculating

$$\bar{\mathbf{f}}_b = \mathbf{M}_F^+ \bar{\tau}, \tag{5.88}$$

which can easily be mapped to motor forces using the mapping matrix (5.69):

$$\bar{\mathbf{f}}_{mot} = \mathbf{M}_{mot}^+ \bar{\mathbf{f}}_b. \tag{5.89}$$

The motor forces can then further be mapped to RPM via relation (5.70).

It is clear that, on a real AM, the system matrices are often based on estimations resp. approximations and do thus not precisely cancel the system's nonlinearities, which degrades the controller's performance. A common approach is thus to rely on the robustness of the controller (e.g.[44]) or to make it adapt to miscalculated mechanical properties as shown in [43].

5.6.2 Tests and Methodology

A quadcopter equipped with a planar 2-DOF manipulator featuring two revolute joints is modeled in URDF and loaded into the real-time physics simulation *Bullet*. The motor and joint forces are modeled as a first-order system (5.72) and applied locally as external forces resp. torques to the simulated body calculated with equation (5.67). The system matrices ($\mathbf{M}$, $\mathbf{C}$, $\bar{\mathbf{g}}$) are generated from the same URDF description and exported to C code for performance reasons resp. to achieve close to bare metal performance for the subsequent performance benchmarks. The simulation and the controller are implemented in Python, making use of *PyBullet* for the real-time physics simulation, *numpy* for the linear algebra and *cython* to wrap and utilize the generated code. The overall structure is as depicted in Fig. 5.10.

The properties and the configuration of the AM is shown in Fig. 5.11, the gains of the controller were hand-tuned and read

$$\mathbf{K}_v = \mathrm{diag}\left(0, 0, 10, 5, 5, 4, 24, 24\right),$$
$$\mathbf{K}_p = \mathrm{diag}\left(0, 0, 30, 40, 40, 30, 60, 120\right).$$

The first two coefficients are zero for the velocity resp. position in x and y due to the lack of direct control authority over those quantities.

Two different tests are carried out. The first one analyzes the tracking performance of the computed torque controller in simulation over an interval of 20 s, performing several maneuvers whilst actuating the joints. The movement of the manipulator is a significant disturbance to the controlled system. Also, since the controller is just a PD controller with a bias term, inconsistencies between the simulated model and the generated dynamics model would inevitably lead to an unstable system.

The second test is a series of benchmarks dealing with the real-time capability of the controller. The benchmarks are carried out on two different platforms that are representative of on-board resp. off-board control scenarios. Herein, an *AMD Ryzen 1600X* is used as a typical ground station, whilst a *Raspberry Pi 4* was picked as a low-cost, low-power platform commonly used as a companion computer on drones to perform more substantial onboard calculations. Several AM systems from zero to 3 links were generated and analyzed. Prior to counting the number of operations on the left-hand side of (5.43) via *SymPy*'s 'count_ops' function, the expression is simplified with the 'expand' function. This operation is very costly but necessary to be able to get comparable results. For that reason, a maximum of 3 links was chosen for this test (no limitations are imposed by the framework itself). The generated C code is compiled with the *LLVM Clang* compiler using the flags '-O1 -ffast-math -fno-math-errno'. The code uses the double-precision floating-point format. The choice of using '-O1' (minimal optimizations) on the generated C code is motivated by the huge size of the generated code files (up to several megabytes). Performing only minimal optimizations helps with compile times and memory consumption during compilation. Tests with '-O2' only showed marginal improvements in terms of performance. All of the C++ code involving expensive linear algebra is compiled with standard release flags (i.e., '-O3'). For this series of tests, we got rid of all interpreted Python code and replaced the *numpy* functionality with *Eigen*, thus performing our benchmarks on a 'pure' C/C++ codebase.

The average time for one iteration of forward dynamics (5.79) and inverse dynamics (5.81) was measured and averaged over several runs. The inverse dynamics are essential for simulation; the forward dynamics are fed into the controller in terms of known disturbances. The main difference between the two is the additional computational cost associated with the calculation of the inverse mass matrix and the constraint solver, which comes into play for the inverse dynamics.

Lastly, we compare the number of operations of the dynamics equation of a two-link AM using Euler and quaternion parameterization. Furthermore, we test different methods to obtain the dynamics equation from the general Lagrange equations. More precisely: (i) The energy formulation from [101], (5.51), (5.52), which results in a lumped Coriolis force vector. (ii) The factorization method used herein, involves the Christoffel symbols, (5.47), (5.48). (iii) A mix of the two methods, which also yields a lumped expression for the Coriolis term, (5.47), (5.52).

5.6.3 Results

The results of the simulation over a duration of 20 s are plotted in Fig. 5.12. Reference trajectories (position and velocity) have been generated for the joint angles, the altitude, roll, pitch, and yaw angles and fed to the controller. Notice that the controller tracks the reference quaternion obtained from the roll, pitch, and yaw angles. For convenience, those angles have also been plotted by converting the attitude quaternion back to Euler angles.

The results in Fig. 5.12 show stable and satisfactory performance, indicating that the nonlinearities of the system are well compensated by the bias term. The steady-state error in the z-direction between 8.5 s and 10 s is introduced by the pitch angle of the UAV, which leads to a loss in thrust along the z-axis. This behavior is expected from a PD controller (as is the case here). A robust control scheme can be used to fix this problem, or, in this case, a simple addition of an integral term to the virtual control input would be sufficient.

Property	Value
UAV Configuration	Hexacopter
#joints & links	2
Wheel base	1.51 m
Mass (base)	6 kg
Inertia (base)	$(0.48, 0.48, 0.95)\,\mathrm{kg\,m^2}$
Mass (link 1&2)	0.78 kg
Inertia (link 1&2)	$(0.595, 3.824, 3.7)\,10^{-3}\,\mathrm{kg\,m^2}$
CM (link 1&2)	$(0.22, 0, 0)\,\mathrm{m}$
T (joint)	0.05 s
T (prop)	0.1 s
P (prop)	$4500\,\mathrm{min^{-1}}$
P (joint)	12 N m
k_t	$2.165 \times 10^{-6}\,\mathrm{N\,min^{-2}}$
k_p	$5.865 \times 10^{-8}\,\mathrm{N\,m\,min^{-2}}$

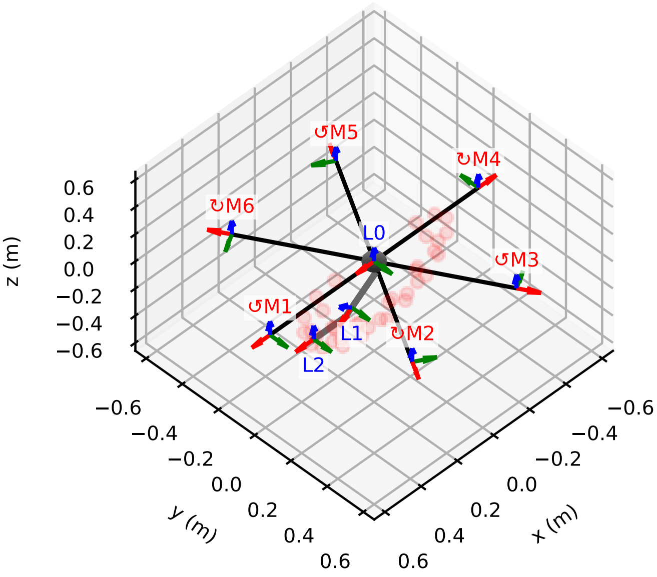


Figure 5.11: Physical properties and configuration of the controlled quadcopter equipped with a 2-DOF manipulator. The range of motion is indicated by the red dots. The present manipulator configuration corresponds to $\bar{\theta} = [45°, -45°]$.

Furthermore, as can be seen from Fig. 5.11 resp. Fig. 5.5, the time constants of the system have been tuned down from 0.2 s to 0.1 s for the propulsion resp to 0.05 s for the joints. The time constants significantly impact the controller's performance as they severely degrade its ability to cancel the nonlinearities via the bias term. The point could be made that AM systems with more but smaller propellers (thus with a shorter time constant) are preferable over systems with fewer but larger propellers.

The benchmark results concerning the computational time and real-time performance on the two model platforms (*AMD Ryzen 1600X* and *Raspberry Pi 4*) are shown in Table 5.2. The results indicate that for the desktop CPU, the number of calculations hardly poses any problem, with an average of just 66 µs per forward dynamics iteration for a very typical UAV equipped with a 2-DOF manipulator, which then grows exponentially, reaching an average of 731 µs for a UAV with a 3-DOF manipulator with a total of almost 6.6 million arithmetic operations. As expected, the inverse dynamics are slightly faster (by the lack of inverse matrix calculations), peaking at 706 µs per iteration.

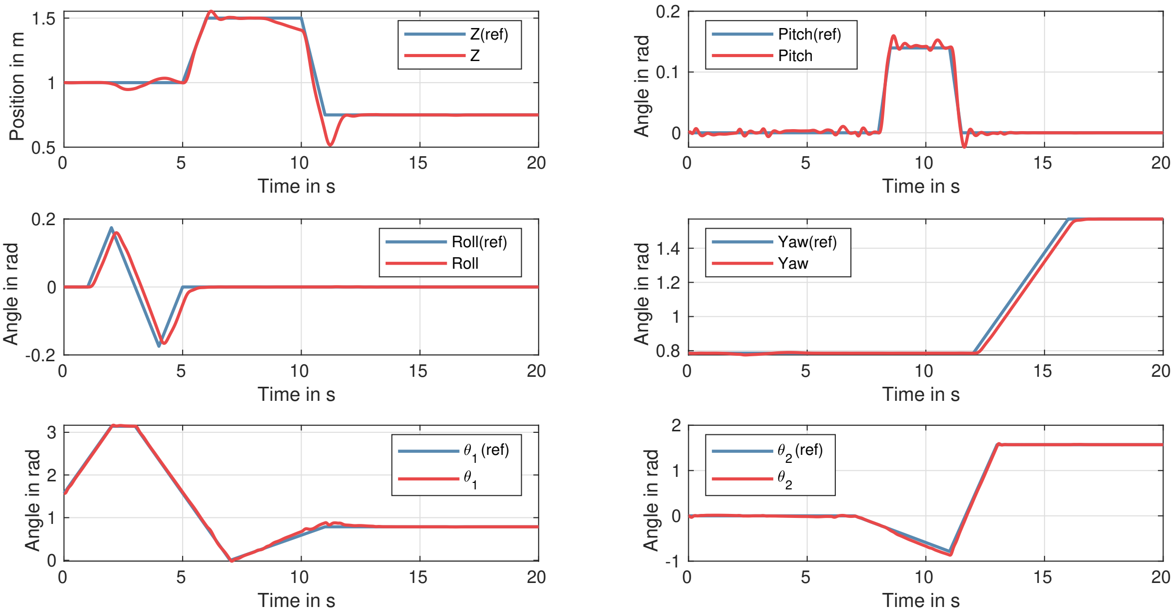


Figure 5.12: Simulation results of the computed torque controller applied to a PyBullet-simulated 2-link AM. Measured and reference signal for the position in z, the roll, pitch, and yaw angles as well as the joint angles θ_1 and θ_2

The Raspberry performs noticeably worse, reaching up to 4.93 ms per forward dynamics iteration in the 3-DOF scenario and only slightly better at 4.89 ms per inverse dynamics iteration. Whether or not this is acceptable for real-time control heavily depends on the application. Typically, the inner loop of commercial autopilots runs at 500 Hz, which would limit the *Raspberry Pi* to a 2-link configuration.

From Table 5.3 it becomes clear that the computational cost increase from choosing quaternion parameterization over Euler angles is substantial. Across the board, the number of operations roughly doubled, passing from Euler angles to quaternion parameterization. This is partially due to the larger state space and where all system matrices are enlarged by one over the Euler model. The numbers also show that even the method itself has a big impact on the complexity of the resulting dynamics expression. We achieved the best results with the ”mixed”-method, (5.47), (5.52) for the quaternion model. The ”energy”-method, (5.51), (5.52), delivered the best results for the Euler model. In all cases, a lumped expression for the Coriolis forces is preferable. Calculating the Christoffel symbols, (5.47), (5.48), always resulted in the largest expression. The presented methods are also vastly different regarding the time it takes for the model to generate. Here, the mixed method was the fastest in all cases, while the energy method was the slowest. This can be attributed to the number of derivatives calculations involved in that method.

		Ryzen 1600X		Raspberry Pi 4	
Configuration	Operations	Forw. dynamics	Inv. dynamics	Forw. dynamics	Inv. dynamics
UAV	5911	2 µs	0.6 µs	10 µs	2 µs
UAV+1 Links	130 328	11 µs	9 µs	60 µs	46 µs
UAV+2 Links	1 041 911	66 µs	63 µs	1031 µs	991 µs
UAV+3 Links	6 549 828	731 µs	706 µs	4925 µs	4886 µs

Table 5.2: Computational performance and number of arithmetic operations of several configurations on two typical compute platforms using equations (5.47), (5.48).

Model	Method	Lumped	Operations	Rel. Difference	Abs. Difference	Gen. Time
Quaternion	Christoffel	no	1 041 911	baseline	baseline	760 s
	Energy	yes	701 721	−32.7 %	−32.7 %	1146 s
	Mixed	yes	698 908	−32.9 %	−32.9 %	613 s
Euler	Christoffel	no	516 078	baseline	−50.5 %	284 s
	Mixed	yes	349 888	−32.2 %	−66.4 %	215 s
	Energy	yes	192 813	−62.6 %	−81.5 %	312 s

Table 5.3: Comparison of the computational cost for quaternion and Euler angle parameterized models for a 2-link AM. Christoffel method uses (5.47), (5.48), energy method uses (5.51), (5.52) and the mixed method uses (5.47), (5.52).

5.7 Conclusion

The kinematic equations for the multi-body AM system were derived, starting from a general URDF description of the system. From those equations, the complete closed-form, singularity-free dynamics of the system was obtained via the Euler-Lagrange equations using quaternion parametrization for the rotational degrees of freedom of the floating base. The dynamics equations of the system were then factorized into the canonical form of mechanical systems. The relation between the generalized forces and the body forces was established whilst providing detailed insights on the calculations resp. the modeling within an AM context.

The obtained dynamics model was numerically validated against its counterpart simulated by the *Bullet* physics engine. The results also confirm that the unity constraint of the quaternion stays satisfied throughout the simulation by enforcing it via a general-purpose holonomic constraint solver.

As an application, the stabilization of an AM using a computed torque controller using the developed dynamics model was shown, indicating satisfactory tracking performance. A robust control approach is, however, necessary to compensate for the loss of vertical thrust due to the inclination of the flying platform. Our data shows that, although the complexity of the closed-form solution grows exponentially, the proposed solution is generally fast enough for real-time applications of typical 2-link AM systems on low-power embedded platforms. Our results also show that the computational cost increase of our quaternion model compared to the common Euler model is quite substantial. The selected method to extract the dynamics equations from the general Lagrange equations shows to have a significant impact on the complexity of the resulting expression.

The current implementation uses a general-purpose holonomic constraint solver to keep the quaternion at unit length, which currently prevents it from being used in a model predictive control framework.

Chapter 6

TRIGGER: A Lightweight Universal Jamming Gripper for Aerial Grasping

*This chapter presents the design, fabrication, and experimental validation of a novel Universal Jamming Gripper specifically designed for aerial applications called TRIGGER (ligh**T**weight unive**R**sal jamm**I**n**G** **G**ripper for a**E**rial g**R**asping). TRIGGER is a soft, omnidirectional, landing-capable aerial grasping system with resilience and robustness to collisions and inherent passive compliance. It miniaturizes the established concept of the Universal Jamming Gripper found on industrial robots and provides key improvements in terms of weight, size and power consumption, and lowers the required activation force. It features a modular, highly integrated design with onboard intelligence and sensors. Leveraging recent developments in particle jamming and soft granular materials, TRIGGER produces* 15 N *of holding force with only a small activation force of* 2.5 N. *Experiments establish the relationship between fill ratio and activation force and further reveal that the holding force can be improved by up to* 52 % *by adding an additive to the membrane's silicone mixture. The grasping concept that envisions TRIGGER mounted on a multicopter as a 'claw' is validated by performing a pick-and-release task under laboratory conditions. Lastly, based on the collected data, a model for robotic simulators is proposed to facilitate the controller design in Chapter 7. The content of this chapter was published in [12].*

6.1 Introduction

To enable the UAV to physically interact with its environment, several drone-grippers featuring a wide variety of grasping mechanisms were developed over the past years (see Table 6.1), e.g., pneumatic soft fingers [121], rigid jaw grippers [122], passive bi-stable grippers for micro UAVs [49], suction cups [123], magnets [124], and flexible limb grippers [125]. Some of these grippers are passively compliant, e.g., the soft finger-based grippers [121], while others are completely rigid, such as the magnetic gripper [124] or the mechatronic jaw [122]. In practice, both approaches are viable, and the optimal choice of the gripper is ultimately application dependant [126]. As a rule of thumb, rigid grippers are less versatile since they are made for specific payloads (e.g., box-shape objects [122], or for specific materials [124]), heavier due to mechanical joints, but in turn, can provide a more secure grasp for the payloads they are optimized for. On the other hand, soft grippers can grasp a broad variety of payloads [121]. Being capable of large deformation, they can passively adapt to the grasped objects. Their soft structure also softens the impact and thus reduces the contact forces that can potentially be dangerous to the drone. Furthermore, soft grippers are generally more tolerant towards position errors which are inevitable due to the ground effect [127]. These features make soft grippers very promising candidates for grasping in complex, unstructured environments [128]. There is however no free lunch, and the gain in versatility is often paid for by compromising in other areas, e.g., cycle time.

In [133], a very particular type of soft gripper was introduced, namely, the *Universal Jamming Gripper* (UG), which is a pneumatic gripper based on the jamming principle of certain granular materials. This UG, being in essence just a bag containing granular material, works via three distinct mechanisms that are simultaneously involved in the grasping process, namely geometric interlocking F_G, suction F_S and friction F_R, which all come into play once the soft membrane of the gripper is pressed against the payload by a force called *activation force*. After jamming (hardening) of the granular material [134], the resulting *holding force* is then the sum of all components: $F_h = F_G + F_S + F_R$. The jamming typically involves creating a vacuum inside the membrane; however, other jamming principles exist, e.g., by

Drone Gripper	Archetype	m (kg)	F_h (N)	F_h/m	Actuation Method	Sensors	Compliant	Omni-directional	Landing-capable	Year
Self-sealing suction cup [123]	suction cup	0.72	12	1.7	pneumatic (pump)	pressure	-	✓	-	2016
Permanent Magnet Hand [124]	magnet	0.30	25.48	8.7	magnetic	contact	-	✓	-	2018
Actively Compliant Gripper [129]	arm+claw	0.30	0.57	0.2	servo (tendon)	/	✓	-	-	2018
Soft Grasper [46]	hand	0.58	10-20	3.5	pneumatic (cartridge)	/	✓	-	-	2018
Small Sleeved Gripper [121]	hand	0.38	52	14.0	pneumatic (cartridge)	/	✓	-	✓	2018
Ultra-fast Robot Hand [35]	hand	0.55	51	9.5	servo (tendon)	proximity	✓	-	-	2019
Mechatronic Jaw Gripper [122]	claw	N/A	2	N/A	servo (direct)	aperture	-	-	-	2020
Soft-Tentacle Gripper [130]	claw	N/A	32	N/A	servo (tendon)	force	✓	-	✓	2021
Micro Bistable Gripper [49]	claw	0.008	2.12	27.0	impact/motor	/	-	-	✓	2021
Hybrid Suction Cup [131]	suction cup	0.050	80	163.1	pneumatic (pump)	pressure	-	✓	✓	2021
RAPTOR [37]	claw	N/A	N/A	N/A	servo (direct)	/	✓	-	-	2022
HASEL Gripper [132]	hand	N/A	0.8	N/A	hydraulic/electrostatic	/	✓	-	-	2022
TRIGGER	UG	0.38	15	4.0	pneumatic (pump)	force, pressure	✓	✓	✓	2023

Table 6.1: Comparison of robotic grippers developed for drones: archetype; the total mass m; maximum holding force F_h; F_h/m the holding force to mass ratio; actuation method; integrated sensors; passive compliance; omnidirectional (indifferent to grasping direction); capability to serve as landing gear; year of publication.

magnetic fields [135], or hydraulic fluids [136]. As discussed in [137], UGs have virtually infinite degrees of freedom that do not need to be controlled explicitly, which gives them the characteristic of being able to grip objects of vastly different shapes thanks to the passively compliant membrane. Given their symmetric shape, they have no preferred in-plane (horizontal) grasping direction and are thus omnidirectional. UGs tolerate relatively large positional and angular (tilt) errors during the grasp [133]. It is shown in [137] that off-center grasping with a positional error of up to 60 % of the membrane's radius does not degrade the gripper's grasping capability. The versatility and the relaxed requirements for positional and angular accuracy and, consequently, less stringent control requirements serve as the main motivation for developing the UG described herein. The structure of the UG is well suited to double as the drone's landing gear (contrary to the ubiquitous soft finger grippers). Furthermore, UGs provide rigid-like grasps by the jamming of the granular material.

This work is the first study that adapts the original concept of the *Universal Jamming Gripper* presented in [133] for use in aerial manipulation. More precisely, this work introduces TRIGGER (ligh**T**weight unive**R**sal jamm**I**n**G** **G**ripper for a**E**rial g**R**asping), a novel design and implementation of a pneumatic UG in a small and compact form factor that is suitable for small to medium-sized multicopters (see Fig. 6.1). The goal is to address some of the open challenges, particularly in grasping scenarios where only very few assumptions can be made

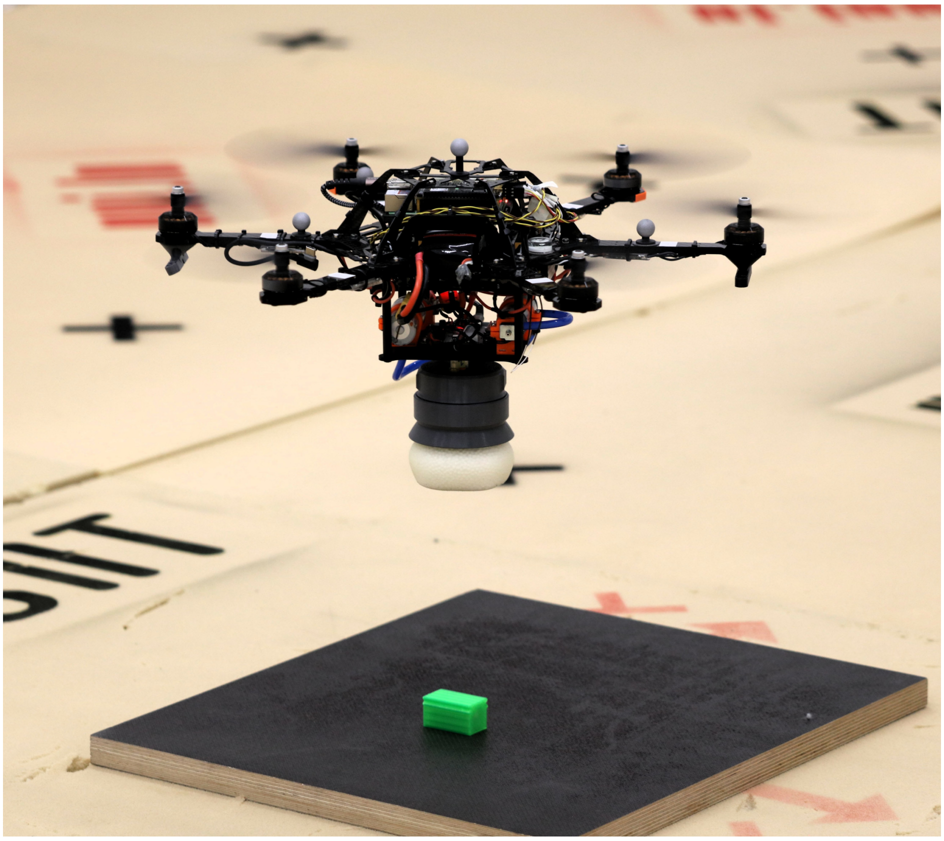


Figure 6.1: General aerial grasping concept. TRIGGER is attached to the UAV's cargo bay (claw configuration). The UAV's landing gear has been removed as the gripper assures this functionality. The box (green) serves as the dummy payload.

about the shape, size, weight, and position of the payload, which paves the way for a variety of practical applications of this type of gripper in the context of aerial grasping. This chapter also shows that UGs are inexpensive and easy to build due to their simple construction and offer distinct advantages over other drone grippers.

The main contributions of this chapter are as follows:

1. To the best of the authors' knowledge, this is the first work that conceptualizes, discusses and uses UGs in aerial manipulation. UGs can relax the aerial vehicle's dependency on the grasping direction and the required positional accuracy. Therefore, they address open challenges regarding aerial grasping in complex environments [128].
2. The introduction of a new design of a UG called TRIGGER, a lightweight and com-

pact gripper for aerial manipulation. The proposed design shares many advantages of available soft grippers in the field, e.g., [47], [37], such as resilience and robustness to collisions and the inherent passive compliance, which decouples the UAV from the environment. However, the salient features of the proposed system lie in the intuitiveness of the design, in the simplicity of its omnidirectional grasping mechanism and in its ability to also act as landing gear. The developed grasping system is modular, energy-efficient, and highly integrated while still being structurally simple and inexpensive to fabricate.

3. Extensive experimental validation of the gripper's design is provided based on the experimental data obtained from the custom test jig. By analyzing the relation between activation force and fill ratio, TRIGGER was designed to work with a much lower activation force than traditional UGs (e.g., [138], [139]), making it suitable for small to medium-sized aerial platforms. Furthermore, it is shown that the holding force can be substantially increased with the help of a silicone additive. Using the collected data, a model of the developed UG is proposed for use in robotic simulators.

4. Lastly, a pick-and-release task with TRIGGER attached to a multicopter is shown, which validates the overall concept.

The rest of this chapter is organized as follows. Section 6.2 introduces the main design challenges and solutions associated with the design and manufacturing of TRIGGER. In Section 6.3, TRIGGER is characterized followed by the presentation of the experimental results regarding the activation force and the impact of the silicone additive on the holding force. Based on the experimental data, a model of the developed gripper for robotic simulators is proposed in Section 6.4. Section 6.5 showcases TRIGGER in an aerial application. Section 6.6 discusses the design and main findings. This chapter is summarized in Section 6.7.

6.2 Concept and System Architecture

This section introduces the general concept of TRIGGER and the associated design challenges in the context of aerial manipulation. An in-depth look is taken at the electro-mechanical, pneumatic, and software components.

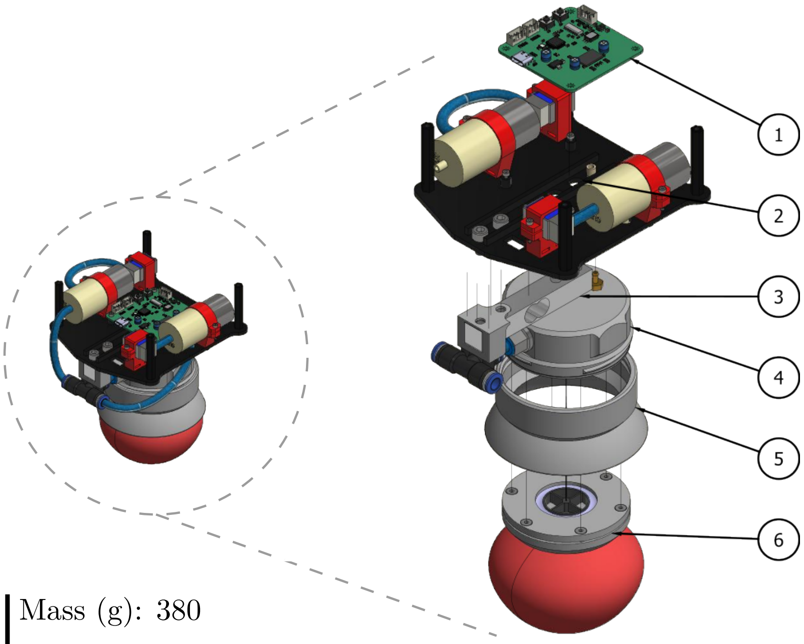
6.2.1 Concept

Multirotor platforms come with many benefits but also with a set of limitations. The most relevant ones are their limited payload capability, the constrained volume for attachments, their underactuated nature, and the challenging dynamics coupling. The dynamics coupling is particularly important for aerial systems carrying manipulators [11], but it also poses a problem for simpler 'claw' setups where only the grasping element gets in contact with the environment. Elastic elements inserted in the construction of the grasping device efficiently reduce the dynamic coupling by softening the hard socks associated with typical grasping operations. Those elastic elements are inherently present in UGs as represented by their soft membrane. UGs are thus an ideal fit, provided they can be constructed to fit the size, weight and power envelope of aerial platforms.

The proof-of-concept aerial platform is a medium-sized (wheelbase of 430 mm), modified *AscTec Firefly* hexacopter with a maximum payload capacity of 1 kg. This airframe conveniently features a cargo bay measuring 120 mm × 120 mm, which is used as the anchor point for the developed gripper. This particular mounting scheme with the gripper oriented toward the bottom is commonly called a 'claw'.

Compatibility with state-of-the-art autopilots (e.g., Pixhawk) is assured by either directly connecting the gripper to the autopilot via UART or by connecting it to the corresponding companion computer using USB. For ease of integration, the concept envisions being directly powered by the UAV's main 3S-4S battery, which eliminates the need to carry a dedicated battery for the gripper. Lightweight construction, modularity, and tight integration of the electronics, the sensors, and the software are the driving concepts of TRIGGER.

To make this work easily reproducible, accessible, and low-cost (below $100, without the manufacturing equipment), the design only requires widely available and inexpensive manufacturing techniques, where a Fused Deposition Modeling (FDM) printer and a high-power single-stage vacuum pump represent the bulk of the cost. Furthermore, the grasper is designed around customary off-the-shelf parts.



Mass (g): 380

Volume (mm^3): 114×131×165

Figure 6.2: Complete assembly of the grasping system. ① Custom controller board, ② base assembly with pneumatic system, ③ load cell, ④ gripper-floor, ⑤ wedge with thread, ⑥ membrane module

The complete grasping system is detailed in Fig. 6.2. Its major subsystems are explained hereafter.

6.2.2 Pneumatics and Mechanics

The role of the pneumatic system is twofold: 1. to pressurize the membrane and thus allow the contained granular material to flow easily within the free, air-filled volume; 2. to vaccumize the membrane and consequently jam (i.e., solidify) the granular material.

UGs can be realized in two distinct topologies, i.e., either as closed-loop or open-loop systems. In closed-loop systems, the fluid surrounding the granular material stays contained within the system. An example of such a system is the magnetorheological fluid-based UG shown in [135] for the hydraulic UG presented in [136]. Generally, these systems have the main disadvantage that the fluid has to stay contained within the system (e.g., in tanks that add weight and cost) and that leakage must be considered as a critical failure mode.

On the other hand, open-loop systems exchange their fluid with their environment. The operating fluid in that case is thus typically air, resp. water for underwater applications [140]. Those systems have the salient advantage that their fluid is abundantly present in their surroundings, which eliminates the storage needs and reduces the severity of leakage, e.g., due to membrane rupture. Open-loop systems are generally better suited for lightweight construction and require less engineering effort.

Therefore, the pneumatic system presented in this paper (Fig. 6.3) has an open-loop structure and uses air as its operating fluid. It consists of two small, non-reversible diaphragm pumps (P1, P2) coupled to two pneumatic solenoid 2/1-way valves (V1, V2). The air pressure in the system is measured by the Microelectromechanical System (MEMS) pressure sensor P. This particular setup is very low cost and has a favorable mass distribution due to symmetry. By design, diaphragm pumps act as one-way check-valves, not restricting the airflow in their nominal direction, which therefore requires closing the valve associated with the antagonistic pump such that they can establish a pressure differential. This particular topology also permits to seal the system off. The membrane can thus remain pressurized (resp. in a state of vacuum) without powering the pumps, which saves energy.

Two 12V, 7 W, `SC3704PM` diaphragm pumps rated for a pressure differential of 46 kPa at $2\,\mathrm{L\,min^{-1}}$ are used. The miniature 2/1 air valves are of type `SC0520FVG`. The low-power pneumatic system typically consumes less than 10 W, contrary to other systems frequently featured in the literature, which use heavy (more than 1 kg) stationary, high-power vacuum pumps in the 500 W range and reaching pressure differentials beyond 80 kPa [133], [141]. Notice that a lower-power system naturally comes with longer cycle times and a lower maximum pressure differential (here approximately 28 kPa), 3× lower than conventional solutions. However, it will be shown in Section 6.3 that this does not adversely affect the performance of the UG.

Concerning the mechanical structure, the modular design approach is shown in Figs. 6.2 and 6.4. It consists of three larger sub-assemblies, namely 1. the base, containing the pumps, valves, and controller board, 2. the gripper-floor, forming the interface between the pneumatic system and the detachable membrane module, 3. the membrane module, which firmly holds

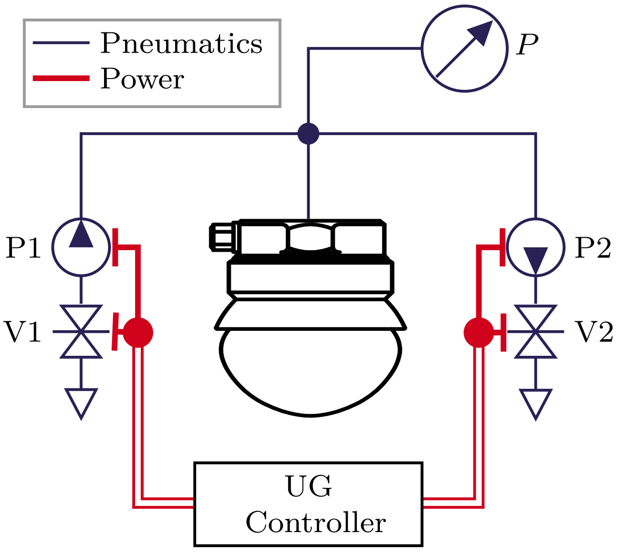


Figure 6.3: Pneumatic system. The jamming gripper is fed by two pumps. Pump P1 inflates the balloon and fluidizes its content. Pump P2 creates a vacuum strong enough to jam the particles inside the balloon. The solenoid valves V1, V2 are required to prevent air from leaking through the inactive pumps. All pneumatic components are actuated based on the control inputs from the main controller.

onto the filled, custom silicone membrane. It contains a paper filter that seals off the filler material from the environment while permitting air to circulate freely. A mechanical support structure prevents it from tearing under load. The membrane module is firmly pressed against the cast-in-place silicone seal on the gripper-floor by screwing the wedge onto the external printed thread to create an air-tight seal.

This modular concept has three main advantages: *first*, it enables quick iteration on membrane module designs; *second*, it allows to quickly and effortlessly swap between different membrane modules during the tests; *third*, it enables the platform to be compatible with different types of grippers, given that some geometries cannot be picked up by a UG (e.g., large flat surfaces), which require highly specialized grippers such as vacuum cups.

TRIGGER is designed to be mounted like a 'claw' on a multirotor; therefore, it does double duty, i.e., it operates as a gripper but also serves as the landing gear. As such, it is dimensioned to withstand the total weight and impact of a landing UAV, which comes with several advantages that are discussed in Section 6.6.

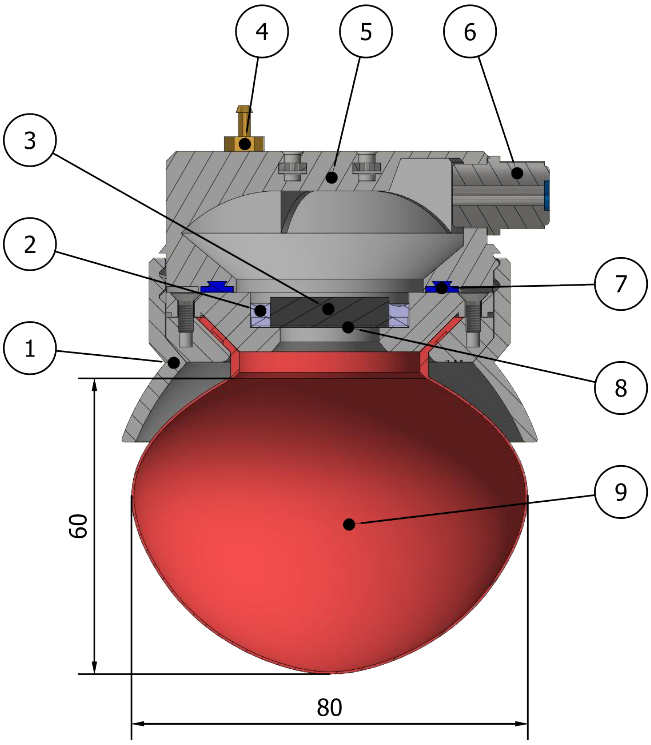


Figure 6.4: Section view of the main gripping module (membrane module attached to the gripper-floor). ① wedge with integrated thread, ② hot-glue seal, ③ mechanical filter support, ④ pressure gauge fitting, ⑤ upper shell, ⑥ main air inlet/outlet, ⑦ cast-in-place silicone gasket, ⑧ paper filter, ⑨ membrane filled with granular filler material.

6.2.3 Material Selection

The gripper's membrane is made from the soft silicone rubber *Trollfactory Type 23*, shore hardness 10 A with 600 % elongation at break. The reasons for selecting such a soft rubber are twofold: *first*, it allows us to widen the tolerances on the membrane's thickness as small deviations no longer have a significant impact on the overall stiffness; *second*, it maximizes the contact area between the membrane and the payload and, therefore, the quality of the grasp is increased. Furthermore, this particular silicone can be mixed with a silicon additive called *deadener* (also sometimes referred to as *slacker*), which gives the silicone more human skin-like physical properties. This further increases the softness of the material and, more importantly, makes it sticky. The intensity of those effects is controlled by the relative amount of additive added to the mixture. This specific silicone rubber is very viscous (14 Pa s) and thus does not flow easily, which has to be considered for the mold design and casting process to avoid trapping air inside the mold and thus creating voids in the thin membrane.

For the printed structural parts, PET-G was chosen over PLA for its higher impact resistance and lower density. Furthermore, PLA is prone to creep under sustained load. The structural parts would not benefit from high-end polymers such as PA6-CF or PEEK as there are no special requirements concerning the stiffness or heat resistance that could motivate such a choice.

Aiming for a lightweight design, EPS is chosen as filler material as it has a density of only $17\,\mathrm{g\,L^{-1}}$, which is by an order of magnitude lower than other commonly used materials such as ground coffee or glass beads (Table 6.2). Moreover, the soft EPS particles develop higher holding forces than rigid particles due to the squeezing effect, which is a result of the elasticity of the EPS beads themselves [141].

6.2.4 Fabrication

Based on previous experience with silicone [142], silicone casting was chosen as the manufacturing process to create the membrane. A three-part mold (i.e., left and right shell, plus core) was printed from PET-G using a common FDM 3D printer. This approach is similar to [136]; however, due to the very thin 0.6 mm membrane and the high viscosity of the silicone

Material	Density ($\mathrm{g\,L^{-1}}$)	Particle Size (mm)
EPS	17	1-4
Coffee	308	0.2-2
Polymer	940	0.1-0.2
Glass	2500	0.2-0.4

Table 6.2: Comparison of filler materials. EPS has by far the lowest density.

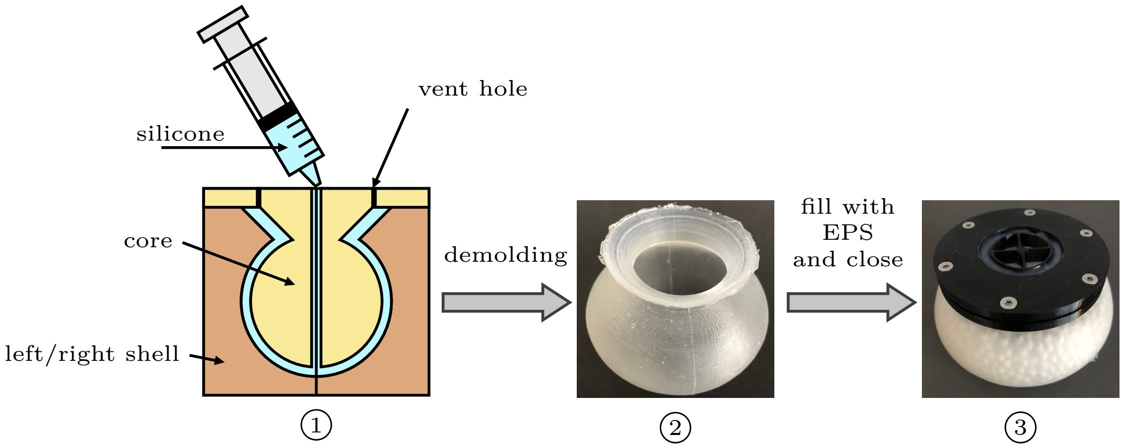


Figure 6.5: Fabrication process of the membrane. ① The degassed silicone mixture is injected with a syringe through the mold's core. The air inside the mold escapes through the vent holes at the top. Precise alignment between the shells is critical as the membrane is only 0.6 mm thick. ② The obtained membrane after demolding. ③ The membrane is filled with EPS and closed with a mechanical assembly that contains the filter. The resulting gripper module is thus fast and easy to swap out in case of damage.

rubber, the process had to be adapted. More precisely, instead of pouring the silicone into the mold, it is injected directly through the core using a syringe (Fig. 6.5). This technique enables very thin-walled castings (assuming proper alignment of the shells). But, more importantly, it allows the silicone mixture to spread evenly with a fairly low risk of catching air bubbles in the process. The usual precautions should be taken when working with silicone, such as properly degassing the silicone after mixing. The membrane has a nominal diameter of 80 mm, a nominal thickness of 0.6 mm, a height of 60 mm, an encompassing volume of 0.2 L and a total mass of only 18 g (without filler).

The structural parts were also fabricated from PET-G using FDM printing. The resulting parts have proven to be sufficiently airtight using optimal print settings. At the mating point

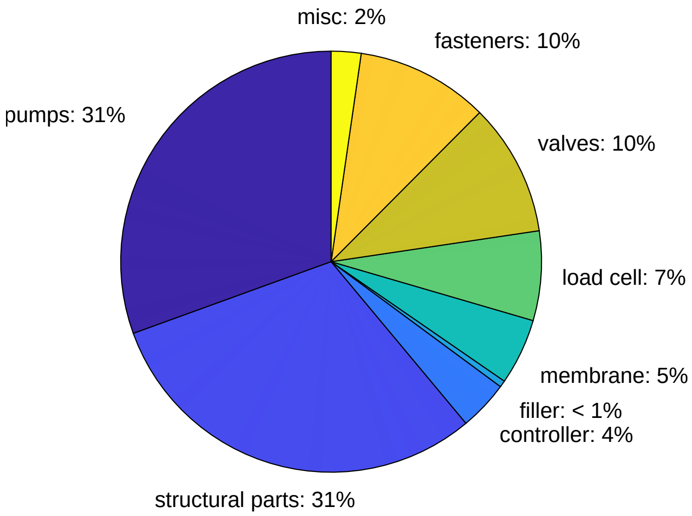


Figure 6.6: Mass breakdown of the 380 g gripper. The pumps and the structural plastic parts form the bulk of the mass.

of two structural parts (i.e., ⑤ and ⑨ in Fig. 6.4), a silicone gasket is introduced that assures an air-tight connection between the two parts.

The filler material consists of a mixture of Expanded Polystyrene (EPS) beads of various sizes ranging from 1 mm to 4 mm. Contrary to rigid filler materials such as glass beads, the softness of the particles gives birth to a squeezing effect which is reported to increase the holding force within certain limits [141]. Another consideration for the choice of the filler material was the density or, more precisely, the resulting weight of the filled membrane. EPS beads have a very low density and thus do not add much mass to the system. Other materials such as ground coffee with a density of $308\,\mathrm{g\,L^{-1}}$ or glass beads with $2500\,\mathrm{g\,L^{-1}}$ result in significant extra weight. 2.2 g of filler material (0.13 L) were added to the membrane, which corresponds to a fill ratio of 66 %.

The total mass of the assembly (380 g) is distributed among the different components as shown in Fig. 6.6. The pneumatic system represents the bulk of the mass (160 g), followed by the structural plastic parts (115 g) and the fasteners (35 g), fittings and tubing (less than 8 g).

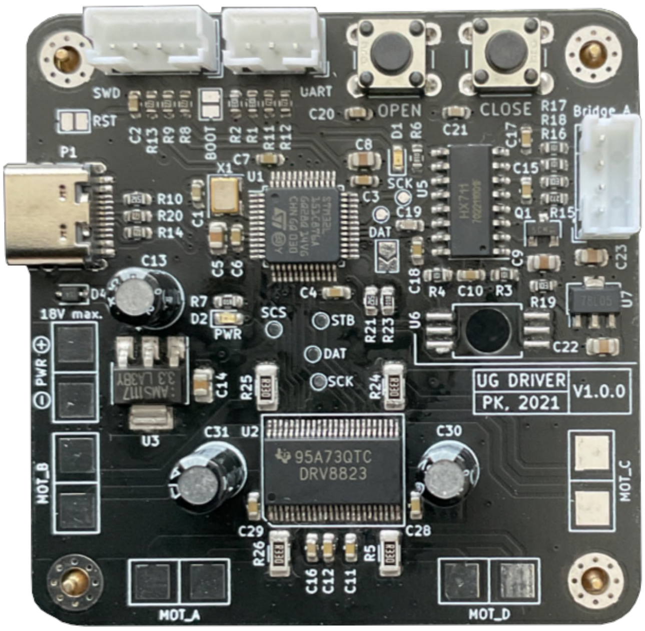


Figure 6.7: The custom controller board with an STM32 MCU at its heart. The quad-channel motor driver powers the pumps and solenoid valves. An ADC with an integrated programmable-gain amplifier and multiplexer samples the analog sensors. Communication with the computer is generally assured over USB.

6.2.5 Electronics and Firmware

The system depicted in the block diagram in Fig. 6.9 is implemented on a single, completely custom 47 mm × 47 mm controller board which is shown in Fig. 6.2 and Fig. 6.7. It is designed to work and integrate easily with common UAV hardware. As such, it can be powered directly from the main power bus of the drone. Furthermore, it features USB and UART serial ports for communication with an autopilot or an off-board computer.

At the heart of the controller is an ultra-low power STM32L1 microcontroller that does the logic processing, collection/processing of the sensor data, the communication with the off-board peripherals, and the control of the quad-channel motor driver that powers the pneumatic hardware.

Due to the low power requirements of the controller (less than 50 mA at 12 V) linear

DC/DC regulators were favored over switching converters for the 5 V and 3.3 V rails as the latter greatly increase the design complexity and cost. The output stage (valves and pumps) is directly powered by the main power bus. Current chopping motor drivers ensure that each actuator operates at its nominal operating point regardless of the bus voltage.

The load cell and the onboard air pressure sensor provide the required data for the system to monitor itself and to work autonomously. The processed sensor readings are exposed via serial to enable more advanced applications. Such applications include activation force tracking, the possibility of feeding back the weight of the grasped payload to the controller as a known disturbance, and the detection of a successful or unsuccessful grasp after takeoff based on the load cell readings. The measured force is referred to as F_m in gram-force 'gf' and the measured pressure as P in 'kPa'. This enables applications such as controlling the activation force and also empowers the internal logic to control the pressure inside the membrane and to prevent conditions such as membrane rupture due to over-pressure and to assure a consistent air pressure while approaching the payload.

Two pressure thresholds were defined, namely $P_{min} = -21\,\mathrm{kPa}$, the lower trigger point, and $P_{max} = 0.5\,\mathrm{kPa}$ the upper trigger point. Those trigger points are used to switch reliably between the 'closed' and 'opened' states of the gripper. In particular, $P \geq P_{max}$ signals that the membrane is full and any additional air would stretch the membrane (consequently increasing the internal pressure). $P \leq P_{min}$ signals that a vacuum is established and thus the gripper is considered 'closed'.

The firmware on the MCU is making use of *FreeRTOS*, running two tasks using preemptive multitasking as shown in Algorithm 1. Task 1 handles sensors and actuation, and task 2 handles serial communication. Inter-task communication takes place over thread-safe FIFO queues.

The underlying state machine (automaton) is shown in Fig. 6.8. At the beginning (power-up), the state of the gripper is undefined as it could be either jammed or fluidized. Thus, starting with ①, a 'startup' cycle is performed which pulls all the air out of the membrane, then refills it until $P > P_{max}$ is reached, then enters the 'opened' state via ② indicating that the gripper is ready for operation. The transitions ③ and ⑥ depend on time and internal

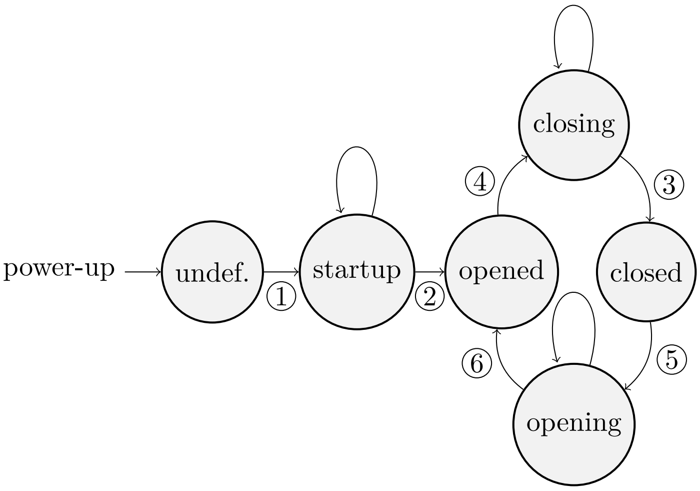


Figure 6.8: Gripper automaton. The gripper's behavior is governed by the state machine, transitioning between states once certain conditions are met. The main states are 'opened' and 'closed' with intermediary states to handle transitions in between. In the beginning, the state of the gripper is not known; therefore, it requires an initial boot procedure. Afterward, the state is well-defined by the data the sensors are providing.

pressure. For the closing operation ③ the condition is $P < P_{min}$ or $t > t_{vacc}$. Transition ④ is automatically triggered if $F_m > F_{thr}$ is measured. The opening condition for transition ⑥ is defined as $P > P_{max}$ or $t > t_{infl}$. The time condition is an additional safety feature in case of sensor malfunction. State transition ⑤ is typically triggered via user command over the serial port. During the 'opening' and 'closing' states, one of the two pumps is running, pumping air either in or out of the system.

Remark 1. *The STM32L1 does not support floating point operations in hardware. For that reason, the controller uses gram-force '*gf*' as the unit of force as it permits performing all of the internal calculations by only relying on integer operations.*

6.2.6 Grasping Procedure

Although usage of the UG by hand is straightforward and allows grasping of a variety of shapes and materials (see Fig. 6.10), a defined grasping procedure is required for the aerial platform such that successful grasps can be achieved without relying on human intuition.

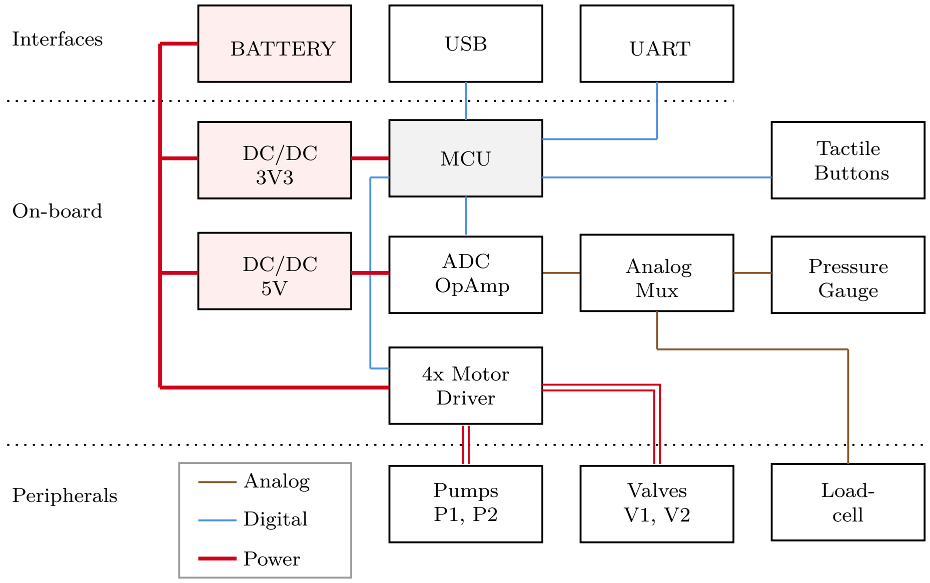


Figure 6.9: Sensors and control topology. The system features a quad-channel current-chopping motor driver for the pumps P1, P2, and valves V1, V2. The sensors (pressure gauge and load cell) are interfaced through a multiplexer into a differential operational amplifier + ADC. Communication with the main microcontroller is assured via USB or UART. The power rails are generated from the battery using cascading linear voltage regulators.

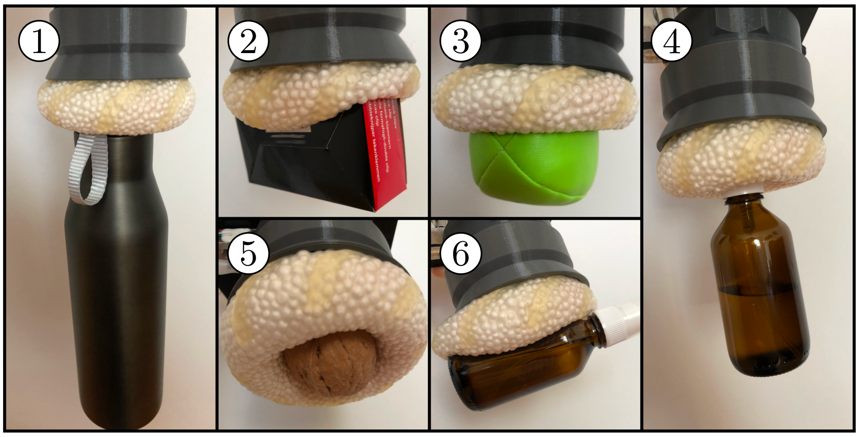


Figure 6.10: TRIGGER grasping a variety of objects, showing its universal grasping ability. ① Empty aluminium bottle, ② cardboard box, ③ semi-soft ball, ④ filled glass flask hold from the small top cover, ⑤ walnut, ⑥ filled glass flask held from the smooth, slippery side.

Algorithm 1 FreeRTOS, sensor acquisition and actuation

```
procedure TASK 1: SENSORS AND ACTUATION
    let $k_{gr}$ be the current gripper state
    let $S(k_{gr})$ be the automaton in Fig. 6.8
    let $f_1$, $f_2$ be lowpass FIR filters
    loop
        collect push button states $\mathbf{u_{bt}}$
        fetch raw sensor data $P^*$, $F_m^*$
        process sensor data $P \leftarrow f_1(P^*)$, $F_m \leftarrow f_2(F_m^*)$
        create state vector $\mathbf{q} \leftarrow (t, k_{gr}, P, F_m)$
        process automaton $\mathbf{u_a} \leftarrow S(\mathbf{q}, \mathbf{u}))$
        apply $\mathbf{u_a}$ to actuators
    end loop
end procedure
procedure TASK 2: COMMUNICATION
    loop
        outbound communication, send $\mathbf{q}$
        inbound communication, receive $\mathbf{u_{usr}}$
        create command vector $\mathbf{u} \leftarrow (\mathbf{u_{usr}}, \mathbf{u_{bt}})$
    end loop
end procedure
```

Typically this procedure consists of four main steps (Fig. 6.11, ① to ④):

① The grasp starts by pushing the fluidized gripper against the payload. Doing so elastically deforms the membrane and the filler material flows freely, distributing itself around the payload. At this point, valves V1 and V2 are still closed such that the free volume remains unchanged. The evacuation phase is then triggered once the measured force reaches the desired activation force, i.e., $F_m \geq F_a$.

② Evacuating the air out of the membrane takes a couple of seconds (governed by the flow rate of the pumps). During that period, the membrane shrinks, and the contact force drops in response to that unless the gripper is further moved toward the payload. In the context of low activation forces, it is essential to keep good contact with the payload. Failure to do so will lead to a poor or unsuccessful grasp as the filler hardens without properly surrounding the payload. We thus track the nominal activation force during this interval. Other publications in this field usually avoid this step by pushing the gripper with a very high force into the payload, e.g., with 17 N as seen in [139], which is, however, not possible with most small to medium aerial systems. The pressure inside the membrane starts dropping once it reaches its minimal volume (see ⓐ). At that point the granular material is compacted. Furthermore, the drop in pressure follows an exponential law as the remaining air molecules become become harder to extract.

③ Once the membrane's internal pressure satisfies $P \leq P_{min}$ (vacuum, ⓑ), the grasping procedure is considered completed. The gripper is then retracted from the payload (here, at a constant velocity). Since the payload is fixated on the support and cannot be lifted, a negative force is measured. The peak of the force corresponds to the maximum holding force F_h. In practice, with the gripper mounted on a UAV, instead of the holding force, the actual weight of the lifted payload would be measured, which can be fed back into the autopilot.

④ Releasing the payload (i.e., opening resp. resetting the gripper) is achieved by pumping air into the membrane ⓒ until $P \geq P_{max}$ is reached at ⓓ, then closing valves V1 and V2. The gripper is now ready to grasp the next object.

The activation force F_a is an essential quantity for a successful grasping operation. An insufficient activation force causes the grasping operation to fail. On the other hand, choosing F_a too high may destabilize the aerial platform and also cause damage to the force sensor and membrane. Therefore, the optimal activation force has to be chosen sufficiently high not to sacrifice performance but also as low as possible to minimize the impact on the aerial platform.

6.3 Experiments

The following section introduces the experimental setup as well as the experiments to, *first*, determine the minimal required activation force for the grasping procedure, and *second*, to access the influence of the silicone additive *deadener* on the grasping performance.

This experimental setup is then also used to collect data regarding the stiffness of the UG

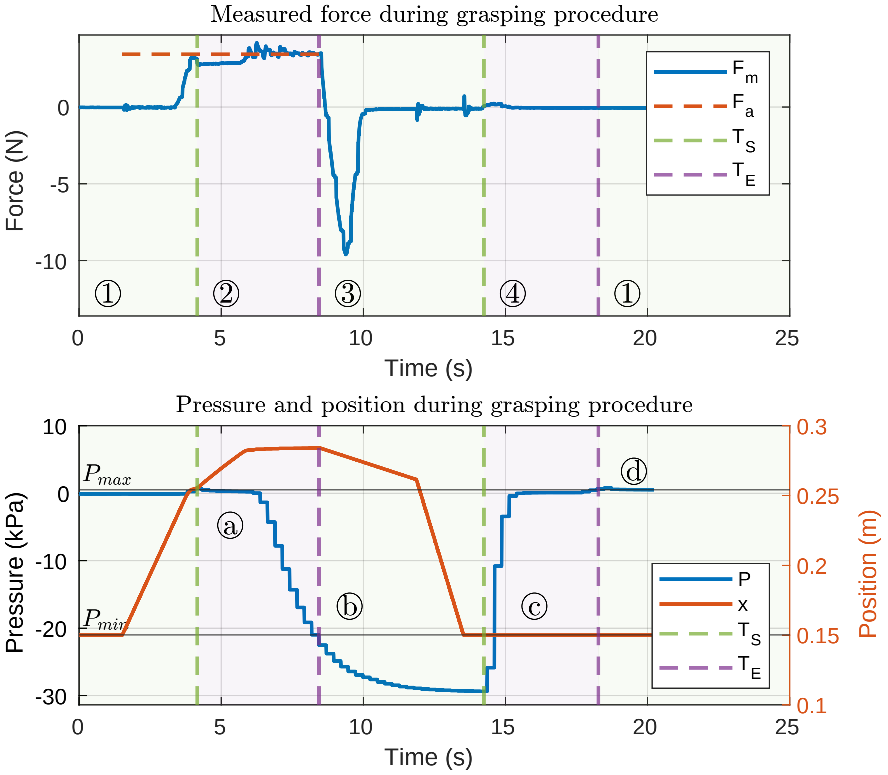


Figure 6.11: Grasping procedure. ① Approach, ② evacuation phase where the air is pumped out of the membrane, ③ retraction phase where the peek marks the maximum holding force, ④ reset of the gripper by pumping air into the membrane, ⓐ the pressure starts dropping once the volume reaches the minimum, ⓑ end of closing procedure triggered by $P \leq P_{min}$, ⓒ pressure rises as air is pumped in, ⓓ end of opening procedure triggered by $P \geq P_{max}$. T_S and T_E mark the start and the end of the state transition between open and close and vice-versa.

in various states with the goal to create a simple simulation model resp. contact model for common robotics simulators (Section 6.4).

6.3.1 Experimental Setup

The experimental jig used for benchmarking is depicted in Fig. 6.12. It consists of a 12 V supply and a single belt-driven linear axis (capable of fast movements) moving the entire gripper assembly fixated to the horizontal beam. The stepper motor is powered and controlled by an off-the-shelf *3D printer board*. The jig does not possess any sensors since they are already integrated into the gripper itself.

Velocity commands u_{feed} as well as position commands (G-codes) are accepted by the custom firmware on the jig's controller. The current position and velocity are sent back to the host computer for the purpose of logging. Likewise, the gripper's controller sends back its current state and sensor data (force, pressure, and input voltage) while also accepting state transition commands (open/close).

Cylindrical test pegs (blue) represent the dummy payloads. They are fixated at the bottom of the linear axis, such that they cannot be lifted. The center of the pegs is aligned with the center of the membrane. They have no features that would allow for geometric interlocking. As such, the results concerning F_h can be seen as a worst-case scenario.

All of the experiments require tracking of a nominal activation force. As such, given a nominal force F_d and the measured force F_m, the error e_f needs to be driven close to zero:

$$e_f = F_d - F_m \tag{6.1}$$

The input u_{feed} is the commanded linear velocity of the sled. The system itself acts as an integrator since F_m is a function of the position, and as such e_f is guaranteed to be driven close to zero by the simple proportional control law

$$u_{feed} = K_p e_f, \tag{6.2}$$

where K_p is a positive constant.

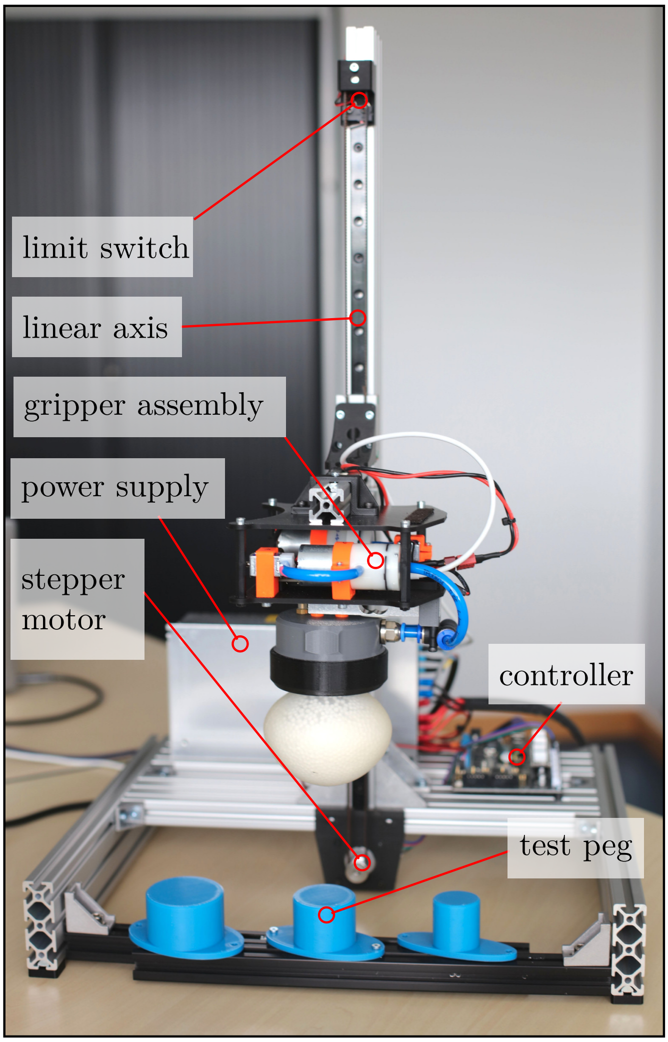


Figure 6.12: Experimental jig. The UG is attached to a belt-driven linear rail, moving the gripper assembly into contact with the test peg (blue). The test peg is a simple cylindrical object with no features allowing geometric interlocking.

6.3.2 Minimum Activation Force

The activation force is a crucial factor for a successful grasp. However, in the context of aerial manipulation, where the base of the gripper is floating, this becomes an even more important factor as any external forces have the potential to cause stability issues. The following key aspects should be considered:

- An aerial platform is severely limited in the amount of force it can apply to its environment before hitting its stability margins.
- Under some circumstances, e.g., when the target is poorly supported, it is impossible to apply a substantial activation force.
- The need for a (large) activation force can thus be seen as a net disadvantage of these types of grippers (this also includes suction cup grippers). Reducing it is therefore considered beneficial.

Recent research reported that there is a monotonic relation between the activation force and the resulting holding force [139]. An older study found that after reaching a certain threshold, the holding force stays constant [138]. The results presented here confirm the findings of both studies and indicate that this threshold depends on the fill ratio of the membrane.

In this chapter, the focus is on small activation forces $F_a < 650\,\mathrm{gf}$ as they are the most useful for aerial grasping. Six test series are thus conducted with nominal activation forces ranging from 150 gf to 650 gf for fill ratios of 66% and 90%. Each test series follows the grasping procedure described in Section 6.2.6 and is repeated eight times using the $\varnothing 40\,\mathrm{mm}$ peg. For this and all subsequent experiments, $K_p = 6$ was chosen experimentally as a compromise to achieve reasonable force tracking while the membrane is still soft and to reduce overshoots when the membrane hardens at the end of the evacuation phase.

The results of this study are shown in Figs. 6.13 and 6.15. The data shows that TRIGGER can reach a maximum holding force $F_{h,max}$ of about 10 N for the test peg with $D = \varnothing 40\,\mathrm{mm}$ and without geometric interlocking. The evacuation period typically takes $T_{SE} = T_E -$

$T_S = 4.3\,\mathrm{s}$ (Fig. 6.11). Furthermore, there is a clear relationship between the fill ratio and the minimal activation forces required to reach the maximal holding force, as illustrated in Figs. 6.14 and 6.15. Lower fill ratios generally reduce the required minimum activation force. The gripper with the membrane having the lower fill ratio of 66% reaches $F_{h,max}$ with an activation force of only 250 gf (see Fig. 6.14). Increasing the activation force beyond that threshold does not significantly increase the resulting holding force. In case of the higher infill ratio of 90%, 650 gf are required to get the same holding force (see Fig. 6.15). Higher fill ratios naturally come with a smaller free volume, i.e., the volume in which the grains can move freely. In turn, the mobility of filler particles is impaired, which then requires a higher effort to redistribute the filler within the membrane during contact. This manifests in a monotonic relationship between the activation force and the holding force. Therefore the required activation force increases with the fill ratio.

It is thus concluded that an activation force $F_a \geq 250\,\mathrm{gf}$ is adequate for successful grasping without compromising the holding capability of the gripper. It is low enough to work on a wide range of UAVs without significantly impacting the stability of the aerial system. Furthermore, lower fill ratios (in the 60% range) are preferable since they require lower activation forces.

6.3.3 Deadener (Additive)

Deadener (also called *slacker*) is an additive that is added to the silicone during the mixing process. It alters the physical properties of the cured silicone by increasing its softness and stickiness.

Herein, the effect of adding 0 % to 15 % of deadener (by weight) to the mixture was measured. At around 15 % deadener, the membrane reached a consistency similar to chewing gum. Further increasing the percentage was thus deemed impractical.

For the test series, four membranes with 0 %, 5 %, 10 % and 15 % deadener were created. The holding force test was repeated 6 times for each of the membranes, with an activation force of 350 gf, on a ⌀40 mm test peg. The results are shown in Fig. 6.16. Starting with no deadener, the expected holding force of around 10.1 N was obtained. A median of 12.8 N and 13.1 N was measured for 5 % and 10 % deadener, respectively. Further increasing it to 15 %,

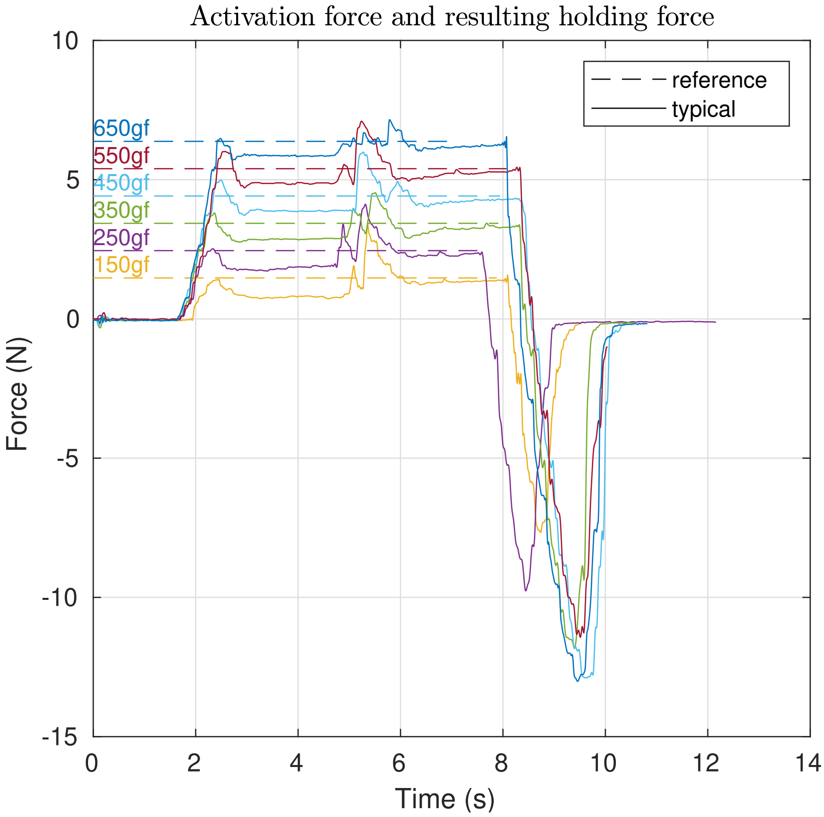


Figure 6.13: Activation and holding force for various nominal activation forces from 150 gf to 650 gf. The nominal activation force is tracked during the hole evacuation phase (2 s to 8 s). The maximum holding force is then measured during the retraction phase (8 s to 10 s), where it shows up as the peak negative force.

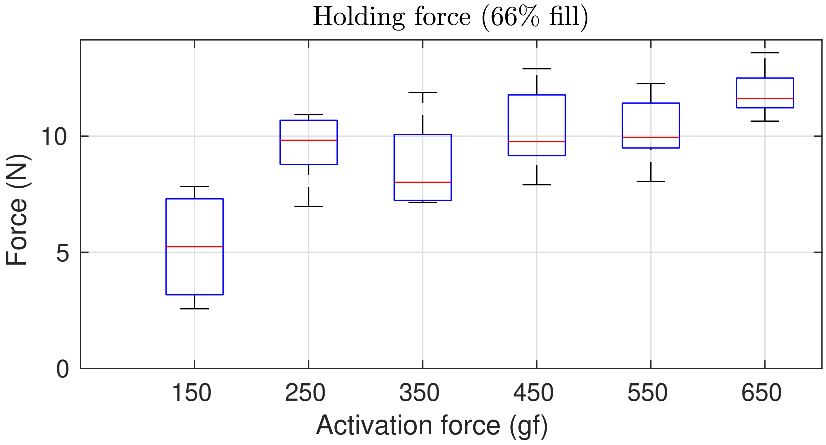


Figure 6.14: Holding force in relation to the nominal activation force, with a fill ratio of 66 %. The holding force stays constant after reaching the threshold of 250 gf.

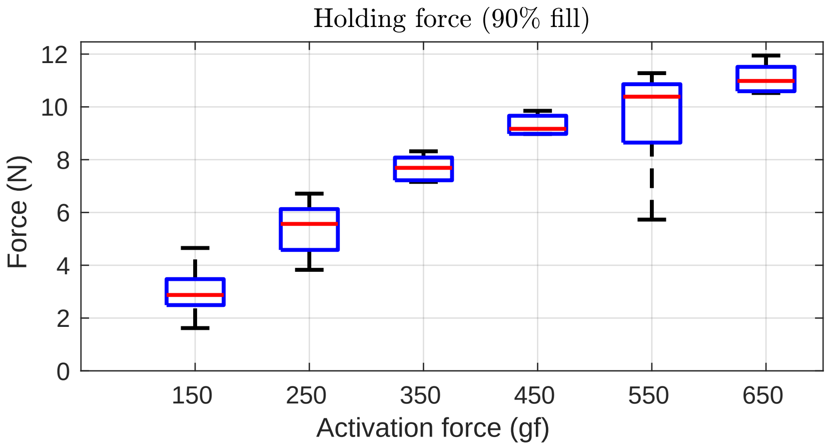


Figure 6.15: Holding force in relation to the nominal activation force, with a higher fill ratio of 90 %. The holding force increases monotonically with the activation force.

resulting in a median holding force of 15.4 N, a significant increase of 52 % compared to no deadener.

The increase in performance can be explained as follows. On one hand, the additive turns the membrane slightly tacky, thus increasing the friction coefficient between the membrane and the test peg. On the other hand, due to the increased softness, the membrane's ability to conform to the test peg's shape is improved, resulting in a larger contact surface. Both of them cause the contribution of the friction F_R to increase, which results in a greater F_h. It should be noted that the stickiness is a temporary effect and dwindles over time as the silicone ages and dust and dirt accumulate on the membrane's surface. For a more lasting effect, a clean environment is thus required. Alternatively, a periodic replacement/renewal of the membrane module (which is fully supported by the presented design, see Fig. 6.2) should be considered.

6.4 Modeling

In the context of aerial manipulation, the UG exhibits challenging dynamic behavior as it transitions from a soft state to a jammed (almost rigid) state. This section proposes a homologous model of TRIGGER based on observations and measurements obtained from

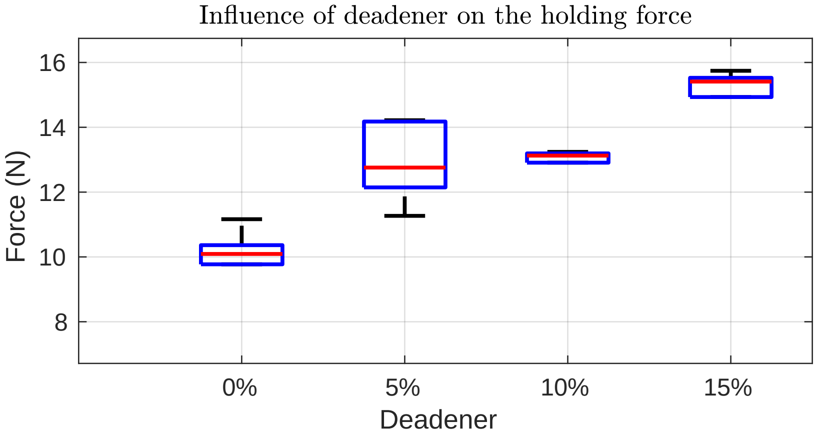


Figure 6.16: Holding force for various percentages of deadener added to the silicone mixture. Increasing the amount of deadener turns the membrane softer and stickier, increasing the holding force.

our experiments.

Our model will faithfully represent the following main aspects of the UG:

1. the normalized free volume β represented as a first-order system, which models the transition between the membrane's open/closed states.
2. the membrane shrinkage $x_a(\beta)$ caused by the changing free air volume.
3. the contact force contribution of the air-filled membrane represented by the compression spring k_{air} as a function of the payload diameter and the normalized free volume β.
4. the contact force contribution due to the lumped elasticity represented by the compression spring k_{lmp}, which takes into account the complete assembly with the gripper being jammed.

We assume negligible damping, constant volumetric airflow and $k_{air} \ll k_{lmp}$ and that the membrane does not touch the ground during the grasping phase (satisfied for any reasonably sized payload).

We propose the contact model shown in Fig. 6.17, consisting of the two non-linear compression springs k_{lmp} and k_{air}, where the latter is of variable stiffness, i.e., dependant on the size of the targeted object. The spring k_{lmp} represents the lumped stiffness of the jammed

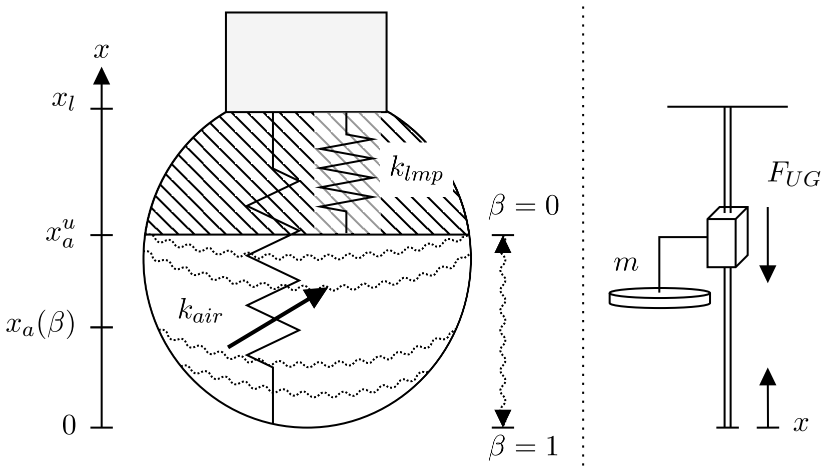


Figure 6.17: The UG membrane is separated into two components: the air-filled elastic membrane and the rest of the system. Both components can be seen as compression springs k_{air} and k_{lmp} in a parallel configuration (left). The simulated system consists of a disk (body) with a mass m attached to a prismatic joint with finite travel subjected to the combined elastic force F_{UG} (right).

filler material and the structural parts of the assembly. The spring k_{air} represents the compression of the air-filled membrane during contact. Its stiffness is tied to many parameters, such as the effective contact area, the non-linear elastic behavior of the membrane, internal pressure, filled volume, and other factors of which most cannot be measured. Its dynamic behavior is, to some extent, akin to an air spring, e.g., [143]. However, such a precise model is very hard to identify and has no practical benefits in this context.

We employed non-linear regression analysis to identify the relation between the depth of entrance x, the payload/peg diameter $D \in (0, 60]$ (in mm) and the resulting elastic force F_{air} as follows (Fig. 6.18)

$$\begin{aligned} F_{air}(x, D) &= a_1 D x^{a_2} \\ &= 102.87 \cdot D x^{1.88}. \end{aligned} \tag{6.3}$$

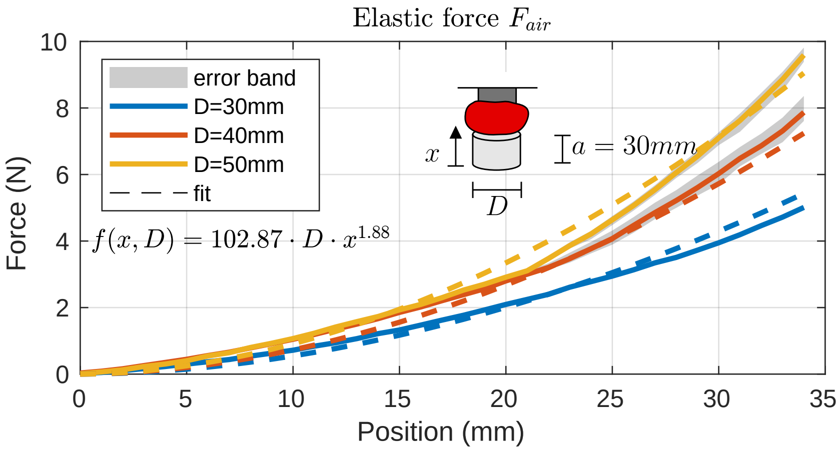


Figure 6.18: Elastic force F_{air} of the inflated membrane during contact with different sized test pegs from 30 mm to 50 mm. Larger effective areas generate a higher elastic force.

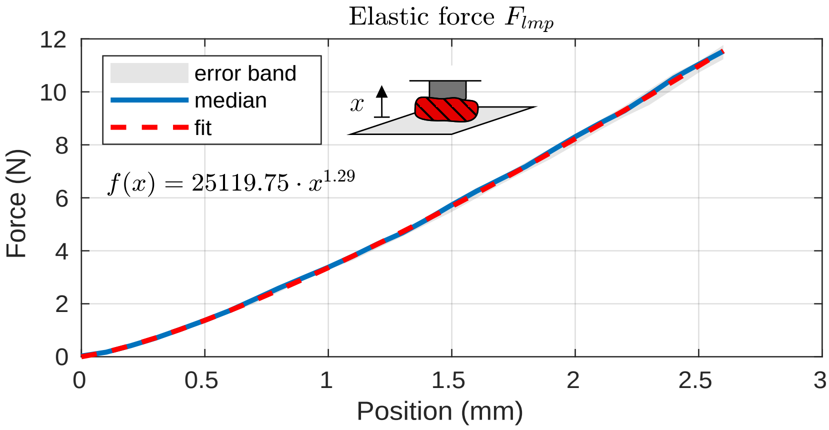


Figure 6.19: Lumped elastic force F_{lmp} of the jammed membrane-gripper system during contact with the test peg. The diameter of the test peg does not affect the results since the shape of the membrane does not change.

Similarly, we identified the lumped elastic force of the system to be (Fig. 6.19)

$$\begin{aligned} F_{lmp}(x) &= a_3 x^{a_4} \\ &= 25119.75 \cdot x^{1.29}. \end{aligned} \tag{6.4}$$

The combined elastic force F_{ug} is then obtained by

$$F_{ug}(x, D) = F_{air}(H(x), D) + F_{lmp}(H(x - x_a)), \tag{6.5}$$

where

$$H(x) = \begin{cases} 0 & \text{if x} < 0, \\ x & \text{if x} > 0, \end{cases} \tag{6.6}$$

accounts for the fact that the springs act in compression only.

Another key aspect represented by the model is the shrinkage of the membrane during the evacuation phase (see Figs. 6.11 and 6.17), where the air is pumped out of the membrane, which consequently shrinks in the process. Herein, it is assumed that this shrinkage follows an exponential law and we designate this internal state by the letter $\beta \in [0, 1]$, where $\beta = 1$ indicates that the membrane is completely filled with air (without stretching it) and $\beta = 0$ means that all the air is evacuated (the filler is jammed resp. in the process of getting jammed). The transition phase is modeled as a first-order system defined as

$$\beta(s) = \frac{R(s)}{1 + sT}, \tag{6.7}$$

where $T = 63\,\% \cdot T_{SE} = 2.8\,\mathrm{s}$ is the time constant and $R(s)$ being a unit step (either 0 or 1, depending on the desired state transition). Based on the value of β, we define three discrete

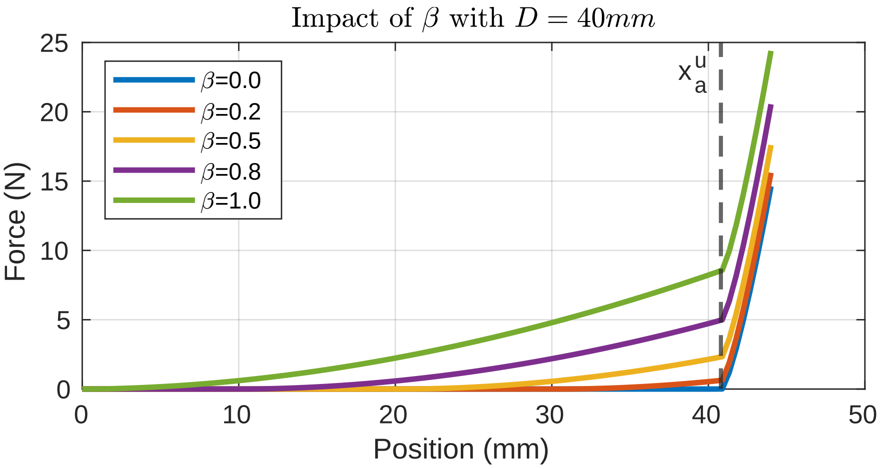


Figure 6.20: Contact force model for different membrane fill ratios from $\beta = 0$ empty/jammed to $\beta = 1$ completely filled with air. The lumped stiffness dominates starting from $x_a^u = 40.8\,\mathrm{mm}$.

gripper states such that

$$k_{gr}(\beta) = \begin{cases} \text{closed} & \text{if } \beta \leq 1\,\% \\ \text{opened} & \text{if } \beta \geq 99\,\% \\ \text{in transition} & \text{otherwise} \end{cases} . \tag{6.8}$$

Next, we assume (simplify) that the free length x_a^u depends on β as described by the linear mapping

$$x_a\left(\beta\right) = x_a^u \cdot \beta, \tag{6.9}$$

where the upper bound x_a^u is defined by the minimal volume occupied by the filler material in the jammed state. Herein, we identified $x_a^u = 40.8\,\mathrm{mm}$ experimentally, with it being the position at which the flattened, jammed membrane first makes contact with the test peg. We thus conclude with the contact force model as shown in Fig. 6.20.

The contact model is the key component in creating a digital UG for use in a robotics simulator such as *Gazebo*. We propose modeling the UG as a disk with a diameter of 80 mm and a mass of $m = 20\,\mathrm{g}$ (weight of filler plus membrane). That disk is attached to a prismatic joint that simulates the shrinkage of the membrane as a function of β, the intrusion of the payload into the membrane, and is subjected to a force F_{UG} governed by the contact force model (6.5). The travel limits of the joint are defined to be in the range of $[0, x_l]$, where $x_l = 60\,\mathrm{mm}$ is the height of the membrane (see Fig. 6.17, right). The UG's internal state β is governed by the system model (6.7).

As for the grasping part, we suggest considering a grasp as successful as long as an activation force greater than 250 gf is maintained with the payload during the entire evacuation phase. A successful grasp should then establish a rigid connection between the gripper and the payload motivated by the jamming (hardening) of the filler material. That connection should be removed if $F_m \geq F_h$ or as a result of the state transition $\beta > \epsilon > 0$, where ϵ is a very small value.

Another application of the contact model (6.5) could be the estimation of payload size with the help of characteristic curves of the elastic forces identified during the initial contact, as shown in Fig. 6.18. However, this requires precise measurements of the position, which may not be available in real-world conditions.

6.5 Aerial Application

This section briefly presents a pick-and-release manipulation task with the UG attached to a UAV. The overall goal of this experiment is to validate the fundamental concept shown in Fig. 6.1.

The proof-of-concept platform is based on the frame of the *AscTec Firefly* and features a *Raspberry Pi 3* with a *Navio 2* running the *PX4* autopilot. A custom *PX4* firmware module communicates with the gripper and links one of the remote control channels to trigger its opening resp, closing state transition. Herein, the UAV is carefully manually piloted in position control mode, relying on human intuition to keep a sufficient amount of activation force.

The whole experiment is pictured in Fig. 6.21 and the supplementary video is available

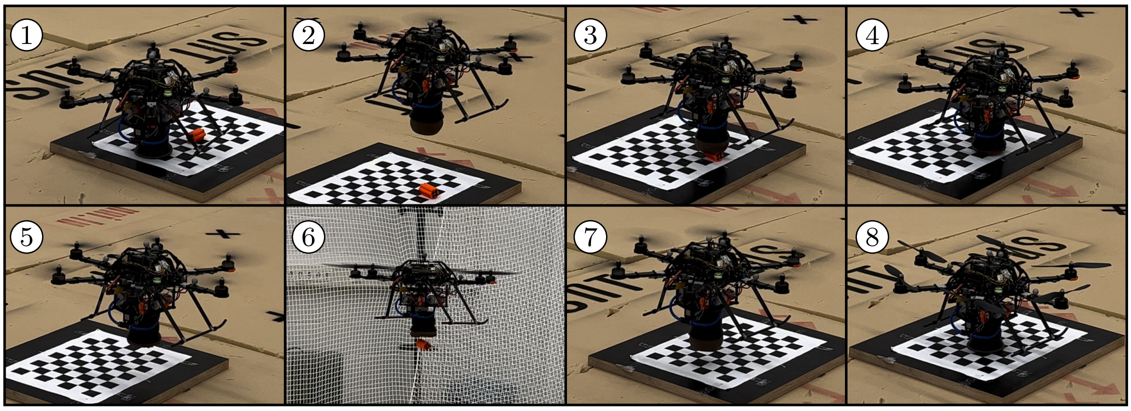


Figure 6.21: Aerial grasping application. ① The UAV rests on the gripper, ready for takeoff. ② The UAV approaches the payload (orange). ③ The UAV gets in contact with the payload. ④ The air is pumped out of the gripper and the payload is gripped. ⑤ Takeoff with the payload. ⑥ Air is pumped into the gripper, which releases the payload. ⑦ The UAV approaches the landing platform and ⑧ lands on the gripper (the landing gear does not touch the ground).

online [1]. The task of the UAV is to take off, grab the payload (orange), and drop it in the drop-off area. During the setup of the experiment, the UAV is manually placed on the checkerboard. The gripper is then closed using the push buttons on the controller board. The membrane then forms a flat and rigid surface for the UAV to rest on. At that stage, the UAV is ready to take off.

After successful takeoff, the membrane of the UG is fluidized (triggered by the remote) and the payload is approached. Ideally, the membrane would hit the payload dead center, but the UG is fairly tolerant to positional errors. After the activation force crosses a threshold of 250 gf, the evacuation phase is automatically triggered and the filler material hardens, creating a firm grasp on the payload. Now, the drone is piloted to the drop-off zone, where it releases its payload by fluidizing the gripper, triggered via the remote control.

Lastly, the UAV is piloted back to the checkerboard, where it safely lands on the UG. Notice that the landing gear, although present, never touches the ground. It was kept for safety reasons only.

---

[1] Supplementary video: https://youtu.be/Az5bXnZUNlY

6.6 Discussion

The interest in developing soft grippers for aerial vehicles stems from the fact that by leveraging the properties of soft materials, soft grippers are a natural match for aerial grasping. In contrast to their rigid counterparts, soft grippers are tolerant toward unknown object geometries and surfaces and do not require high positional accuracy for successful grasps. By developing a lightweight soft jamming universal gripper attached to a UAV we further advanced the potential of soft aerial grasping. Compared to available soft grippers for aerial grasping, the developed system exhibits several distinguishing characteristics.

First, the developed gripper is highly integrated and modular. The tight integration of the electronics, software, sensors and mechanics leads to significant weight savings and enables a well-defined grasping procedure that can be automated for use in autonomous systems. The modularity not only helps in iterating the design rapidly, but it also addresses some concerns typically associated with UGs. By having an explicit interface between the grasping part (membrane module) and the supporting hardware, we assure that the membrane is quick and easy (toolless) to swap in case of damage. Other types of pneumatic grippers could also make use of this interface, e.g., suction cups. Thanks to the specific characteristics of our UG's construction, it is particularly well suited for aerial vehicles, comparable to the typical multi-fingered soft grippers, but with some unique features (e.g., omnidirectionality, or the ability to use as landing gear).

Second, TRIGGER is omnidirectional in contrast to other available soft aerial grasping systems (e.g., claws), which are sensitive to the angle the payload is approached. The same applies to lateral position errors, where the UG tolerates displacements as large as 60 % of its diameter. This relaxes the requirements in terms of necessary grasping accuracy, which is especially advantageous for aerial systems that are subjected to external disturbances (e.g., wind gusts, ground effect) and sensor inaccuracies. During contact, the UAV retains most of its degrees of freedom due to the gripper's elasticity. As such, it can still rotate (pitch and roll) and therefore preserve hover conditions, but at the same time, the translational degrees of freedom are soft-locked by the friction between the payload and the gripper. This would address one concern associated with soft finger grippers. The authors of [46] stated that during their experiment, the multicopter had to land on the ground due to ground effects and the resulting lack of precise position control. During our aerial experiment, we did not observe such a problem as the UAV passively stayed locked in place during the grasp.

Third, unlike soft fingers, our UG forms a rigid-like flat surface once a vacuum is established and thus enables the UAV to rest on it. This feature makes our manipulating UAV system exceptional as it removes the need for a dedicated landing gear that also often interferes with the attached gripper resp. the sensors required for autonomous grasping. Moreover, using the UG as landing gear further reduces the weight of the aerial system. The system is also able (within limits) to compensate for some terrain imperfections (e.g., slanted surfaces or small rocks), assuring optimal takeoff and land conditions. Traditional soft finger grippers often cannot prevent the payload from moving after the grasp is established, which can create further disturbances during flight. Contrary to our UG, which forms a system behaving much more akin to a single rigid body due to the jamming of the granular material.

Forth, hard shocks typically associated with the impact of two bodies are problematic both from a mechanical perspective, like the risk of damage, and also from a control perspective (e.g., potential instability). Passive mechanical compliance alleviates this problem by spreading the impact over a larger time interval. The developed UG is completely soft during the first contact phase and is thus passively compliant and absorbs and dampens shocks. This applies to both landing and grasping scenarios.

Fifth, our gripper develops 15 N of holding force on our test peg that do not allow for geometric interlocking and thus purely relied on friction and suction (to a much lesser extent), which is, therefore, a worst-case scenario. As indicated in [138], geometric interlocking can dramatically increase the holding force. Comparisons with other UGs are hard to make due to the lack of a standardized test procedure. However, comparing our results with the work of [138], [144] and [139], it can be said that the measured holding force for objects without geometric interlocking is in the same neighborhood, i.e., 10 N-30 N, whilst being significantly lower power (less than 10 W against several hundreds of watts). Consequently, also the cycle times of our solution are longer (11 s against 4 s) and have to be handled properly, and failure to do so will result in degraded or even unsuccessful grasps. Therefore, in the larger context of UGs, our results indicate that high-power pumps are not strictly required. In practice, fitting larger, heavier pumps is limited by the payload capacity of the aerial platform.

6.7 Conclusion

This chapter introduced TRIGGER, a novel, highly integrated, and lightweight UG with a low activation force requirement for aerial manipulation. The presented gripper was experimentally validated and the optimal minimum activation force required to work reliably was determined. It was concluded that the relation between the activation force and the resulting holding force is highly dependent on the fill ratio, i.e., lower fill ratios are preferable since they lower the required activation force. It was shown that significant holding force improvements are obtainable by using a silicone additive (*deadener*), which yielded a 52 % higher holding force compared to the reference case without deadener.

Based on our experimental results, a simulation model was developed that faithfully represents the most relevant aspects of TRIGGER intended for numerical robotics simulators. A successful pick and release task under lab conditions was carried out with TRIGGER attached to a hexacopter.

Regarding the design, it is believed that significant weight savings can still be achieved by having some of the structural components fabricated out of carbon fiber (e.g., the base plate), which would, nonetheless, increase the costs of the solution. The membrane casting technique allows the creation of internal and external features on the membrane, which could potentially have interesting effects on the behavior of the gripper. The design and the potential effects of those features could be a topic of future work.

As shown by the experiments, the presented gripper benefits from controlling the activation force during the grasping interval. Such a force controller is thus designed in Chapter 7.

Chapter 7

Autonomous Aerial Grasping

Autonomous grasping is the symbiosis of various components ranging from the grasping hardware to the various onboard sensors and control algorithms. Contrary to ground robots where a human operator can manually control the whole grasping procedure, AMs require at least some degree of automation as the aerial platform is floating and thus in constant need of readjustments of at least 4-DOF which is considered too challenging for human beings. An automation concept is thus required that relieves the operators of some of the mental burden such that they can focus on instructing and supervising the aerial system. Such a concept is presented herein. It involves the development of several key components such as a fast GPU-accelerated object detection pipeline, algorithms for force-controlled grasping and trajectory optimization, and lastly, the system architecture. The concept is finally validated in simulation by autonomously picking up an IED and transporting it to the drop-zone.

7.1 Introduction

Autonomous grasping requires the interplay of many different components. Contrary to ground-based robots, aerial robots put a much higher mental load on the operator due to their high mobility, fast dynamics, and by having to control their four degrees of freedom simultaneously. Thus, it is necessary to provide adequate support to the operator so that grasping operations can be performed safely without overburdening the pilot. One solution could be to take away explicit control over some DOF, leaving the operator with less choice and thus a lower mental load. A very basic example of that is the ’altitude control’ mode found in common autopilots, which still allows the drone operator to change the altitude, but without requiring constant adjustments to keep it. Inhere, a similar approach is followed; however, due to lack of proper HMI, the solution is fully autonomous. Nonetheless, the different building blocks are easily reusable for such a semi-autonomous solution with an adequate HMI.

The solution envisioned here has three main components:

(i) Object detection to reliably detect the target. This information is then made available to all algorithms, which can help keep the target in view.

(ii) Trajectory optimization; this component defines the motion of the AM according to a set of objectives. In other words, it helps the operator to pilot the UAV safely in accordance with the desired behavior and rules.

(iii) Force control; Manually keeping track of the activation force is virtually impossible. Therefore, a tracking force controller is introduced to handle this challenge.

The detection of the payload is related to the broad field of computer vision and, more precisely, object detection. The authors of [145] classify object detection into four distinct categories based on their approach taken: feature-based detection, template-based detection, classifier-based detection, and motion-based detection. Deep learning-based detection methods can be added to those approaches [146], representing the most powerful and versatile object detection algorithms which nowadays drive most state-of-the-art object detectors [147].

Within the IED context, object detection becomes particularly challenging as the nature of the object is a priori unknown. The initial detection or classification of the object as an IED is the responsibility of the specialized personnel in charge of the disarmament. One approach to this problem could thus be the operator selecting the object resp. features of interest as seen by the AM’s camera. Those features can then be tracked using optical flow, which does not require a priori knowledge [148]. Likewise, template-based object detection

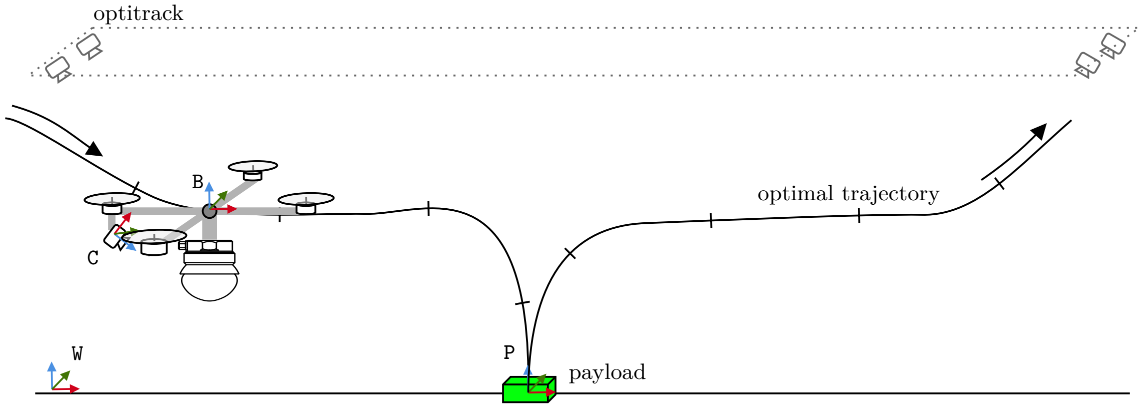


Figure 7.1: Autonomous grasping setup with coordinate frames. The AM `B` is equipped with a depth camera `C` that localized the payload `P`. The AM is localized by the Optitrack system.

could present a good candidate. A reconnaissance drone could be tasked to take pictures of the potential IED, which then serve as a template for the object detector. Contrary to deep neural networks, template-based methods do not require labeling or learning resp. retraining. However, also object detection using deep neural networks can be applicable in some cases, as IEDs are often disguised as commodity items, which are reliably recognized by state-of-the-art neural networks [149].

Each of these techniques has its pros and cons. However, within the scope of this work, it is assumed that the payload has easily identifiable features. A feature that is easy and reliable to detect is the color (hue), as it remains unchanged depending on the viewing angle and is also relatively stable under varying lighting conditions [145]. Within this chapter, the object detection part is greatly simplified as the focus is on the holistic concept rather than on object detection. However, due to the limited onboard computing power shared between the vision pipeline and the MPC, the developed object detection pipeline was specifically designed to be very fast, making optimal use of the onboard computer's resources (CPU and GPU). For easy detectability, our dummy payload was crafted with distinct features (plain surfaces and a vibrant, uniform color).

Trajectory optimization formulated as an optimization problem, resp. MPC, is a well-established concept for aerial robots. The authors of [150] have demonstrated perception-aware trajectory optimization on micro UAV equipped with a camera. Obstacle avoidance with a self-adaptive differential evolutionary algorithm was shown in [151]. In [152], the authors showcased dynamic obstacle avoidance with safety guarantees using a stochastic model. The solution developed herein is based on these developments and adopts it to the specific grasping approach mandated by the UG.

It follows the design of a tracking force controller that assures a defined contact force between the gripper and target during the whole grasping interval, which is required by the UGs as discussed in Chapter 6. The approach is inspired by impedance control [153], [154], although adapted for the much simpler use-case ('claw' configuration) which only exhibits uni-axial dynamics.

Finally, the concept is validated in simulation, where a hexacopter equipped with a simulated UG and a depth camera is tasked to perform the disposal of an IED while avoiding any obstacles in its path. A *mission planner* orchestrates all the developed components and ensures the successful and autonomous execution of the task.

7.2 Object Detection Pipeline

Visual servoing is an essential part of autonomous grasping as it guides the AM towards the targeted object using sensor feedback. By doing so, it relies heavily on the information provided by the visual sensors, e.g., cameras. The camera's raw image information is fed through several processing stages such as (morphological) filtering, feature extraction, cropping, and pooling. This chain of processing stages forms a vision pipeline in which the raw image enters and (after several processing stages) the location of the payload (either 2D or 3D) is calculated.

The challenging aspects of a vision pipeline are robustness and computational cost. This is especially true for mobile autonomous robots, where onboard computing power is limited due to energy, weight, and size constraints. It is thus key to leverage all of the hardware at hand.

For this application, the UAV is equipped with the stereo camera *Realsense D435* that

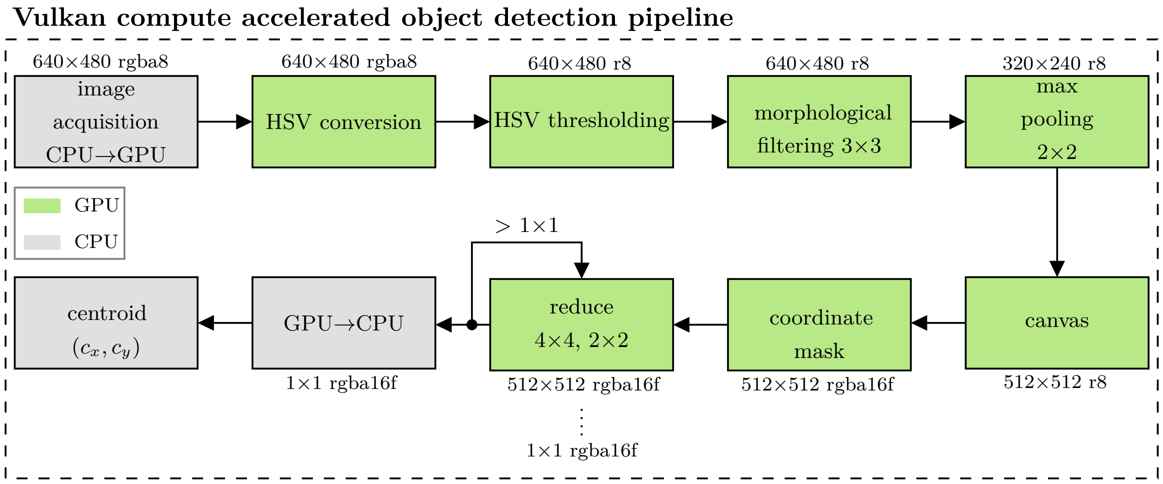


Figure 7.2: After uploading the image to the GPU, it is processed by seven compute elements (shaders) which perform feature extraction, filtering, pooling and clustering. Each stage uses an ideal representation of the image data. The final result is then transmitted back to the CPU, and the centroid of the object is calculated.

provides an RGB image with an associated depth channel. A Raspberry Pi 4 provides onboard computing power and is thus responsible for executing the object detection pipeline represented in 7.2. Lacking support for OpenCL, popular computer vision frameworks such as OpenCV have to execute all computations on the CPU (thus not using the GPU), resulting in a very high all-core CPU load with the potential to starve other tasks that are executed in parallel. Since the Raspberry Pi is also serving as the UAV's autopilot (in addition to running the NMPC), it is beneficial to offload a maximum of the compute involved in the vision pipeline to its Graphics Processing Unit (GPU). Herein, Vulkan API's compute shaders (higher-level compute kernels) are leverage to offload work to the GPU. This approach has the advantage of being very portable as it is supported by virtually any modern device (even in the embedded space) supporting the Vulkan graphics API. The main merit of the presented pipeline is thus its speed and portability, therefore making optimal use of the hardware.

The object detection pipeline depicted in Fig. 7.2 consists of 7 distinct processing elements which are sequentially applied to the raw image obtained from the camera. These processing blocks (kernels) are executed on the GPU, leaving the CPU free to perform other tasks (e.g., serve interrupts, process the MPC, communicate with ROS, etc.). The only time the CPU gets involved is during the image acquisition phase and for the transfer of the images to the GPU memory resp. back to CPU memory. The dispatch of each compute kernel is performed in several workgroups, where each workgroup runs several invocations of the compute shader in parallel (depending on the hardware). The compute kernels are described as follows:

1. **HSV conversion:** The native color space of the camera is in RGB format, and while that is optimal for displays, it is not useful for color detection. Thus, as a first step, a color-space conversion is performed from RGB to HSV color space, where the hue, saturation and value are calculated and separated into their own channels respectively.

2. **HSV threshold:** In HSV color space, filtering the image for a certain color (hue) is trivial since this information is now condensed into a single channel. However, filtering exclusively on hue is not very robust and thus also saturation and value are considered. The filtering is performed by

   $$p_{out}\left(x,y\right)=\bigcap_{i}\left(v_{min,i}\leq p_{in}\left(x,y\right)\leq v_{max,i}\right),\qquad i\in\left\{h,s,v\right\}. \tag{7.1}$$

   The resulting image is thus binary (either in range or not).

3. **Morphological filtering:** The obtained binary image might contain some noise. The morphological 'close' operation is used to remove those stray pixels by performing an *erode* and *dilate* pass with a 3×3 kernel.

4. **Max pooling:** After feature extraction and filtering, the maximum values of all pixels in 2×2 squares are calculated. Doing so reduces the amount of information (decreasing the size of the image by a factor of two) while preserving the features in the image. This yields a noticeable speedup in the subsequent steps of the pipeline, without having a huge impact on the accuracy.

5. **Canvas:** The binary image is padded such that the resulting image is square and with dimensions equal to a power of two. This is a necessary condition to obtain a 1×1 image with consecutive divisions by two.

6. **Coordinate mask:** This operation assigns each pixel a weight [155] based on its x and y coordinate and value in the image such that

$$\begin{pmatrix} c_{m,r} \\ c_{m,g} \\ c_{m,b} \end{pmatrix} = \begin{pmatrix} \frac{p(x,y)\cdot x}{width} \\ \frac{p(x,y)\cdot y}{height} \\ p(x,y) \end{pmatrix}, \tag{7.2}$$

   where $p(x, y)$ is a function that performs a pixel lookup into the source image at the location x and y. At this stage, the pixel values are binary (either 0 or 1). The results are stored in the red, green, and blue channels of the output image.

7. **Reduce 4×4/2×2:** Calculates the weighted average over an area of 4×4 (or 2×2) pixels. This operation is performed in multiple passes and leverages the parallel compute of the GPU. Each thread with the invocation id u, v thus calculates each color channel $k \in \{r, g, b\}$ by

$$c_{a,k} = \frac{\sum_{i=1}^{4} \sum_{j=1}^{4} p\left(4u + i, 4v + j\right)}{4}. \tag{7.3}$$

   Consequently, each pass reduces the size of the image by a factor of 4, and terminates once the size of the output image reaches 1×1. The information required to calculate the centroid is then stored in the RGB channels of that last picture.

Finally, the normalized position of the centroid of the detected color blob is thus calculated by

$$\bar{\mathbf{c}} = 2\left(\frac{c_{a,r}}{c_{a,b}}, \frac{c_{a,g}}{c_{a,b}}\right) - 1, \qquad c_i \in [-1; 1]. \tag{7.4}$$

Combining this information with the camera's depth map $\mathbf{D}$ and the knowledge of its intrinsic I_{cam} yields the 3D position in camera space via deprojection, which is denoted as ${}^{\mathtt{C}}\bar{\mathbf{p}}_{\mathtt{v}}\left(\bar{\mathbf{c}}, \mathbf{D}, I_{cam}\right)$. Since the homogeneous transformation ${}^{\mathtt{B}}\mathbf{T}_{\mathtt{C}}$ is known from the geometry of the AM and ${}^{\mathtt{W}}\mathbf{T}_{\mathtt{B}}$ is measured by the Optitrack system, the payload's position in world

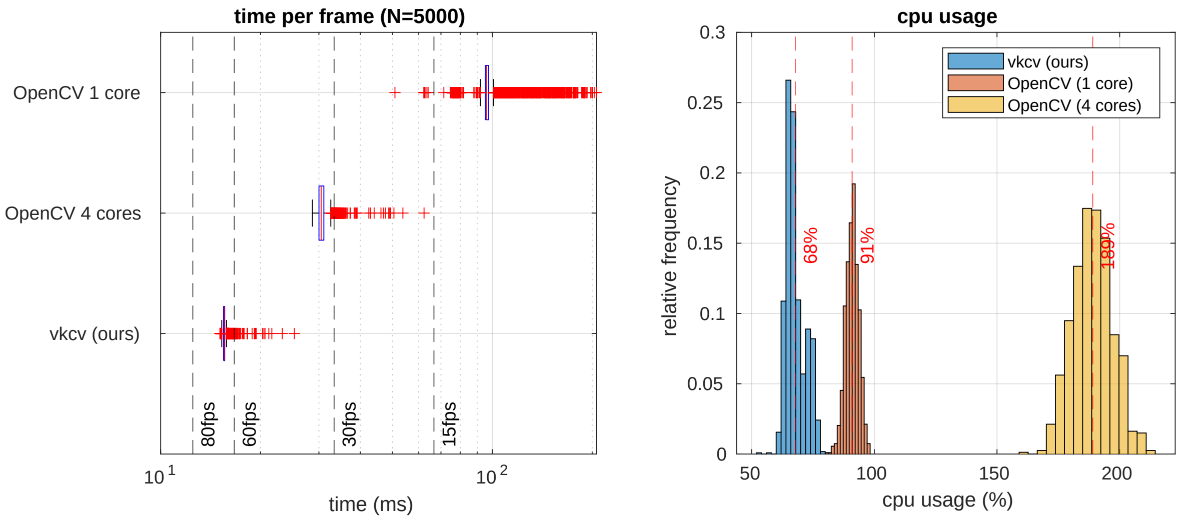


Figure 7.3: The GPU-accelerated object detection pipeline shows very consistent frame timings and thus meets firm real-time requirements (left). The pipeline outperforms the OpenCV implementation and only requires a single CPU core (right).

coordinates is given by

$$^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{v}} = {}^{\mathtt{W}}\mathbf{T}_{\mathtt{B}}{}^{\mathtt{B}}\mathbf{T}_{\mathtt{C}}{}^{\mathtt{C}}\bar{\mathbf{p}}_{\mathtt{v}}. \tag{7.5}$$

Even though the Pi's GPU is lacking power, it does easily manage 30 frames per second with a mean of 15.6 ms (or 64 fps) and a standard deviation of only 0.4 ms for the aforementioned object detection pipeline as can be seen in Fig. 7.3. During this test, the object detection pipeline is assigned to a single CPU core, leaving the other cores free to perform other critical tasks such as running the MPC or the autopilot. It should also be noted that even though strict real-time behavior cannot be guaranteed, the data suggest that the pipeline meets the criteria for firm real-time computing. The developed framework "vk_cv" is available online[1].

---

[1] vk_cv: `https://github.com/snt-arg/vk_cv`

7.3 Trajectory Optimization with Soft Perception, Tracking, and Grasping Constraints

The optimal trajectory taken by the AM is a compromise between several factors:

1. The energy spent along the path; the optimal path should be as cheap as possible, which in practice means minimizing the control effort.
2. The visual servoing mandates that the targeted object stays in the field of view of the camera such that it can be localized properly.
3. The grasping approach that respects the particularities of the UG requires the AM to stay at a distance z_{grab} from the floor such that the UAV does not collide with the payload during the approach. However, as soon as the UAV comes into close proximity to the target, it has to descend to engage the gripper with the payload.

Mathematically, the described compromise is achieved by solving the continuous-time optimization problem stated in (3.36). However, analytically solving (3.36) is (generally) not possible. Nonetheless, the solution can be approximated numerically, e.g., with a Newton solver.

For the problem visualized in Fig. 7.1, the following state vector is defined

$$\bar{\mathbf{x}} = \left[{}^{\mathtt{W}}\bar{\mathbf{p}}, {}^{\mathtt{B}}\bar{\mathbf{v}}\right] \in \mathbb{R}^{8}, \tag{7.6}$$

with

$$\begin{aligned} {}^{\mathtt{W}}\bar{\mathbf{p}} &= [x, y, z, \psi]\,, && (7.7a) \\ {}^{\mathtt{B}}\bar{\mathbf{v}} &= [v_x, v_y, v_z, v_\psi]\,, && (7.7b) \end{aligned}$$

where ${}^{\mathtt{W}}\bar{\mathbf{p}}$ is the position, ψ the yaw angle of the UAV, and ${}^{\mathtt{B}}\bar{\mathbf{v}}$ the velocity of the robot. Lastly, $\bar{\mathbf{z}}$ is defined as the vector of parameters which includes

$$\bar{\mathbf{z}} = \left({}^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{ref}}, v_0, v_N, {}^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{vis}}, c_{lock}, z_{safe}, \bar{\mathbf{o}}_{\mathtt{1}}, \ldots, \bar{\mathbf{o}}_{\mathtt{N_o}}\right) \in \mathbb{R}^{11+4N_o}, \tag{7.8}$$

with

$$
\begin{aligned}
{}^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{ref}} &= \left[x_{ref}, y_{ref}, z_{ref}, \psi_{ref}\right], && (7.9a)\\
{}^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{vis}} &= \left[u_{ref}, v_{ref}, w_{ref}\right], && (7.9b)\\
\bar{\mathbf{o}}_{\mathtt{i}} &= \left[o_{x,i}, o_{y,i}, o_{sx,i}, o_{sy,i}\right], && (7.9c)
\end{aligned}
$$

where ${}^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{ref}}$ is the desired position of the UAV, ${}^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{vis}}$ is the visual target location (point of interest), v_0 and v_N are the desired velocity at the start resp. end of the horizon, c_{lock} is the binary state indicating whether the visual lock has been acquired or not (i.e., whether a visual target is present), z_{safe} designates the safety altitude and $o_{(\cdot),i}$ represents the xy-centerpoint and minor and major axis of the i'th ellipsoidal obstacle. The separation of the visual target and the desired position in combination with c_{lock} gives a great deal of flexibility. By having ${}^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{ref}} \neq {}^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{vis}}$, the AM can be tasked to explore the vicinity of the object of interest while keeping it in the field of view of the sensor. At the same time, the sensor lock can be disabled by setting $c_{lock} = 0$ to prevent the MPC from tracking a potential target when it is not opportune to do so (e.g., when moving to the drop-off zone). For the actual grasping, however, ${}^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{ref}} \approx {}^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{vis}}$ as the visual target position generally corresponds to the payload position.

The state dynamics are defined as a UAV hover model:

$$
f\left(\bar{\mathbf{v}}, \bar{\mathbf{u}}\right) = \begin{cases} {}^{\mathtt{W}}\dot{\bar{\mathbf{p}}} & = \begin{pmatrix} {}^{\mathtt{W}}\mathbf{R}_{\mathtt{B}}\left(\psi\right) {}^{\mathtt{B}}\bar{\mathbf{v}}_{[\mathtt{xyz}]} \\ {}^{\mathtt{B}}\bar{\mathbf{v}}_{[\psi]} \end{pmatrix} \\ {}^{\mathtt{B}}\dot{\bar{\mathbf{v}}} & = \frac{1}{\bar{\tau}}\left(\bar{\mathbf{u}} \odot \bar{\mathbf{k}} - {}^{\mathtt{B}}\bar{\mathbf{v}}\right) \end{cases} \qquad (7.10)
$$

The gain and time constants of the system were determined from the simulated AM (*AscTec Firefly* depicted in Fig. 7.17) using the recorded data shown in Fig. 7.4 and classical

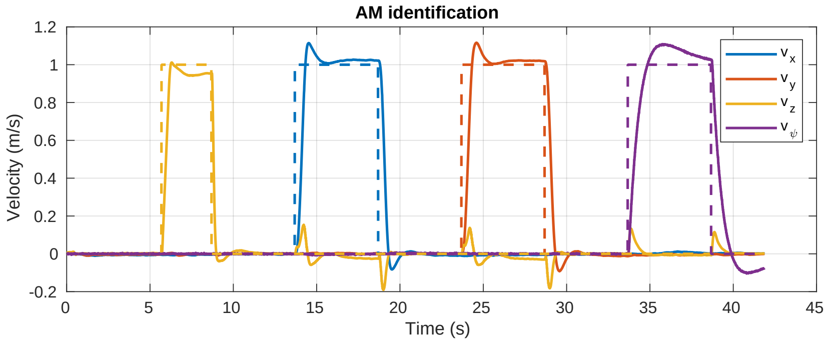


Figure 7.4: By commanding a sequence of unit steps, the AM is identified as a set of first-order systems (one for each direction).

direction	k_i	τ_i
x	1.0	0.51
y	1.0	0.51
z	1.0	0.40
ψ	1.0	0.54

Table 7.1: Identified time constants and gains of the simulated first-order AM system.

step response tangent method:

$$\tau_i = \frac{3}{2}\left(t_{y=63\%} - t_{y=28\%}\right), \tag{7.11a}$$

$$k_i = \frac{y(\infty)}{u(\infty)}. \tag{7.11b}$$

The results of the system identification are given in Table 7.1.

7.3.1 Discrete Time Formulation

The numeric solution of (3.36) leads to the following formulation of the discrete-time optimisation problem [156]:

$$\underset{\bar{\mathbf{x}},\bar{\mathbf{u}}}{\text{Minimize}} \sum_n J(\bar{\mathbf{x}}_n, \bar{\mathbf{u}}_n) + E(\bar{\mathbf{x}}_N) \tag{7.12a}$$

subject to:

$$\bar{\mathbf{x}}(0) = \bar{\mathbf{x}}_0 \tag{7.12b}$$

$$\bar{\mathbf{x}}_{n+1} = f\left(\bar{\mathbf{x}}_n, \bar{\mathbf{u}}_n, t\right) \tag{7.12c}$$

$$\bar{\mathbf{u}}_n \in U_n \tag{7.12d}$$

$$H_n\left(\bar{\mathbf{x}}_n, \bar{\mathbf{u}}_n\right) \leq 0 \tag{7.12e}$$

$$H_N\left(\bar{\mathbf{x}}_N\right) \leq 0 \tag{7.12f}$$

$$n \in \{1, \ldots, N\} \tag{7.12g}$$

where J and E are numerical approximations of the stage and terminal costs, respectively. H_n and H_N represent general inequality constraints.

Within this work, the Proximal Averaged Newton-type Method for Optimal Control (PANOC) algorithm [156] is used to solve the non-convex OCP that is part of the NMPC formulation. The PANOC algorithm solves the fixed-point equations with a fast converging line-search, single direct shooting method. The algorithm, being a first-order method, is particularly well suited for embedded onboard applications with limited memory and compute power and generally outperforms SQP methods.

Remark 2. *The solution of problem* (7.12) *is only optimal within the fixed-length moving horizon. Globally seen, the output (i.e., the trajectory) can be suboptimal, even with a risk of getting trapped in a local optimum.*

7.3.2 Cost Function: Perception

For a safe and reliable approach, the targeted object should preferably stay in the Field of View (FoV) of the AM. But imposing this as a hard constraint would in many cases lead to an infeasible problem. By softening those constraints, the NMPC gets the flexibility to achieve a solution that is a more favorable compromise between the objectives, e.g., saving a lot of energy at the expense of slightly violating the perception constraint.

The violation of a constraint is a binary state; either it is violated '1', or it is not '0'. If

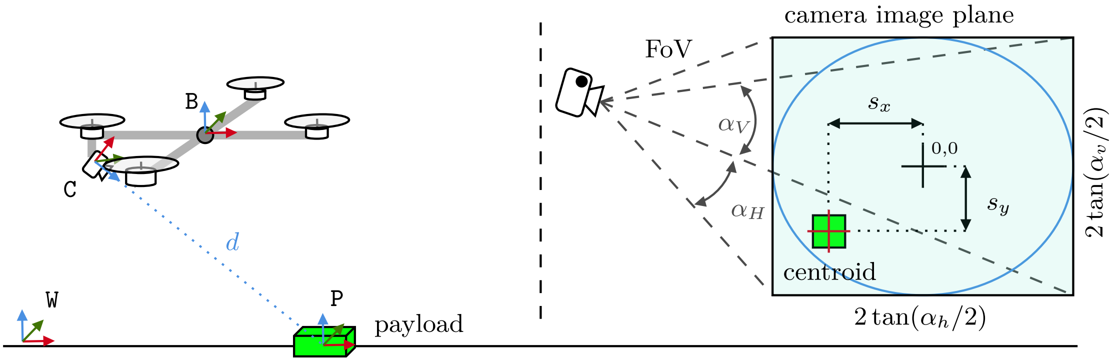


Figure 7.5: The UAV's camera detects the payload in green and assigns it a position within its frame of reference. The centroid of the payload is projected back into the camera's image plane, where it is assigned the normalized coordinates c_x and c_y.

it is, there is a non-zero cost associated with it, which incentivizes the optimizer to find a solution that does not violate the constraint. That binary behavior is akin to the Heaviside step function

$$H(x) = \begin{cases} 1, & x > 0 \\ 0, & \text{otherwise,} \end{cases} \tag{7.13}$$

Its discontinuous nature makes it unsuitable for numerical optimization. Better candidates are found in the family of logistic functions. Herein we use the sigmoid function instead of the Heaviside step function as it is continuously differentiable over its entire domain. The sigmoid function is defined as

$$\sigma\left(x, x_0, k_s, k_g\right) = \frac{k_g}{1 + \exp\left(-k_s\left(x - x_0\right)\right)}, \tag{7.14}$$

where k_s defines the steepness of the transition between 0 and k_g. The parameter x_0 defines the center point of the transition, i.e., $\sigma\left(x, x_0, k_s, k_g\right) = k_g/2$ for $x = x_0$. Based on the sigmoid function, the unit pulse function is defined as

$$\text{pulse}\left(x, k_s, k_g\right) = 4k_g\sigma\left(x, x_0 = 0, k_s, k_g = 1\right)\left(1 - \sigma\left(x, x_0 = 0, k_s, k_g = 1\right)\right), \tag{7.15}$$

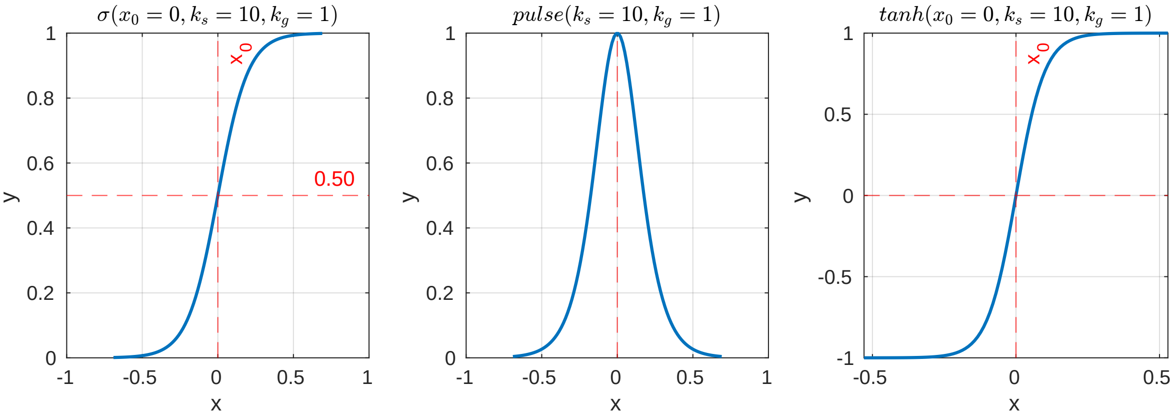


Figure 7.6: The steepness of the logistic functions σ and tanh as well as the pulse are adjustable via k_s.

and the tanh logistic function is given by

$$\tanh\left(x, x_0, k_s, k_g\right) = 2\sigma\left(2x, 2x_0, k_s, k_g\right) - 1. \tag{7.16}$$

The two logistic functions (7.14), (7.16) and the unit pulse are depicted in Fig. 7.6.

To satisfy the perception constraints, the MPC has to ensure that the centroid of the target object stays within the image plane of the camera as depicted in Fig. 7.5. The formulation herein does, however, not incentivize the MPC to keep the target in the center of the image plane as to not overconstrain the problem and to avoid unnecessary movements.

The object detection pipeline provides the position of the target object in world coordinates denoted as ${}^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{vis}}$, resp. in local camera coordinates as ${}^{\mathtt{C}}\bar{\mathbf{p}}_{\mathtt{vis}} = \left({}^{\mathtt{W}}\mathbf{T}_{\mathtt{B}}{}^{\mathtt{B}}\mathbf{T}_{\mathtt{C}}\right)^{-1}{}^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{vis}}$. Its coordinates $\bar{\mathbf{s}}$ in the virtual image plane are then obtained with the help of the following relationship:

$$\bar{\mathbf{s}}\left({}^{\mathtt{C}}\bar{\mathbf{p}}_{\mathtt{vis}}\right) = \begin{pmatrix} {}^{\mathtt{C}}\bar{\mathbf{p}}_{\mathtt{vis}[x]} / {}^{\mathtt{C}}\bar{\mathbf{p}}_{\mathtt{vis}[z]} \\ {}^{\mathtt{C}}\bar{\mathbf{p}}_{\mathtt{vis}[y]} / {}^{\mathtt{C}}\bar{\mathbf{p}}_{\mathtt{vis}[z]} \end{pmatrix}. \tag{7.17}$$

Eq. (7.17) has a singularity at ${}^{\mathtt{C}}\bar{\mathbf{p}}_{\mathtt{vis}[z]} = 0$. In practice, this is not an issue since the distance from the camera to the target can never be zero. In fact, the Realsense D435 has a minimal distance of 8 cm. Taking this into account, Eq. (7.17) is conditioned to eliminate the singularity at ${}^{\mathtt{C}}\bar{\mathbf{p}}_{\mathtt{vis}[z]} = 0$ as follows:

$$\bar{\mathbf{s}}\left({}^{\mathtt{C}}\bar{\mathbf{p}}_{\mathtt{vis}}\right) = \begin{pmatrix} {}^{\mathtt{C}}\bar{\mathbf{p}}_{\mathtt{vis}[x]}/z^* \\ {}^{\mathtt{C}}\bar{\mathbf{p}}_{\mathtt{vis}[y]}/z^* \end{pmatrix}, \tag{7.18a}$$

$$z^* = {}^{\mathtt{C}}\bar{\mathbf{p}}_{\mathtt{vis}[z]} + s_p, \tag{7.18b}$$

where $s_p = \operatorname{pulse}\left(x = {}^{\mathtt{C}}\bar{\mathbf{p}}_{\mathtt{vis}[z]}, x_0 = 0, k_s = 80, k_g\right)$ and therefore $\lim_{{}^{\mathtt{C}}\bar{\mathbf{p}}_{\mathtt{vis}[z]} \to \bar{\mathbf{0}}} z^* = k_g$, with k_g being a very small positive number.

The inequality constraints that keep $\bar{\mathbf{s}}$ in the FoV of the camera are thus $|\bar{\mathbf{s}}_{[x]}| < \tan\frac{\alpha_h}{2}$ and $|\bar{\mathbf{s}}_{[y]}| < \tan\frac{\alpha_v}{2}$, or more conveniently expressed as an inequality that restricts $\bar{\mathbf{s}}$ to lay within an ellipsoid centered around $\bar{\mathbf{0}}$ defined by its minor and major axis $h = \tan\alpha_h$ and $w = \tan\alpha_v$ such that

$$\frac{\bar{\mathbf{s}}_{[x]}^2}{(w/2)^2} + \frac{\bar{\mathbf{s}}_{[y]}^2}{(h/2)^2} < 1. \tag{7.19}$$

By making use of the sigmoid logistic function, (7.19) becomes

$$c_{p,1}\left({}^{\mathtt{C}}\bar{\mathbf{p}}_{\mathtt{vis}}\right) = \sigma\left(\frac{\bar{\mathbf{s}}_{[x]}^2}{(w/2)^2} + \frac{\bar{\mathbf{s}}_{[y]}^2}{(h/2)^2}, 1, k_{s1}, k_{g1}\right). \tag{7.20}$$

Eq. (7.20) has, however, two problems; *first*, it represents a double elliptical cone, i.e., the constraint is satisfied even if the target is behind the camera, *second*, $\nabla c_{p,1} \approx 0$ in regions where the constraint is violated, which makes recovering from constraint violations unlikely. The solution to the first problem is to add a constraint that guarantees ${}^{\mathtt{C}}\bar{\mathbf{p}}_{\mathtt{vis}[z]} > 0$ (target in front of the camera), resp. $c_{p,z} = \sigma\left(-{}^{\mathtt{C}}\bar{\mathbf{p}}_{\mathtt{vis}[z]}, 0, k_{s2}, k_{g2}\right)$, which extends (7.20) to become

$$c_{p,2}\left({}^{\mathtt{C}}\bar{\mathbf{p}}_{\mathtt{vis}}\right) = c_{p,z} + c_{p,1}\left(1 - c_{p,z}\right). \tag{7.21}$$

Lastly, a quadratic term and a bias term are added to the formulation such that the gradient

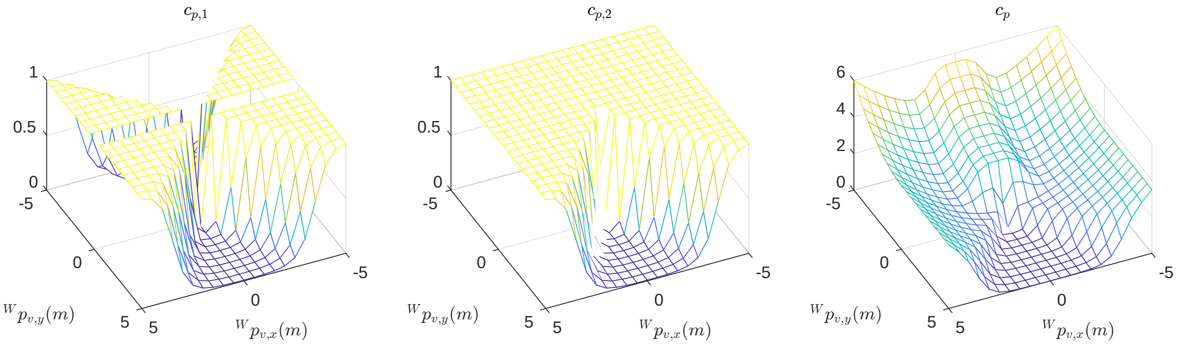


Figure 7.7: Perception constraints. (left) $c_{p,1}$ describes a double elliptical cone that is indifferent to whether the target is in front or behind the camera and also comes with a singularity at ${}^{\texttt{C}}\bar{\mathbf{p}}_{\texttt{vis}[z]} = 0$, (center) $c_{p,1}$ imposes that the target has to be in front of the camera, (right) a quadratic penalty and bias is added to c_p to help the optimizer to converge faster resp. at all.

of non-compliant regions fulfills $\nabla c_p > 0$:

$$c_p\left({}^{\texttt{C}}\bar{\mathbf{p}}_{\texttt{vis}}\right) = c_{p,2}\left(1 + k_g \left\| \begin{pmatrix} {}^{\texttt{C}}\bar{\mathbf{p}}_{\texttt{vis}[x]} \\ {}^{\texttt{C}}\bar{\mathbf{p}}_{\texttt{vis}[y]} \\ c_{p,z}\,{}^{\texttt{C}}\bar{\mathbf{p}}_{\texttt{vis}[z]} \end{pmatrix} \right\|^2 + c_{p,z} \cdot \text{pulse}\left(\left({}^{\texttt{C}}\bar{\mathbf{p}}_{\texttt{vis}[x]}\right)^2, 0, k_{s3}, k_{g3}\right)\right). \tag{7.22}$$

The quadratic term includes the condition $c_{p,z}$ such that the distance of the target from the camera in the positive z-direction is not penalized. The bias term in form of a pulse gives the UAV an incentive for the optimizer to accept the penalty of turning towards the target. An xy-slice through $z = 0$ of the constraint equations (7.20) to (7.22) is plotted in Fig. 7.7. The constraint function (7.22) is included as perception cost function

$$J_P\left(\bar{\mathbf{x}}\right) = k_P \cdot c_{lock} \cdot c_p \tag{7.23}$$

in (7.12), where k_P is a positive weight and $c_{lock} \in \{0, 1\}$ accommodates for the case where no target is detected, and the perception cost should consequently be ignored. This also accounts for cases where the UAV loses sight of the visual target, e.g., due to an obstacle blocking the line of sight with the target.

7.3.3 Cost Function: Tracking

Generally, the AM is commanded to reach some target position given by ${}^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{ref}}$ and ψ_{ref} with its actual position and orientation given by ${}^{\mathtt{W}}\bar{\mathbf{p}}$ and ψ. This section thus derives a cost function that encourages the UAV to approach the commanded location both for translation and rotation, starting with the latter.

The yaw angle error $\psi_{err} = \psi_{ref} - \psi$ is not a useful quantity to feed into the OCP due to the discontinuity at 0° resp. 360°. Instead, the orientation is encoded in the two-dimensional direction vector

$$\bar{\mathbf{q}}\left(\psi\right) = \begin{pmatrix} \cos\psi \\ \sin\psi \end{pmatrix}, \tag{7.24}$$

with the orientation error being

$$\bar{\mathbf{q}}_{\mathtt{err}}\left(\bar{\mathbf{x}}\right) = \frac{1}{2}\left(\bar{\mathbf{q}}_{\mathtt{ref}}\left(\psi_{ref}\right) - \bar{\mathbf{q}}\left(\psi\right)\right). \tag{7.25}$$

Likewise, the position error is defined as

$$\bar{\mathbf{p}}_{\mathtt{err}} = {}^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{ref}} - {}^{\mathtt{W}}\bar{\mathbf{p}}. \tag{7.26}$$

Lastly, to have control over the velocity at the start $v_0 > 0$ and the end $v_N > 0$ of the horizon, the following quantity is defined

$$v_{ref} = \left(1 - \alpha\right) v_0 + \alpha v_N, \tag{7.27a}$$

$$v_{err} = \left\|{}^{\mathtt{B}}\bar{\mathbf{v}}\right\| - v_{ref}, \tag{7.27b}$$

$$\alpha = n/N, \tag{7.27c}$$

where N is the number of states, and n is the current stage. The velocity at each stage of the moving horizon is thus changed linearly between v_0 at the beginning to v_N at the end to incentivize a gradual (linear) change of velocity and thus prevent an aggressive braking

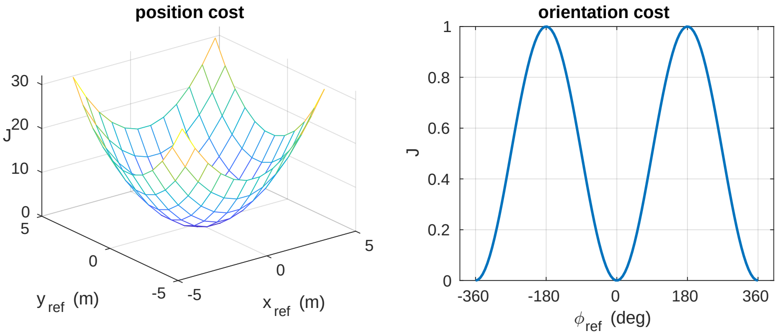


Figure 7.8: The tracking cost for position (left) and orientation (right).

maneuver when approaching the commanded location.

The cost function J_T associated with the position tracking is composed by (7.24), (7.27) and (7.26). To account for the varying objectives of that cost function, c_{lock} to switch between either tracking the reference angle or targeted object via the perception constraint. The cost equation to include in (7.12) is therefore

$$J_T\left(\bar{\mathbf{x}}\right) = k_{T,1}\left\|\bar{\mathbf{p}}_{\mathbf{err}}\right\|^2 + k_{T,2}v_{err}^2 + k_{T,3}\dot{\psi}^2 + \left(1 - c_{lock}\right)k_{T,4}\left\|\bar{\mathbf{q}}_{\mathbf{err}}\right\|^2, \tag{7.28}$$

where $k_{T,i}$ are positive weights. The cost plotted for each component is shown in Fig. 7.8.

7.3.4 Cost Function: Grasping

The grasping cost function takes the particularities of the gripper into account; it guarantees that the gripper stays at a safe altitude from the ground as long as it is not located above the target, and it only allows the AM to descent while it is in its close vicinity. At the same time, if the UAV for some reason (e.g., a wind gust) gets dislocated from the target's location, it is forced to regain in altitude. For that reason, two constraint functions are defined. The altitude penalty constraint is defined as

$$c_4 = 1 - \sigma\left(z, z_{min}, k_{s4}, k_{g4}\right), \tag{7.29}$$

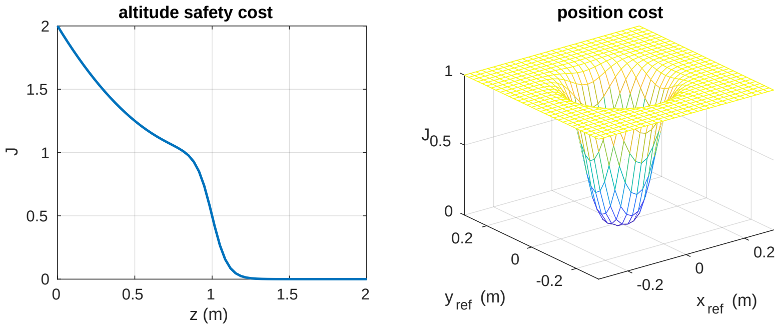


Figure 7.9: The minimum safety altitude $z_{min} = 1$ keeps the aircraft above z_{min} to prevent collisions (left). The grasping constraint associated with the error in xy-direction forms a cone, allowing the robot to descend at the target's location (right).

where z_{min} marks the safe minimal altitude the AM has to keep. However, as this would prevent the AM from descending to the payload, an additional constraint is needed based on the xy-position relative to the target, i.e.,

$$c_5 = 1 - \text{pulse}\left(\left\|{}^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{ref}\,[xy]} - {}^{\mathtt{W}}\bar{\mathbf{p}}_{[xy]}\right\|^2, k_{s5}, k_{g5}\right), \tag{7.30}$$

where k_s is chosen as a function of the radius r of the cone through which the UAV is funneled to the target. Herein $k_s\,(r = 0.1\,\text{m}) = 176.27$ was determined from solving the following equations:

$$x_{0.5}(k_s) \coloneqq 1 - \text{pulse}\,(x, k_s, k_g = 1) = 0.5 \tag{7.31a}$$

$$k_s(r) \coloneqq x_{0.5}(k_s) = r^2. \tag{7.31b}$$

The pulse function has the particularity of $\text{pulse}\,(x, k_s, k_g) = 0$ at $x = 0$, resulting in no residual cost at ${}^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{ref}\,xy} = {}^{\mathtt{W}}\bar{\mathbf{p}}_{xy}$. The total cost associated with the grasping constraints is thus defined as

$$J_G\,(\bar{\mathbf{x}}) = k_{G,1}c_4c_5\left(1 + k_{G,2}\,(z_{min} - z)^2\right) + k_{G,3}(1 - c_5)\left\|{}^{\mathtt{B}}\bar{\mathbf{v}}\right\|^2, \tag{7.32}$$

where $k_{G,i}$ are tuneable weights and $k_{G,3}(1 - c_5)\left\|{}^{\mathtt{B}}\bar{\mathbf{v}}\right\|^2$ penalizes for any velocity in close proximity to the target. The grasping costs are visualized in Fig. 7.9.

7.3.5 Cost Function: Repulsion

In many applications, it is useful to define areas that should be avoided during flight, e.g., pedestrians, lamp posts, walls or cars. Herein, it is assumed that those areas can be approximated with an ellipsoid. Furthermore, given the criticality of the disarmament task, it is also reasonable to assume that the targets are static, localized, and cannot be overflown (for safety resp. physical reasons), thus resulting in a 2D xy-problem.

To that account, an axis-aligned ellipsis is defined, located at the world position (o_x, o_y) with its minor and major axis defined by (o_{sx}, o_{sy}). A point (p_x, p_y) is inside the ellipsis if

$$\frac{(p_x - o_x)^2}{(o_{sx}/2)^2} + \frac{(p_y - o_y)^2}{(o_{sy}/2)^2} < 1. \tag{7.33}$$

Reformulating that expression using the sigmoid logistic function yields

$$c_R = 1 - \sigma\left(\frac{\Delta p_x^2}{(o_{sx}/2)^2} + \frac{\Delta p_y^2}{(o_{sy}/2)^2}, 1, k_{s6}, k_{g6}\right). \tag{7.34}$$

The repulsion cost function for a single area of repulsion i is thus

$$J_{R,i} = c_{R,i} \cdot \sum\left(\begin{pmatrix} o_{sx,i}^2 \\ o_{sy,i}^2 \end{pmatrix} - \begin{pmatrix} \Delta p_{x,i}^2 \\ \Delta p_{y,i}^2 \end{pmatrix}\right), \tag{7.35}$$

where a quadratic term was added to improve the gradient in the non-compliant region (see Fig. 7.10). The total repulsion cost of all areas is, therefore

$$J_R(\bar{\mathbf{x}}) = k_R \sum_{i}^{N_o} J_{R,i}. \tag{7.36}$$

In practice, an environment may contain hundreds of obstacles, and including each obstacle in (7.8) makes solving the OCP very computationally expensive. However, there are generally only a handful of obstacles near the UAV that must be considered. Therefore by simply feeding the momentarily closest obstacles into the OCP's parameter vector, a large environment with many obstacles can be considered with N_o still being reasonably small

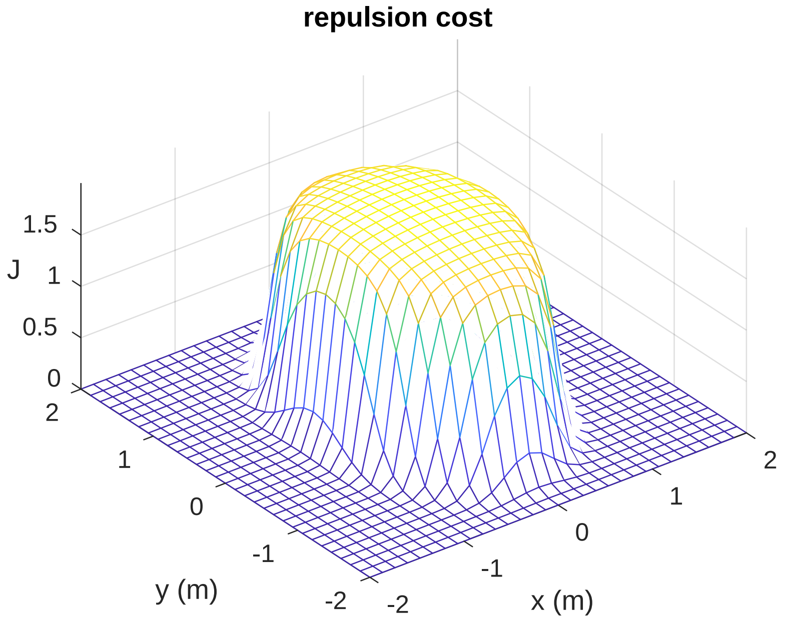


Figure 7.10: Repulsion cost due to an obstacle located at $(0, 0)$ m approximated by an ellipsis with minor and major axis of $(2, 3)$ m.

(here, $N_o = 2$). Furthermore, the AM's body size can be accounted for by simply expanding the ellipsis of the obstacle by the radius r_B of the body, i.e., $(\hat{o}_{sx}, \hat{o}_{sy}) = (o_{sx} + r_B, o_{sy} + r_B)$. Finally, an obstacle with a non-ellipsoidal shape can be approximated by placing several ellipses within its contour.

7.3.6 Numerical Validation

The objective of this section is to verify that the developed cost functions yield the intended behavior. To that aim the developed cost functions are included in (7.12) as the sum of the individual cost functions for perception (7.23), position tracking (7.28), grasping (7.32), repulsion (7.36), and an additional term J_U that penalizes for energy-inefficient trajectories:

$$J(\bar{\mathbf{x}}, \bar{\mathbf{u}}) = J_P(\bar{\mathbf{x}}) + J_T(\bar{\mathbf{x}}) + J_G(\bar{\mathbf{x}}) + J_R(\bar{\mathbf{x}}) + J_U(\bar{\mathbf{u}}) . \tag{7.37}$$

The quadratic control effort penalty is defined as

$$J_U\left(\bar{\mathbf{u}}\right) = \bar{\mathbf{u}}^T \mathbf{Q}_u \bar{\mathbf{u}}. \tag{7.38}$$

A potential issue can arise as a consequence of using soft constraints on the objectives (7.23), (7.32) and (7.36) where any non-bounded cost in (7.37) can dominate all other objectives, which, ultimately, leads to an inopportune violation of the soft constraints. Herein, the position tracking cost (7.28) represents such a non-bounded cost. Therefore, our implementation includes a carrot-chasing law for position tracking that feeds a virtual target location ${}^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{ref,v}}$ to the MPC that cannot exceed a certain distance from the UAV's current position. This effectively means that (7.28) is bounded, regardless of the actual distance to the target. The virtual target displacement $\bar{\mathbf{w}}$ is defined as follows

$$\bar{\mathbf{\Delta}} = {}^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{ref,user}} - {}^{\mathtt{W}}\bar{\mathbf{p}}, \tag{7.39a}$$

$$\left\|\bar{\mathbf{\Delta}}\right\| \in \left[0; w_{max}\right], \tag{7.39b}$$

$$w_{max} = N \cdot t_s \cdot v_{max}, \tag{7.39c}$$

The maximum distance that can be covered during the moving horizon at speed v_{max} is designated by w_{max}. The virtual target point ${}^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{ref,v}}$ is fed into the MPC as ${}^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{ref}} = {}^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{ref,v}}$ with ${}^{\mathtt{W}}\bar{\mathbf{p}}_{\mathtt{ref,v}} = {}^{\mathtt{W}}\bar{\mathbf{p}} + \bar{\mathbf{w}}$.

Next, the reference velocities v_0, v_N are defined as follows

$$v_0 = \alpha \cdot v_{max}, \tag{7.40}$$

$$v_N = \begin{cases} 0 & \text{if} \left\|\bar{\mathbf{\Delta}}\right\| < w_{max}, \\ v_{max} & \text{otherwise}, \end{cases} \tag{7.41}$$

$$\alpha = \left\|\bar{\mathbf{w}}\right\| / w_{max} \in \left[0; 1\right]. \tag{7.42}$$

The UAV is thus encouraged to travel at maximum speed if the virtual target is located at a distance $\left\|\bar{\mathbf{\Delta}}\right\| \geq w_{max}$, i.e., the target cannot be reached within the horizon. Once the target gets closer, i.e., $\left\|\bar{\mathbf{\Delta}}\right\| < w_{max}$, the UAV has to slow down and is incentivized to reach zero velocity at the end of the moving horizon. The initial reference velocity v_0 is gradually reduced as the robot gets closer to the target point.

The developed MPC is validated numerically in three different scenarios, testing perception, tracking, grasping, and obstacle avoidance.

Methodology The OCP described in (7.12) is implemented in the OpEn [75] framework, which handles code generation and auto-differentiation using CasADi [76] and provides the fast PANOC solver for embedded application.

The authors of [157] state that the prediction horizon must be chosen carefully such that a change to the input $\bar{\mathbf{u}}$ has a chance to realize an effect on the system states. Herein, the duration of the moving horizon is therefore based on the slowest subsystem of (7.10), which has the time constant $\tau = 0.54\,\mathrm{s}$ (see Table 7.1). Since a first order system reaches steady-state after 5τ, and given an adequate sampling interval of $T_s = 0.1\,\mathrm{s}$, the number of stages can be calculated as

$$N = \frac{5\tau}{T_s} = 26. \tag{7.43}$$

The simulation performed inside MATLAB applies the first set of inputs from the MPC to the model defined in (7.10) and updates the system states using the forward Euler integration scheme. The empirically tuned weights of the MPC used throughout the simulated scenarios are as indicated in Table 7.2.

Cost function	Tuneable constants	Assigned values
perception (7.23)	$(k_{s1}, k_{g1}, k_{s2}, k_{g2}, k_{s3}, k_{g3})$	$(1, 5, 0, 30, 1, 20)$
tracking (7.28)	$(k_{T,1}, k_{T,2}, k_{T,3}, k_{T,4})$	$(10, 20, 10, 100)$
grasping (7.32)	$(k_{G,1}, k_{G,2}, k_{G,3}, k_{s4}, k_{g4}, k_{s5}, k_{g5})$	$(50, 1, 80, 20, 1, 0, 176.2747)$
obstacle (7.36)	(k_R, k_{s6}, k_{g6})	$(1, 1, 20)$
control effort (7.38)	$\mathbf{Q}_u$	$(10, 10, 10, 10)$

Table 7.2: The empirically tuned gains of the MPC.

Scenario 1 (perception, inside FoV) In this scenario, the UAV starts with its initial state set to $(2\,\mathrm{m}, -2\,\mathrm{m}, 1\,\mathrm{m}, 225°)$ and with the visual target located at $(0, 0, 0)\,\mathrm{m}$. The commanded trajectory is a circle in xy-plane centered around the origin with a radius of $2\,\mathrm{m}$. The robot starts with the visual target in view, but due to the circular trajectory, it has to continuously adjust its heading to keep up with the target. The soft constraint concerning the vertical location of the target in the virtual image plane allows the UAV to keep the commanded altitude even though the target is not at the center of the frame. Furthermore, an obstacle was added defined as $\bar{\mathbf{o}}_1 = (0, 2, 2, 1)\,\mathrm{m}$. Due to its zone of repulsion, the robot leaves its circular path such that it avoids the obstacle. The 3D path taken by the robot and the corresponding graphs are shown in Fig. 7.11.

Scenario 2 (perception, outside FoV) This scenario is identical to scenario 1, except that the UAV start with the camera pointing away from the target, i.e., with starting position $(2\,\mathrm{m}, -2\,\mathrm{m}, 1\,\mathrm{m}, 45°)$, which represents a worst-case-scenario. Without the bias term in the perception cost, the optimization problem tends to converge in the sense that the camera keeps pointing away from the visual target. But with the formulation in (7.23), as shown in Fig. 7.12, the robot does manage to orient itself correctly to the target.

Scenario 3 (grasping approach) In this scenario, the UAV starts with its initial state set to $(2\,\mathrm{m}, 0\,\mathrm{m}, 2\,\mathrm{m}, -90°)$ and has the positional and visual target set to $(-1.5\,\mathrm{m}, 0\,\mathrm{m}, 0\,\mathrm{m}, 45°)$. Since the target is in the FoV, the MPC ignores the desired orientation and instead assures that the target stays in the FoV. A safety distance of $0.5\,\mathrm{m}$ was defined, which keeps the robot at a safe distance from the ground and only allows it to descend when close to the target (in xy-plane). At the same time, the perception constraint is dropped in close proximity to the target so as not to over-constrain the problem. The scenario is successfully executed as depicted in Fig. 7.13.

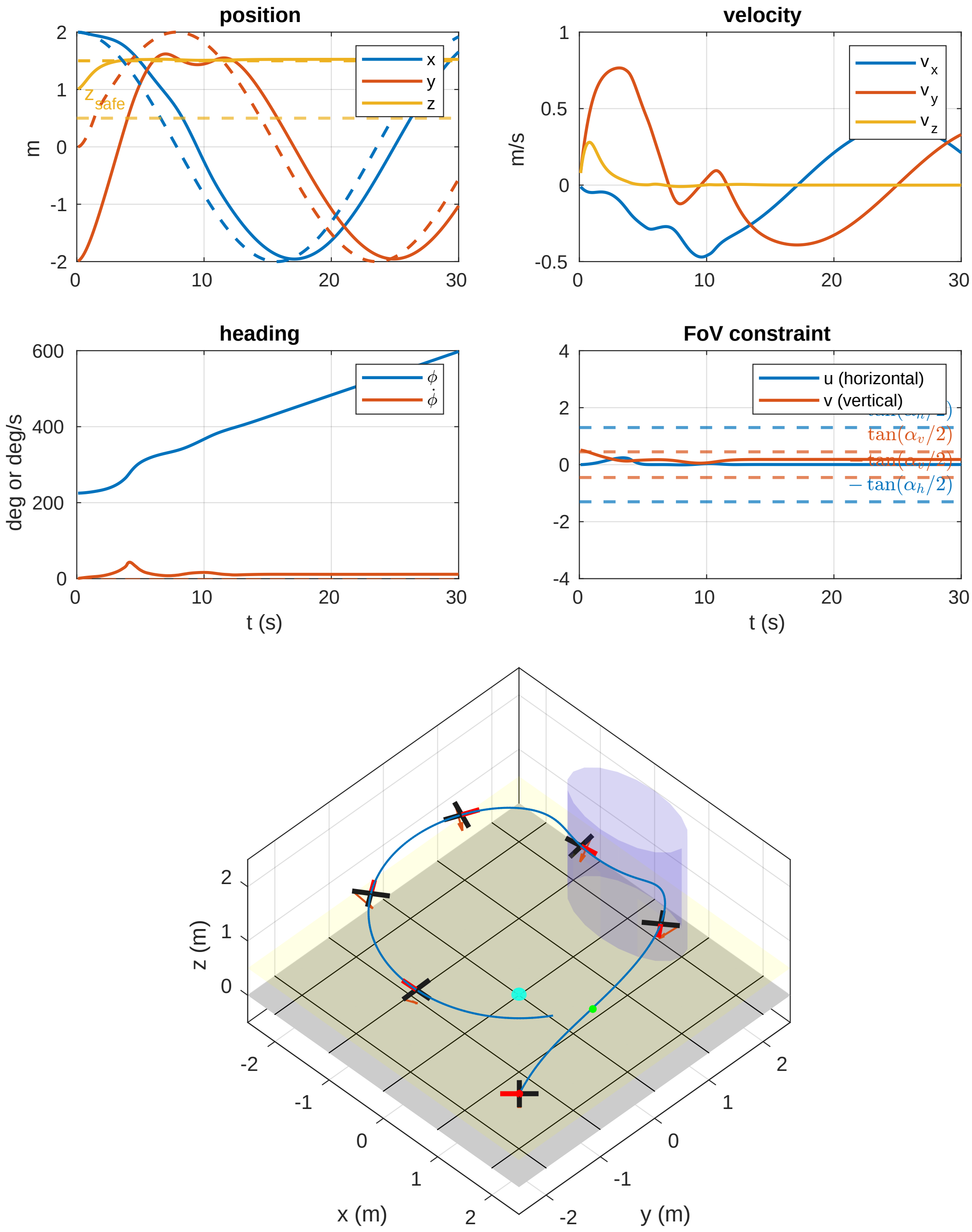


Figure 7.11: Scenario 1: The UAV starts with the visual target (teal) in the FoV of the camera (red arrow). An obstacle (blue ellipsis) was placed in the robot's path. The virtual target point for the last iteration is shown in green. The +x direction of the drone is represented by the red segment.

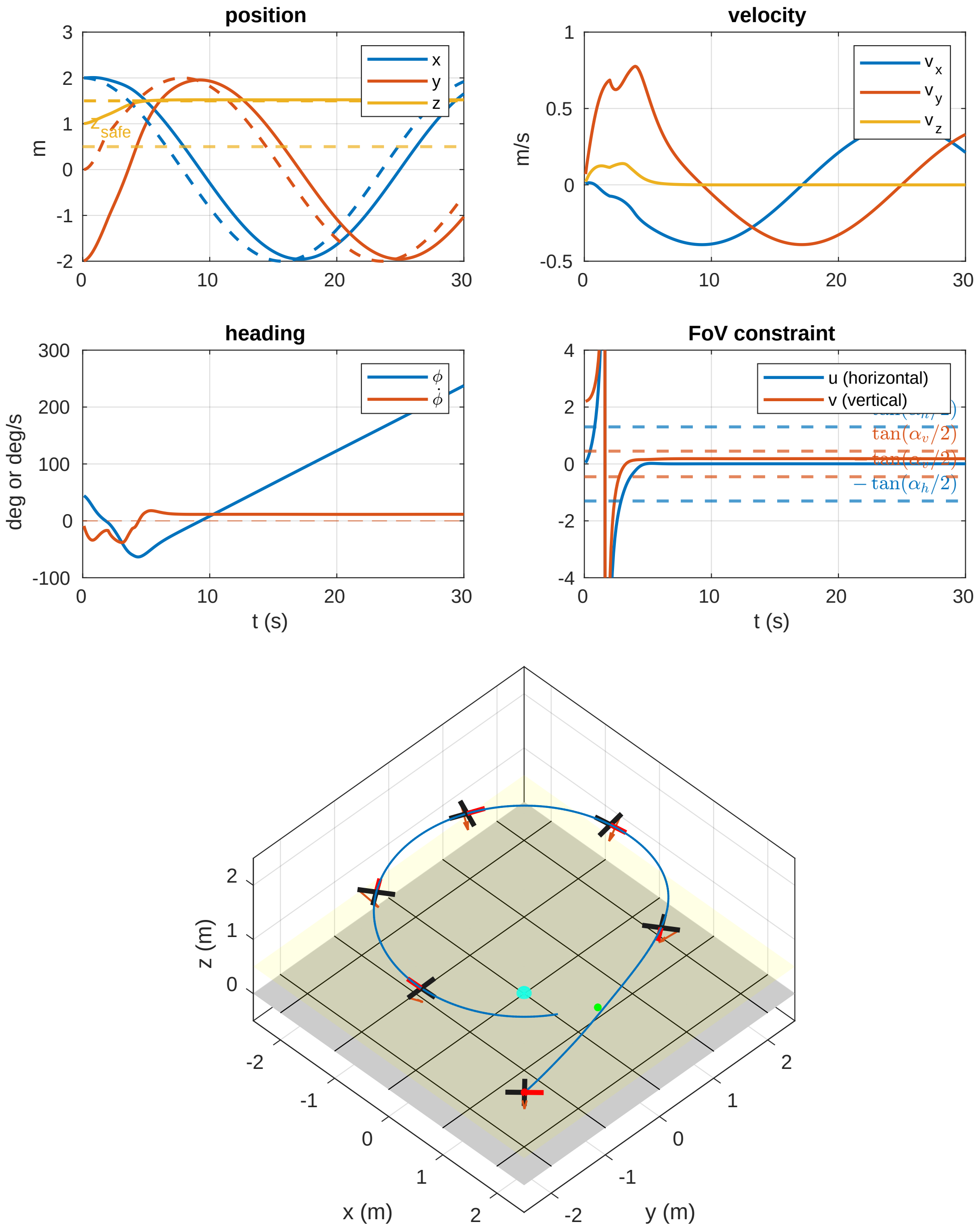


Figure 7.12: Scenario 2: The UAV starts with the visual target outside of the FoV of the camera and consequently has to perform almost a 180° turn to regain vision on the target, which represents a worst-case scenario to the optimizer.

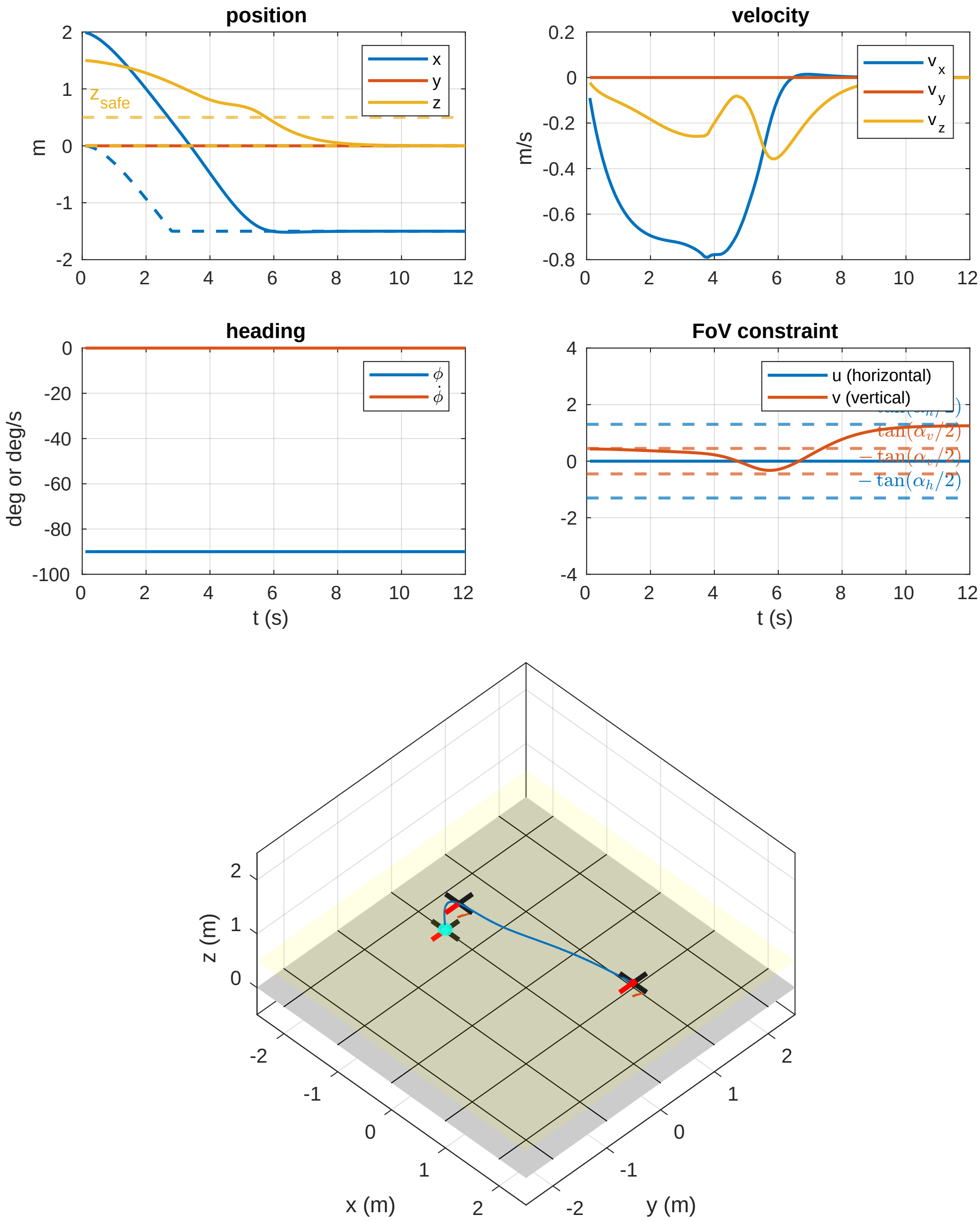


Figure 7.13: Scenario 3: The UAV approaches the target as if it would pick it up. It is not allowed to drop below 1 m until it is relatively close to the target in xy-plane and only then is the constraint lifted. At the same time, the perception constraint is faded out such that it does not over-constrain the problem, leading to erratic movements.

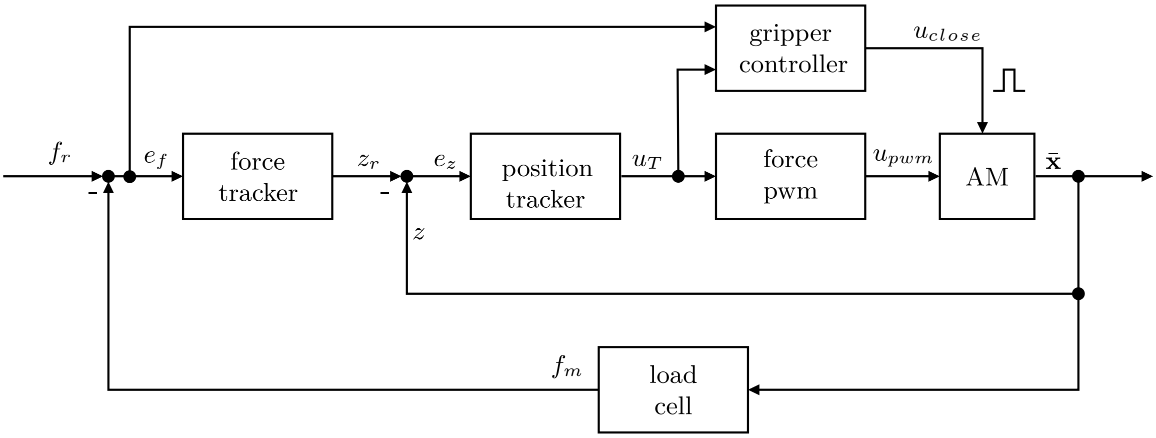


Figure 7.14: The force control architecture consists of a tracking controller for the AM's position (inner loop) and a force tracking controller (outer loop) commanding a reference position dependant on the force error e_f. The gripper controller closes the gripper once steady state is reached.

7.4 Force Control

For optimal operation, the UG requires the activation force to be controlled (i.e., kept constant) over the grasping interval. More precisely, the force should not exceed a certain level as it may cause damage to the gripper, and it must also not drop below a certain threshold as this servery degrades the grasping performance.

The fundamental problem is thus to control the contact force, as it was shown in [158] and [27] for bridge inspection and in [159] in the context of aerial writing. The noticeable difference here is the strong nonlinearity of the elastic element and the shrinkage of the membrane, which consequently changes the stiffness of the system during operation and may also cause the complete loss of contact. The cascading force control approach followed herein is inspired by [160], and [161]. Since the contact force measured by the load-cell and the position of the UAV are linked, the contact force tracking problem can be seen as a position tracking problem where the reference position is a function of the force tracking error. Motivated by this statement, the cascading control architecture shown in Fig. 7.14 is developed.

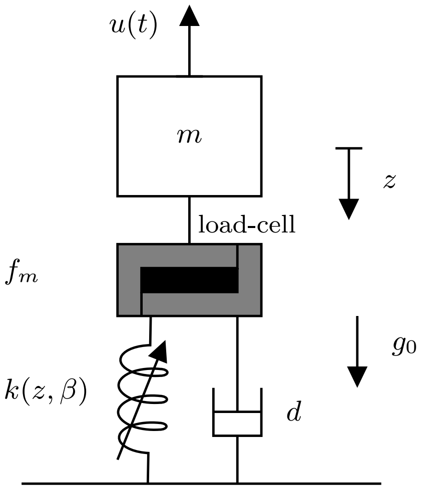


Figure 7.15: The UG is modeled as a mass-spring-damper system, where the spring is highly non-linear.

7.4.1 Problem Formulation

The physical properties and grasping performance of the UG were rigorously analyzed in Section 6.4 with the main conclusion being that the system is highly non-linear and state dependant. In particular, the stiffness of the elastic element changes from very soft to rigid-like.

The homologous model of the gripper is as shown in Fig. 7.15 consisting of a mass-spring-damper system, where the stiffness k is highly non-linear, compression-only and dependant on the gripper's state β. The damping d is unknown. The mass m is the total mass of the AM with $u\left(t\right)$ being the cumulative thrust applied by the aircraft's propulsion system. The dynamics of the spring-mass-damper system are according to the following set of equations

$$\ddot{z} = \frac{1}{m}\left[-f_{kd}\left(z,\dot{z}\right)|_{z>0} + mg_0 - u\left(t\right)\right], \tag{7.44a}$$

$$z\left(0\right) = z_0, \tag{7.44b}$$

$$\dot{z}\left(0\right) = v_0 \in \left]0; v_{max}\right], v_{max} > 0, \tag{7.44c}$$

$$f_{kd}\left(z,\dot{z}\right) = f_k\left(z\right) + f_d\left(\dot{z}\right), \tag{7.44d}$$

where $u(t)$ is the input (applied force) to the system, the states $\dot{z}$ and z can be measured, $f_d(z, \dot{z})$ and $f_k(z)$ are smooth, positive definite functions. It is assumed that the system has an initial downward velocity $\dot{x} = v_0$. The initial position is defined as $z(0) = 0$, which marks the position at which the gripper touches the ground. Assuming ground contact, the load-cell measures the contact force F_m at 100 Hz.

$$F_m = f_{kd}(z, v_z) \tag{7.45}$$

7.4.2 Force Tracking

The activation force is indirectly tracked by controlling the position (altitude) of the drone. The control solution thus consists of a *force tracker* and a *position tracker* (Fig. 7.14).

Position Tracker The position tracker is realized as a SMC based on the discussion in Section 3.3.1. The system can be seen as uni-axial (altitude only) since the UAV will be hovering (stationary in xy-direction) during the grasp. The non-linear uni-axial system is considered as

$$\dot{v}_z = f(\bar{\mathbf{x}}) + g(\bar{\mathbf{x}})\, u(t) + \vartheta(t) \tag{7.46}$$

where $f(\cdot)$ and $g(\cdot)$ are non-linear smooth functions and $\bar{\mathbf{x}}$ being the state vector defined in (7.6). The unknown disturbance is assumed to be bounded $|\varepsilon| < D$. The sliding surface is defined as

$$s = \dot{e_z} - ce_z, \tag{7.47a}$$

$$\dot{s} = \ddot{e_z} - c\dot{e_z}, \tag{7.47b}$$

where $c > 0$ is a design parameter and $e = z_r - z$ is the tracking error. Assuming a constant rate reaching-law

$$\dot{s} = -\eta \cdot \operatorname{sign}(s), \, \eta > 0 \tag{7.48}$$

the input can be defined as

$$u_T = \frac{1}{g}\left(-\eta \cdot \operatorname{sign}(s) - \ddot{z}_r + f + c\dot{e}_z\right) \tag{7.49}$$

which is stabilizes the system under the condition that $\eta > D$, as discussed in Section 3.3.1.

To reduce chatter, the sign function is replaced by the saturation function (3.34):

$$u_T = \frac{1}{g}\left(-\eta \cdot \operatorname{sat}(s) - \ddot{z}_r + f + c\dot{e}_z\right). \tag{7.50}$$

Assuming that the UAV is in hover position, the uni-axial dynamics can be stated as

$$f_z = \begin{cases} \dot{v}_z & = f(\bar{\mathbf{x}}) + gu(t) + \vartheta(t), \\ \dot{z} & = v_z, \end{cases} \tag{7.51}$$

with

$$f(\bar{\mathbf{x}}) = \frac{1}{m}\left(f_m - g_0\right), \tag{7.52a}$$

$$g = \frac{1}{m}. \tag{7.52b}$$

Considering the sliding surface as defined in (7.47a) and the reaching law in (7.48), the control output $u_T(t)$ of the tracker is defined as

$$u_T(t) = -f_m + m\left(g_0 - c\dot{e}_z - \eta \cdot \operatorname{sat}(s)\right) \tag{7.53}$$

Force Tracker The force applied by the UG to the payload, as it is measured by the sensor, is directly dependent on the system states x and v. Therefore, the force can be regulated by commanding and tracking a specific position (altitude) z_r. To that goal, the force tracker is defined as

$$z_r = k_p e_f + w_{e_f} \tag{7.54a}$$

$$\dot{w}_{e_f} = k_i e_f \tag{7.54b}$$

where k_p and k_i are positive design variables.

The gripper is closed once the system has reached steady-state, i.e., $|\dot{u}| < c_u$ and $|e_f| < c_e$ with c_u, c_x being small positive values. Closing the gripper triggers its state transition from soft to rigid. The complete control architecture is depicted in Fig. 7.14.

The simulated system response for the closed-loop system with an initial position $x(0) = 0\,\mathrm{m}$ and velocity of $v(0) = 0.1\,\mathrm{m\,s^{-1}}$ is shown in Fig. 7.16. The design variables were experimentally chosen as stated in Table 7.3.

Equation	Tuneable constants	Assigned values
SMC (7.53)	(c, η)	$(3, 4)$
Force Tracker (7.54)	(k_p, k_i)	$(0.03, 0.05)$
sat (3.34)	ϵ	0.8

Table 7.3: The empirically tuned constants of the SMC.

Remark 3. *On the UAV the altitude z is always relative to the position the gripper first entered in contact with the payload, i.e., the point where $f_m > \delta$ and $\dot{f}_m > 0$ is measured (δ being a small threshold value). The initial position is thus, by definition, always close to zero. Contrary to that, there will always be a certain initial velocity as commanded by the MPC.*

The gripper controller sends the close command around the $t = 4\,\mathrm{s}$ mark as the closed system reaches steady-state. This triggers the UG's state transition from soft to rigid during which the membrane shrinks, and consequently, the UAV has to descend to compensate for the diminishing contact force.

Normalzied Thrust Mapping The force controller outputs a commanded force in Newton, whilst the autopilot only accepts a normalized thrust between 0 and 1. A mapping is thus required that establishes the relation between force u_T and normalized thrust u_{pwm}.

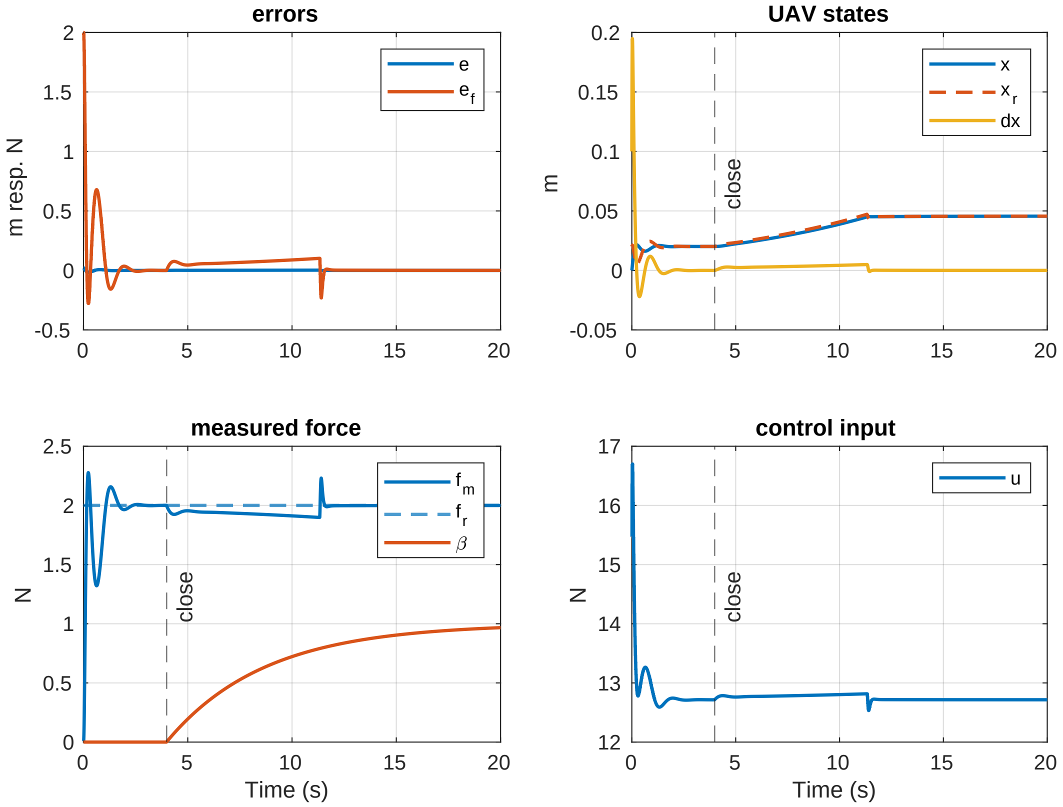


Figure 7.16: Simulated closed-loop response of the force controller. The gripper is closed once the system reaches steady-state. The chattering on the input $u(t)$ is effectively mitigated by the saturation function.

Given that the generated thrust $u_{T,p}$ by a propeller grows quadratically with its angular velocity v [114] and assuming a linear relationship between the Electronic Speed Controller (ESC)'s control signal u_{pwm} (PWM signal) and the resulting angular velocity of the propeller, the relationship between the two quantities is stated as

$$u_{T,p} = \gamma \left(u_{pwm} \cdot v_{max}\right)^2, \tag{7.55}$$

where γ is the motor constant, $u_{pwm} \in [0, 1]$ is the normalized control signal and v_{max} is the maximum angular velocity that can be reached by the propeller. The PX4 autopilot generally linearizes the above relationship. Therefore, by taking into account that the UAV is equipped with $N_p = 6$ propellers, the total thrust generated by the platform as a response

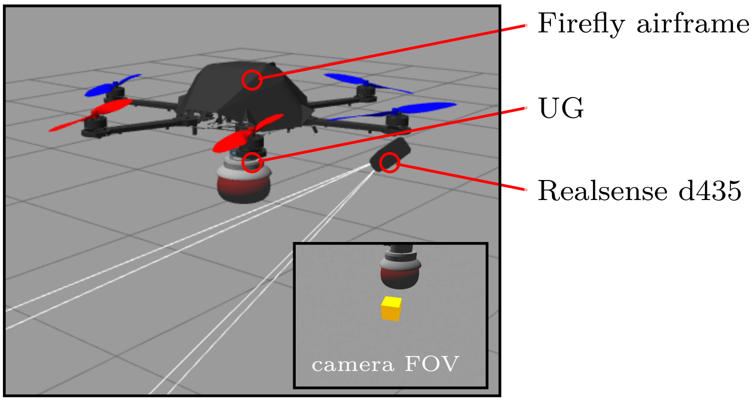


Figure 7.17: The AM setup inside the Gazebo simulation consists of a hexacopter carrying the simulated UG and a stereo-vision camera (depth camera) for the localization of the payload in three dimensions.

to a nominal normalized thrust $u_{pwm,PX4}$ is given by

$$u_T = N_p \cdot \hat{\gamma} \cdot u_{pwm,PX4}. \tag{7.56}$$

The lumped constant $\hat{\gamma} = \gamma \cdot v_{max}^2 = 6.003\,\mathrm{N\,pwm^{-1}}$ was calculated based on the model properties of the UAV (*AscTec Firefly*) in the Gazebo simulator. In practice, this constant can easily be identified experimentally on a thrust test stand.

7.5 Autonomous Grasping in Simulation

In this section, the components developed throughout this work are applied in a model scenario that has the disposal of an IED as its goal. The scenario is set up in Gazebo, a robotics simulator that performs a realistic simulation of the flight dynamics and sensors as well as the autopilot of a UAV inside a virtual environment.

7.5.1 Methodology and System Architecture

The AM used throughout this work is depicted in Fig. 7.17 and consists of a hexacopter carrying the simulated UG as developed in Section 6.4. Furthermore, it features a stereo-

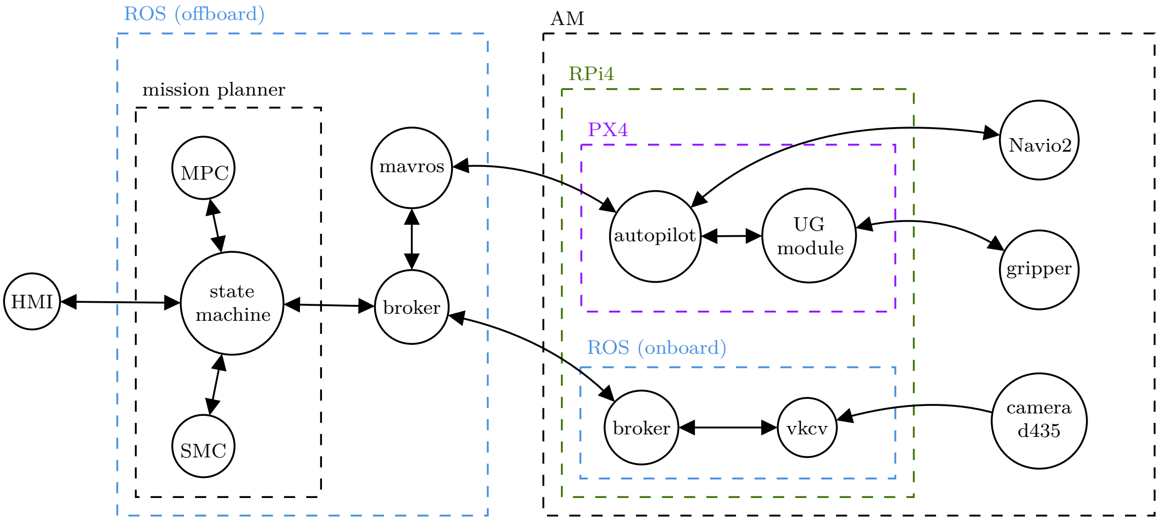


Figure 7.18: The layered system architecture of the AM and the base station.

vision camera that localizes the payload using the object detector described in Section 7.2. A modified PX4 [162] serves as the autopilot for the drone and assumes communication with the UG via a custom interface module. The custom module further exposes the information regarding the state of the gripper to the outside world via MAVlink [163] and consequently to ROS [164] via a custom MAVROS plugin. Furthermore, the commands from ROS are forwarded to the gripper, making it fully integrated into the ROS ecosystem despite being only visible to the autopilot. The complete, layered system architecture is depicted Fig. 7.18.

For safety reasons, the MPC, the force controller, and the state machine handling the disposal mission are implemented in a ROS node called 'mission planner' (instead of being part of the autopilot). The mission planner pulls in all relevant information from the ROS world and provides the autopilot and gripper with the relevant commands based on the state of the mission. The intention is that the operator provides a rough description of the scenario (environment, obstacles, location of the payload), and then, based on that, the mission planner coordinates all of the developed components to ensure the successful execution of the mission. The payload's position can be just a rough estimate as it gets automatically updated once it enters the field of view of the camera. In addition to the mission planner, a 'vkcv' ROS node handles the object detection based on the virtual color and depth images and exposes the detected 3d coordinates of the payload to the mission planner.

Remark 4. *Generally, it is advisable to keep the controller as close as possible to the sensors and actuators. Herein this would be within the PX4 autopilot, where it exhibits the lowest latency. However, for safety and regulatory reasons (tampering with the safety-critical firmware), it was implemented as a ROS node at the expense of additional latency.*

Since there are no moving obstacles, the MPC (Section 7.3) is made aware of potential obstacles by manually specifying their bounds prior to launching the simulation. For real applications, online visual SLAM could be used to create a map of the environment in real-time, e.g., [165], and consequently generating and feeding a list of ellipsoids representing obstacles to the MPC. As mentioned in Section 7.3.3, the list of obstacles is sorted by their distance to the UAV (shortest distance to ellipsoid), and only the closest three obstacles are fed into the MPC's parameter vector (7.8).

Scenario An IED has been identified in a parking garage close to a car. The goal of this simulation is to pick up the approximately localized ($\pm 0.5\,\mathrm{m}$) IED and transport it to a safe disposal site. The AM uses its onboard camera to further refine the localization of the IED during the approach. The MPC handles the trajectory generation throughout the simulation and thus guides the UAV to the IED respecting the FoV of the camera and vertical descend imposed by the gripper, whilst avoiding any obstacles such as the pillars, walls, and cars in the parking garage. Having reached the IED, the control scheme is changed to force control, where a nominal activation force of $2\,\mathrm{N}$ is tracked. After reaching steady-state on the exerted force, the simulated UG is closed, and the IED is securely grasped. The MPC now takes the AM to the disposal site where the IED is released. Finally, the AM returns to its starting position, where it lands again on the UG. The scenario is visualized in Fig. 7.19.

Results The path taken by the UAV is visualized in Fig. 7.20 alongside the obstacles known to the MPC. The UAV successfully avoids all obstacles on its way to the explosive and graciously approaches the IED such that it can be grasped by the gripper. The robot

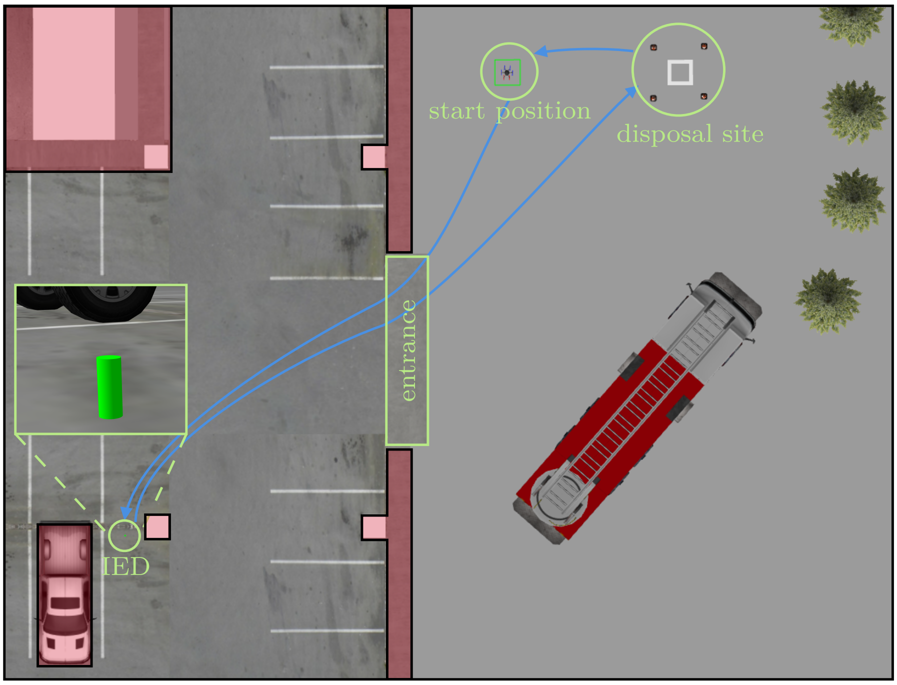


Figure 7.19: The parking lot scenario in Gazebo. An IED was identified close to the pickup truck. A drone equipped with the UG is located at the starting position. It enters the parking via the entrance avoiding the walls and pillars. The AM then transports the cylindrical IED to the safe disposal site.

then moves to the disposal area, again avoiding all obstacles along its path. The MPC's commanded velocities (x, y, z, and yaw) at 100 Hz that are tracked via a set of cascading PID controllers for the roll, pitch and yaw axis resp. the velocities in x and y. The aforementioned SMC tracks the commanded velocity in z. The commanded and actual velocities are visualized in Fig. 7.21. It can be seen that the commanded velocity z-direction approaches a very gentle $-0.06\,\mathrm{m\,s^{-1}}$, which allows for a relatively smooth transition to the force-control scheme which is shown in detail in Fig. 7.22 between the 35 s and 43 s mark. A nominal force of 2 N is tracked by the SMC during the grasping interval. Once the activation force reaches steady-state, the gripper is closed at the 37 s mark. After having acquired a secure grasp, the UAV takes off and, consequently, the gripper's load-cell registers the weight of the payload (200 g), which is fed back to the SMC, which is thus aware of the changed mass of the system.

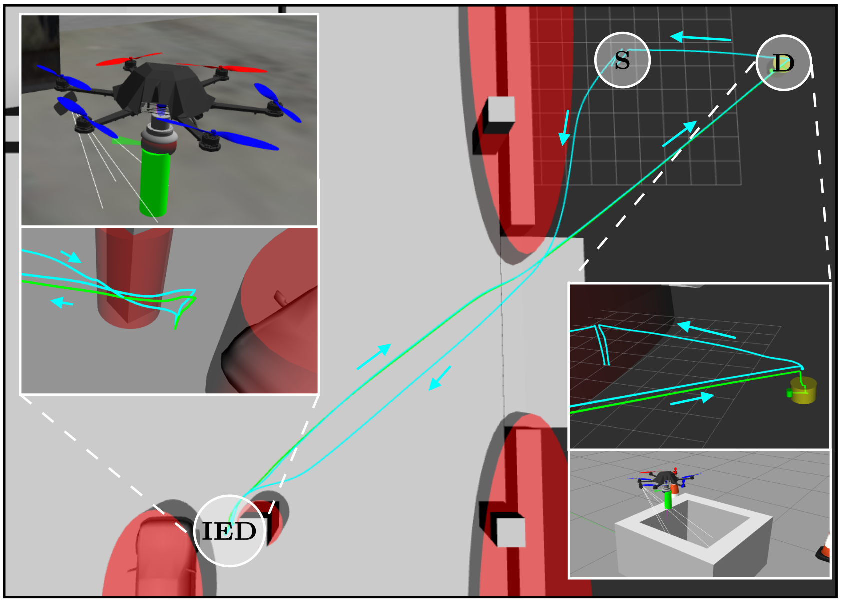


Figure 7.20: The obstacles (walls, pillar, car) were modeled as ellipsis. The path taken by the UAV is traced in blue, and the path taken by the IED is shown in green. The IED is picked up and placed inside in the disposal area 'D' (yellow) before the UAV returns and lands at the start position 'S'.

7.6 Conclusion

In this chapter, an autonomous aerial grasping concept and architecture specifically tailored to the UG was presented. The concept involved the development of several key components, more precisely: (i) an MPC for optimal trajectory planning under consideration of the aerial robot's dynamics and a set of soft constraints (vision, obstacles, and grasping), (ii) a SMC that tracks the activation force on the gripper over the grasping interval, (iii) a fast GPU-accelerated object detection pipeline tailored to weak but power-efficient embedded hardware such as the Raspberry Pi. All of these building blocks are combined in a layered system architecture that orchestrates the interplay between the individual components such that the system is able to perform autonomous grasping operations.

Finally, the architecture was tested in a simulation where a successful IED disposal task was performed. In that scenario, a cylindrical IED hidden behind a pillar in a parking garage was autonomously picked up by the AM using its vision system and simulated universal

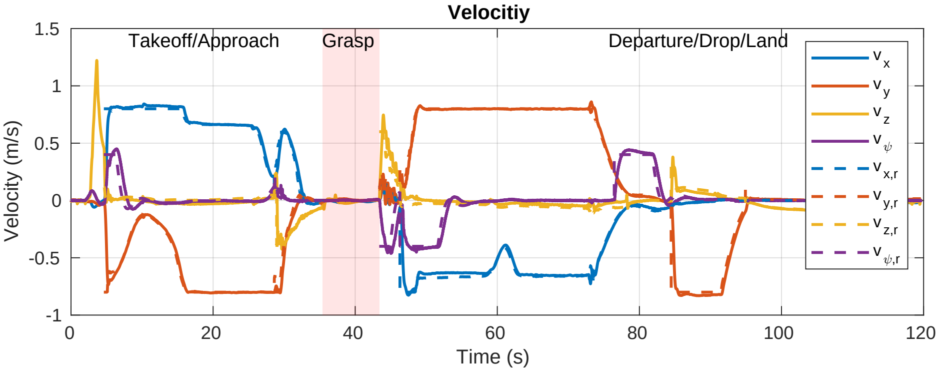


Figure 7.21: The MPC commands velocities in x, y, z, and yaw directions, which are tracked by the lower-level controllers, including the SMC. The payload enters the field of view of the camera at the 30 s mark, where the MPC takes corrective actions such that the payload stays in view. The UAV carefully approaches the payload at the 27 s mark and departs towards the drop zone. After dropping the payload, the UAV lands back on the gripper at its start location.

gripper. It was then transported to a safe location, where it was released for further treatment by the respective specialists.

Future work includes adding visual SLAM to the architecture, which, by mapping the environment, provides the necessary data to generate the obstacle list on-the-fly. In addition to that, pathfinding (e.g., A*) is required to navigate complex environments as the MPC-based trajectory planning cannot provide a globally optimal path. A prime candidate for this could be *voxblox* presented in [166], which is specifically designed to work in unexplored, unstructured environments and with modest compute requirements thanks to its use of truncated signed distance fields. Within this context, it is also beneficial to include cooperative sensing with multiple drones, as shown in our previous work [167], which would permit faster and more precise mapping of the environment and the drones operating therein.

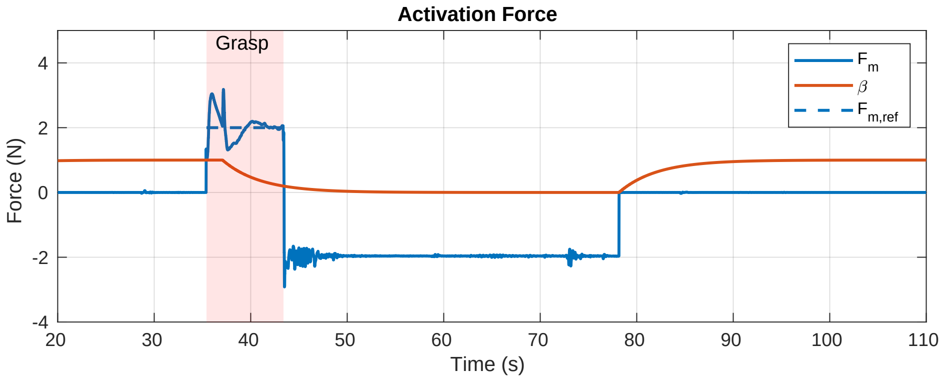


Figure 7.22: A nominal force of 2 N is commanded during the contact period between 35 s and 43 s. The gripper is closed after reaching steady-state at $t = 37$ s. After takeoff, the sensor is used to get an estimate for the weight of the payload, here 200 g, resp. −1.95 N. At the 78 s mark, the payload is dropped into the safe area. The state variable β represents the internal state of the gripper (1 for fluidized, 0 for stiff).

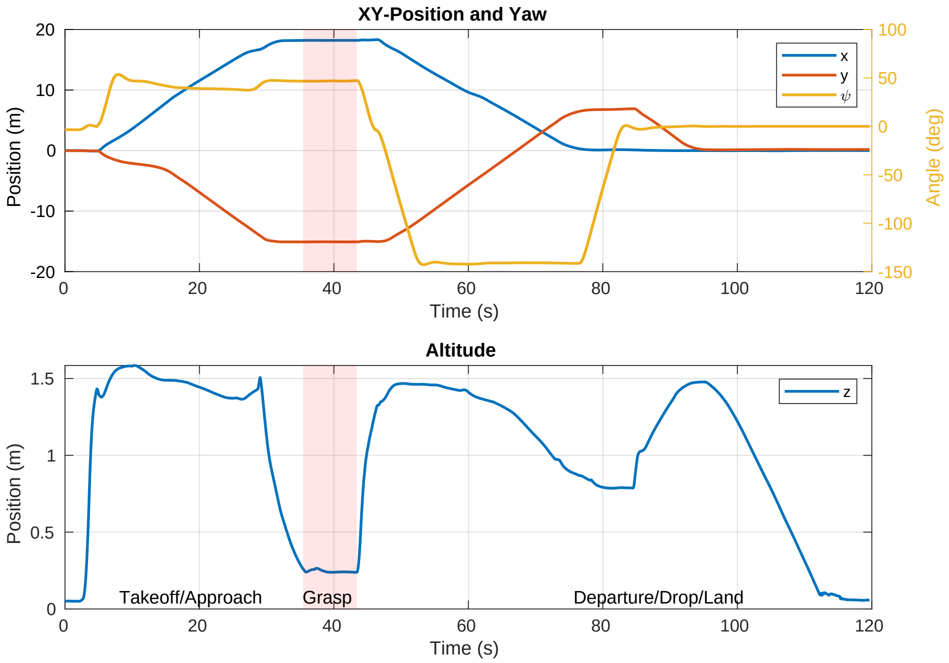


Figure 7.23: The xy-position and yaw angle (top) and the altitude (bottom). The visual constraints lead to the altitude to first rise before sharply decreasing as the UAV comes very close to the payload and the visual constraint is phased out. Likewise, a small adjustment in yaw can be observed at that point (29 s mark).

Chapter 8

Conclusion and Future Work

The context of this thesis is defined by the idea of using aerial robots for IED disposal and thus leveraging their high mobility to effortlessly reach places that are inaccessible for ODRs. Context aside, this thesis provides several contributions to the state of the art in aerial manipulation, which are stated as follows:

- A mechatronic concept of an AM capable of disposing IEDs was introduced. To that aim, this work presented and compared two concepts based on serial manipulators; a tendon-driven arm and a modular arm based on joint modules (direct drive).
- A hybrid modeling approach for aerial manipulators that combines Lagrangian dynamics with quaternion attitude representation was presented. This resulted in a very flexible, easy-to-use, yet powerful dynamics framework that can generate a singularity-free dynamics model of virtually any UAV equipped with an open-chain manipulator.
- A lightweight Universal Gripper with low activation force for aerial grasping was developed. This gripper (TRIGGER) adapts and miniaturizes the original concept targeting large industrial robots. The presented gripper is designed explicitly for small aerial robots with a limited payload and power budget. Furthermore, it is highly integrated, compliant, intelligent and tolerant of positional errors. Several important results were presented: (i) a simulation model that can be used in robotic simulators, (ii) the rela-

tion between fill percentage and activation force was established with to goal of finding the optimal activation force for this specific gripper in an aerial context, (iii) a simple way to significantly improve the holding force by making use of a silicone additive.

- An autonomous aerial grasping concept tailored to TRIGGER was introduced. The concept includes the development of a fast GPU-accelerated object detection pipeline for embedded systems. Additionally, an MPC-based trajectory planner was shown that handles several soft constraints such as vision and obstacles and furthermore respects the particularities of the gripper. A SMC was designed to track the activation force during the grasping period of the gripper, which is vital for successful grasps. The layered system architecture was then presented and validated by performing an IED disposal task in simulation.

The contributions have been published in two journal articles:

- Paul Kremer, Jose Luis Sanchez-Lopez, and Holger Voos. "A Hybrid Modelling Approach for Aerial Manipulators". In: *Journal of Intelligent and Robotic Systems: Theory and Applications* 105.4 (Aug. 2022), p. 74. ISSN: 15730409. DOI: `10.1007/s10846-022-01640-1`. arXiv: `2206.08644`. URL: `https://link.springer.com/10.1007/s10846-022-01640-1`,
- Paul Kremer et al. "TRIGGER: A Lightweight Universal Jamming Gripper for Aerial Grasping". In: (Aug. 2022). arXiv: `2208.10768`. URL: `http://arxiv.org/abs/2208.10768` (resubmitted IEEE Access),

where the work of the latter was partially presented to the soft aerial robotics community during the "Soft Aerial Robotics Workshop" (SoRo) on April 4, 2022, in Edinburgh, UK. The work in Chapter 7 remains to be published.

Future lines of research include:

- Integration: A serial link manipulator on a fully actuated platform. The current AM has the UG mounted in a 'claw' configuration, which limits the workspace of the robot, e.g., it cannot grasp objects located close to walls as the body (and propellers) would collide with the environment. The developed gripper would thus need to be mounted on an AM with a serial manipulator (see Chapter 4) that greatly extends the grasping range beyond the UAV's body. TRIGGER is particularly well suited as a serial manipulator's end effector since all heavy components can easily be placed at the robot's base. The UAV itself would also greatly benefit from being fully actuated or even over-actuated to minimize the dynamic coupling between the drone and robotic arm.

- Grasping is a very complex task, even in well-controlled industrial environments. Grasping unknown objects in unstructured environments is even more challenging, as the geometry and precise location of the objects are unknown. Several publications handle optimal grasping of unknown objects, e.g., [168] and [169]. The Universal Gripper mechanically alleviates this problem but would still benefit from grasping an object in an optimal position.
- Design and development of a fully featured, intuitive Human Machine Interface (HMI). Within the context of IED disposal, the current concept only features a terminal-based HMI, with limited control. However, experts in the field should be confronted with an intuitive graphical interface that supports them in guiding the AM. The HMI thus has to facilitate the transfer of knowledge from the aerial system to the operator and, likewise, has to be able to integrate the expert knowledge of the specialist. A possible solution could, e.g., be an interactive augmented reality environment that the operator sees through an equipped Virtual Reality Headset.

Bibliography

[1] John J. Pantoja et al. "Characterization, Modeling, and Statistical Analysis of the Electromagnetic Response of Inert Improvised Explosive Devices". In: *IEEE Transactions on Electromagnetic Compatibility* 56.2 (Apr. 2014), pp. 393–403. ISSN: 0018-9375. DOI: `10.1109/TEMC.2013.2284964`. URL: `http://ieeexplore.ieee.org/document/6637021/`.

[2] G. Motrycz. "Cases of using improvised explosive devices". In: *Szybkobiezne Pojazdy Gasienicowe* 44.2 (2017), pp. 95–108.

[3] Han Liu et al. "Fragments velocity distribution and estimating method of thin-walled cylindrical improvised explosive devices with different length-to-diameter ratios". In: *Thin-Walled Structures* 175.November 2021 (June 2022), p. 109212. ISSN: 02638231. DOI: `10.1016/j.tws.2022.109212`. URL: `https://linkinghub.elsevier.com/retrieve/pii/S026382312200180X`.

[4] S Costo and R Molfino. "A New Robotic Unit for Onboard Airplanes Bomb Disposal". In: *35th International Symposium on Robotics ISR 2004*. March. 2004, pp. 23–26.

[5] András Domján. "The "Evolution" of Improvised Explosive Devices (IED) in the Light of Technical Development". In: *Műszaki Katonai Közlöny* 32.1 (May 2022), pp. 49–61. ISSN: 2063-4986. DOI: `10.32562/mkk.2022.1.4`. URL: `https://folyoirat.ludovika.hu/index.php/mkk/article/view/5871`.

[6] United Nations. *Improvised Explosive Devices (IEDs) Publication.* URL: https://www.un.org/disarmament/convarms/ieds2/ (visited on Aug. 11, 2022).

[7] Emily Griffith. *Explosive Violence in November 2021.* 2021. URL: https://aoav.org.uk/2021/explosive-violence-in-november-2021/ (visited on Aug. 12, 2022).

[8] Bundeswehr. *Kampfmittelbeseitiger.* 2012. URL: https://www.youtube.com/watch?v=u2URIjt5FDQ (visited on June 24, 2019).

[9] V. Tangtongkid, K. Suwanpakpraek, and B. Patamaprohm. "Design of Lightweight Composite Barrel for Water Jet Disruptor Unit in Bomb Disposal Robot". In: *International Journal of Mechanical Engineering and Robotics Research* 11.3 (2022), pp. 138–144. ISSN: 22780149. DOI: 10.18178/ijmerr.11.3.138-144. URL: http://www.ijmerr.com/index.php?m=content{\&}c=index{\&}a=show{\&}catid=207{\&}id=1707.

[10] Hoa G. Nguyen and John P. Bott. "Robotics for law enforcement: Applications beyond explosive ordnance disposal". In: *SPIE International Symposium on Law Enforcement Technologies.* Ed. by Simon K. Bramble, Edward M. Carapezza, and Lenny I. Rudin. November. Feb. 2001, pp. 433–454. DOI: 10.1117/12.417561. URL: http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=897910.

[11] Paul Kremer, Jose Luis Sanchez-Lopez, and Holger Voos. "A Hybrid Modelling Approach for Aerial Manipulators". In: *Journal of Intelligent and Robotic Systems: Theory and Applications* 105.4 (Aug. 2022), p. 74. ISSN: 15730409. DOI: 10.1007/s10846-022-01640-1. arXiv: 2206.08644. URL: https://link.springer.com/10.1007/s10846-022-01640-1.

[12] Paul Kremer et al. "TRIGGER: A Lightweight Universal Jamming Gripper for Aerial Grasping". In: (Aug. 2022). arXiv: 2208.10768. URL: http://arxiv.org/abs/2208.10768.

[13] Chun Fui Liew et al. "Recent Developments in Aerial Robotics: A Survey and Prototypes Overview". In: (Nov. 2017), pp. 1–14. arXiv: 1711.10085. URL: http://arxiv.org/abs/1711.10085.

[14] David W. Casbeer et al. “Cooperative forest fire surveillance using a team of small unmanned air vehicles”. In: *International Journal of Systems Science* 37.6 (May 2006), pp. 351–360. ISSN: 0020-7721. DOI: `10.1080/00207720500438480`. URL: `http://www.tandfonline.com/doi/abs/10.1080/00207720500438480`.

[15] R.W. Beard et al. “Decentralized Cooperative Aerial Surveillance Using Fixed-Wing Miniature UAVs”. In: *Proceedings of the IEEE* 94.7 (July 2006), pp. 1306–1324. ISSN: 0018-9219. DOI: `10.1109/JPROC.2006.876930`. URL: `http://ieeexplore.ieee.org/document/1677946/`.

[16] Bruno S. Faiçal et al. “The use of unmanned aerial vehicles and wireless sensor networks for spraying pesticides”. In: *Journal of Systems Architecture* 60.4 (Apr. 2014), pp. 393–404. ISSN: 13837621. DOI: `10.1016/j.sysarc.2014.01.004`. URL: `https://linkinghub.elsevier.com/retrieve/pii/S1383762114000204`.

[17] Cornelius A. Thiels et al. “Use of Unmanned Aerial Vehicles for Medical Product Transport”. In: *Air Medical Journal* 34.2 (Mar. 2015), pp. 104–108. ISSN: 1067991X. DOI: `10.1016/j.amj.2014.10.011`. URL: `https://linkinghub.elsevier.com/retrieve/pii/S1067991X14003332`.

[18] Haiyang Chao, Yongcan Cao, and Yangquan Chen. “Autopilots for small unmanned aerial vehicles: A survey”. In: *International Journal of Control, Automation and Systems* 8.1 (Feb. 2010), pp. 36–44. ISSN: 1598-6446. DOI: `10.1007/s12555-010-0105-z`. URL: `http://link.springer.com/10.1007/s12555-010-0105-z`.

[19] Jose Luis Sanchez-Lopez, Manuel Castillo-Lopez, and Holger Voos. “Semantic situation awareness of ellipse shapes via deep learning for multirotor aerial robots with a 2D LIDAR”. In: *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, Sept. 2020, pp. 1014–1023. ISBN: 978-1-7281-4278-4. DOI: `10.1109/ICUAS48674.2020.9214063`. URL: `https://ieeexplore.ieee.org/document/9214063/`.

[20] Ramy Rashad et al. “Fully Actuated Multirotor UAVs: A Literature Review”. In: *IEEE Robotics & Automation Magazine* 27.3 (Sept. 2020), pp. 97–107. ISSN: 1070-9932. DOI:

10.1109/MRA.2019.2955964. URL: https://ieeexplore.ieee.org/document/8978486/.

[21] Tiago P. Nascimento and Martin Saska. “Position and attitude control of multi-rotor aerial vehicles: A survey”. In: *Annual Reviews in Control* 48 (2019), pp. 129–146. ISSN: 13675788. DOI: 10.1016/j.arcontrol.2019.08.004. URL: https://linkinghub.elsevier.com/retrieve/pii/S1367578819300483.

[22] Hossein Bonyan Khamseh, Farrokh Janabi-Sharifi, and Abdelkader Abdessameud. “Aerial manipulation—A literature survey”. In: *Robotics and Autonomous Systems* 107 (Sept. 2018), pp. 221–235. ISSN: 09218890. DOI: 10.1016/j.robot.2018.06.012. URL: https://linkinghub.elsevier.com/retrieve/pii/S0921889017305535.

[23] Fabio Ruggiero, Vincenzo Lippiello, and Anibal Ollero. “Aerial Manipulation: A Literature Review”. In: *IEEE Robotics and Automation Letters* 3.3 (July 2018), pp. 1957–1964. ISSN: 2377-3766. DOI: 10.1109/LRA.2018.2808541. URL: https://ieeexplore.ieee.org/document/8299552/.

[24] Anibal Ollero et al. “The AEROARMS Project: Aerial Robots with Advanced Manipulation Capabilities for Inspection and Maintenance”. In: *IEEE Robotics & Automation Magazine* 25.4 (Dec. 2018), pp. 12–23. ISSN: 1070-9932. DOI: 10.1109/MRA.2018.2852789. URL: https://ieeexplore.ieee.org/document/8435987/.

[25] Jonathan Cacace et al. “Safe Local Aerial Manipulation for the Installation of Devices on Power Lines: AERIAL-CORE First Year Results and Designs”. In: *Applied Sciences* 11.13 (July 2021), p. 6220. ISSN: 2076-3417. DOI: 10.3390/app11136220. URL: https://www.mdpi.com/2076-3417/11/13/6220.

[26] Nathan Michael, Jonathan Fink, and Vijay Kumar. “Cooperative manipulation and transportation with aerial robots”. In: *Autonomous Robots* 30.1 (Jan. 2011), pp. 73–86. ISSN: 0929-5593. DOI: 10.1007/s10514-010-9205-0. URL: http://link.springer.com/10.1007/s10514-010-9205-0.

[27] Takahiro Ikeda et al. “Stable impact and contact force control by UAV for inspection of floor slab of bridge”. In: *Advanced Robotics* 32.19 (Oct. 2018), pp. 1061–1076. ISSN: 0169-1864. DOI: 10.1080/01691864.2018.1525075. URL: https://www.tandfonline.com/doi/full/10.1080/01691864.2018.1525075.

[28] Jesus M. Gomez-de-Gabriel et al. “Methods for Autonomous Wristband Placement with a Search-and-Rescue Aerial Manipulator”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Oct. 2018, pp. 7838–7844. ISBN: 978-1-5386-8094-0. DOI: 10.1109/IROS.2018.8594202. URL: https://ieeexplore.ieee.org/document/8594202/.

[29] Jamy Li and Farrokh Janabi-Sharifi. “Public Opinion About the Benefit, Risk, and Acceptance of Aerial Manipulation Systems”. In: *IEEE Transactions on Human-Machine Systems* 52.5 (Oct. 2022), pp. 1069–1085. ISSN: 2168-2291. DOI: 10.1109/THMS.2022.3164775. URL: https://ieeexplore.ieee.org/document/9763559/.

[30] Roberto Naldi et al. “Output tracking for quadrotor-based aerial manipulators”. In: *Proceedings of the American Control Conference*. Vol. 2015-July. 2. IEEE, July 2015, pp. 1855–1860. ISBN: 9781479986842. DOI: 10.1109/ACC.2015.7171003. URL: http://ieeexplore.ieee.org/document/7171003/.

[31] Matko Orsag et al. “Hybrid adaptive control for aerial manipulation”. In: *Journal of Intelligent and Robotic Systems: Theory and Applications* 73.1-4 (Jan. 2014), pp. 693–707. ISSN: 15730409. DOI: 10.1007/s10846-013-9936-1. URL: http://link.springer.com/10.1007/s10846-013-9936-1.

[32] Xilun DING et al. “A review of aerial manipulation of small-scale rotorcraft unmanned robotic systems”. In: *Chinese Journal of Aeronautics* 32.1 (Jan. 2019), pp. 200–214. ISSN: 10009361. DOI: 10.1016/j.cja.2018.05.012. URL: https://linkinghub.elsevier.com/retrieve/pii/S1000936118301894.

[33] Abdullah Mohiuddin et al. “A Survey of Single and Multi-UAV Aerial Manipulation”. In: *Unmanned Systems* 08.02 (Apr. 2020), pp. 119–147. ISSN: 2301-3850. DOI: 10.

1142/S2301385020500089. URL: https://www.worldscientific.com/doi/abs/10.1142/S2301385020500089.

[34] Robert Ladig et al. "Aerial Manipulation Using Multirotor UAV: A Review from the Aspect of Operating Space and Force". In: *Journal of Robotics and Mechatronics* 33.2 (Apr. 2021), pp. 196–204. ISSN: 1883-8049. DOI: 10.20965/jrm.2021.p0196. URL: https://www.fujipress.jp/jrm/rb/robot003300020196.

[35] Andrew McLaren et al. "A Passive Closing, Tendon Driven, Adaptive Robot Hand for Ultra-Fast, Aerial Grasping and Perching". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Nov. 2019, pp. 5602–5607. ISBN: 978-1-7281-4004-9. DOI: 10.1109/IROS40897.2019.8968076. URL: https://ieeexplore.ieee.org/document/8968076/.

[36] Haili Li et al. "An untethered soft robotic gripper with high payload-to-weight ratio". In: *Mechanism and Machine Theory* 158 (Apr. 2021), p. 104226. ISSN: 0094114X. DOI: 10.1016/j.mechmachtheory.2020.104226. URL: https://linkinghub.elsevier.com/retrieve/pii/S0094114X20304432.

[37] Aurel Appius et al. "RAPTOR: Rapid Aerial Pickup and Transport of Objects by Robots". In: *arXiv preprint* (Mar. 2022), pp. 1–6. arXiv: 2203.03018. URL: http://arxiv.org/abs/2203.03018.

[38] Alejandro Suarez, Guillermo Heredia, and Anibal Ollero. "Lightweight compliant arm with compliant finger for aerial manipulation and inspection". In: *IEEE International Conference on Intelligent Robots and Systems*. Vol. 2016-Novem. IEEE, Oct. 2016, pp. 4449–4454. ISBN: 9781509037629. DOI: 10.1109/IROS.2016.7759655. URL: http://ieeexplore.ieee.org/document/7759655/.

[39] Xiangdong Meng, Yuqing He, and Jianda Han. "Erratum: Survey on Aerial Manipulator: System, Modeling, and Control (Robotica (2020) 38: 7 (1288–1317) DOI: 10.1017/S0263574719001450)". In: *Robotica* 38.7 (July 2020), p. 1343. ISSN: 14698668. DOI: 10.1017/S026357472000048X. URL: https://www.cambridge.org/core/product/identifier/S026357472000048X/type/journal{_}article.

[40] Xu Wei-hong, Cao Li-jia, and Zhong Chun-lai. “Review of Aerial Manipulator and its Control”. In: *International Journal of Robotics and Control Systems* 1.3 (Sept. 2021), pp. 308–325. ISSN: 2775-2658. DOI: `10.31763/ijrcs.v1i3.363`. URL: `https://pubs2.ascee.org/index.php/IJRCS/article/view/363`.

[41] D.H. Wolpert and W.G. Macready. “No free lunch theorems for optimization”. In: *IEEE Transactions on Evolutionary Computation* 1.1 (Apr. 1997), pp. 67–82. ISSN: 1089778X. DOI: `10.1109/4235.585893`. URL: `http://ieeexplore.ieee.org/document/585893/`.

[42] Baohua Zhang et al. “State-of-the-art robotic grippers, grasping and control strategies, as well as their applications in agricultural robots: A review”. In: *Computers and Electronics in Agriculture* 177.April (Oct. 2020), p. 105694. ISSN: 01681699. DOI: `10.1016/j.compag.2020.105694`. URL: `https://linkinghub.elsevier.com/retrieve/pii/S0168169920311030`.

[43] Suseong Kim, Seungwon Choi, and H. Jin Kim. “Aerial manipulation using a quadrotor with a two DOF robotic arm”. In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, Nov. 2013, pp. 4990–4995. ISBN: 978-1-4673-6358-7. DOI: `10.1109/IROS.2013.6697077`. URL: `http://ieeexplore.ieee.org/document/6697077/`.

[44] G. Heredia et al. “Control of a multirotor outdoor aerial manipulator”. In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Iros. Chicago, IL, USA: IEEE, Sept. 2014, pp. 3417–3422. ISBN: 978-1-4799-6934-0. DOI: `10.1109/IROS.2014.6943038`. URL: `http://ieeexplore.ieee.org/document/6943038/`.

[45] Cecilia Laschi, Barbara Mazzolai, and Matteo Cianchetti. “Soft robotics: Technologies and systems pushing the boundaries of robot abilities”. In: *Science Robotics* 1.1 (Dec. 2016), pp. 1–12. ISSN: 2470-9476. DOI: `10.1126/scirobotics.aah3690`. URL: `https://www.science.org/doi/10.1126/scirobotics.aah3690`.

[46] Shatadal Mishra et al. “Design and Control of a Hexacopter With Soft Grasper for Autonomous Object Detection and Grasping”. In: *Volume 3: Modeling and Valida-*

tion; Multi-Agent and Networked Systems; Path Planning and Motion Control; Tracking Control Systems; Unmanned Aerial Vehicles (UAVs) and Application; Unmanned Ground and Aerial Vehicles; Vibration in Mechanical Systems; Vibrat. September. American Society of Mechanical Engineers, Sept. 2018, V003T36A003. ISBN: 978-0-7918-5191-3. DOI: `10.1115/DSCC2018-9107`. URL: `https://asmedigitalcollection.asme.org/DSCC/proceedings/DSCC2018/51913/Atlanta,Georgia,USA/270951`.

[47] Joshua Fishman et al. “Dynamic Grasping with a ”Soft” Drone: From Theory to Practice”. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Sept. 2021, pp. 4214–4221. ISBN: 978-1-6654-1714-3. DOI: `10.1109/IROS51168.2021.9635927`. arXiv: `2103.06465`. URL: `https://ieeexplore.ieee.org/document/9635927/`.

[48] Justin Thomas et al. “Toward autonomous avian-inspired grasping for micro aerial vehicles”. In: *Bioinspiration & Biomimetics* 9.2 (May 2014), p. 025010. ISSN: 1748-3182. DOI: `10.1088/1748-3182/9/2/025010`. URL: `https://iopscience.iop.org/article/10.1088/1748-3182/9/2/025010`.

[49] Haijie Zhang et al. “Compliant Bistable Grippers Enable Passive Perching for Micro Aerial Vehicles”. In: *IEEE/ASME Transactions on Mechatronics* 26.5 (Oct. 2021), pp. 2316–2326. ISSN: 1083-4435. DOI: `10.1109/TMECH.2020.3037303`. URL: `https://ieeexplore.ieee.org/document/9257095/`.

[50] Todd W. Danko and Paul Y. Oh. “Design and Control of a Hyper-Redundant Manipulator for Mobile Manipulating Unmanned Aerial Vehicles”. In: *Journal of Intelligent & Robotic Systems* 73.1-4 (Jan. 2014), pp. 709–723. ISSN: 0921-0296. DOI: `10.1007/s10846-013-9935-2`. URL: `http://link.springer.com/10.1007/s10846-013-9935-2`.

[51] Alejandro Suarez et al. “Aerial manipulator with rolling base for inspection of pipe arrays”. In: *IEEE Access* 8 (2020), pp. 162516–162532. ISSN: 21693536. DOI: `10.1109/ACCESS.2020.3021126`.

[52] Josep Martí-Saumell et al. “Full-Body Torque-Level Non-linear Model Predictive Control for Aerial Manipulation”. In: (July 2021). arXiv: 2107.03722. URL: http://arxiv.org/abs/2107.03722.

[53] Hossein Bonyan Khamseh and Farrokh Janabi-Sharifi. “UKF-based LQR control of a manipulating unmanned aerial vehicle”. In: *Unmanned Systems* 5.3 (July 2017), pp. 131–139. ISSN: 23013869. DOI: 10.1142/S2301385017400015. URL: https://www.worldscientific.com/doi/abs/10.1142/S2301385017400015.

[54] Fengyu Quan et al. “Simulation Platform for Autonomous Aerial Manipulation in Dynamic Environments”. In: *2021 IEEE International Conference on Robotics and Biomimetics, ROBIO 2021*. IEEE, Dec. 2021, pp. 589–594. ISBN: 9781665405355. DOI: 10.1109/ROBIO54168.2021.9739356. arXiv: 2103.10792. URL: https://ieeexplore.ieee.org/document/9739356/.

[55] Marco Tognon, Sanket S. Dash, and Antonio Franchi. “Observer-Based Control of Position and Tension for an Aerial Robot Tethered to a Moving Platform”. In: *IEEE Robotics and Automation Letters* 1.2 (July 2016), pp. 732–737. ISSN: 23773766. DOI: 10.1109/LRA.2016.2523599. URL: http://ieeexplore.ieee.org/document/7395300/.

[56] Hanspeter Schaub and John L. Junkins. “Stereographic orientation parameters for attitude dynamics: A generalization of the Rodrigues parameters”. In: *Journal of the Astronautical Sciences* 44.1 (1996), pp. 1–19. ISSN: 00219142.

[57] Basile Graf. “Quaternions and dynamics”. In: *arXiv: Dynamical Systems* (Nov. 2008). arXiv: 0811.2889. URL: http://arxiv.org/abs/0811.2889.

[58] Michael Möller and Christoph Glocker. “Rigid body dynamics with a scalable body, quaternions and perfect constraints”. In: *Multibody System Dynamics* 27.4 (Apr. 2012), pp. 437–454. ISSN: 1384-5640. DOI: 10.1007/s11044-011-9276-5. URL: http://link.springer.com/10.1007/s11044-011-9276-5.

[59] Joan Solà. “Quaternion kinematics for the error-state Kalman filter”. In: *arXiv* (Nov. 2017). arXiv: 1711.02508. URL: http://arxiv.org/abs/1711.02508.

[60] Joan Solà, Jeremie Deray, and Dinesh Atchuthan. “A micro Lie theory for state estimation in robotics”. In: *arXiv* (Dec. 2018), pp. 1–17. arXiv: `1812.01537`. URL: `http://arxiv.org/abs/1812.01537`.

[61] Richard P. Feynman et al. “The Feynman Lectures on Physics; Vol. I”. In: *American Journal of Physics* 33.9 (Sept. 1965), pp. 750–752. ISSN: 0002-9505. DOI: `10.1119/1.1972241`. URL: `http://aapt.scitation.org/doi/10.1119/1.1972241`.

[62] Herbert Goldstein et al. *Classical Mechanics, 3rd ed.* thrid edit. New York, NY: Pearson, July 2002, p. 638. ISBN: 978-0-201-65702-9. DOI: `10.1119/1.1484149`.

[63] Dietmar Gross et al. *Technische Mechanik 3*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2019. ISBN: 978-3-662-59550-3. DOI: `10.1007/978-3-662-59551-0`. URL: `http://link.springer.com/10.1007/978-3-662-59551-0`.

[64] S. J. Gambhire et al. “Review of sliding mode based control techniques for control system applications”. In: *International Journal of Dynamics and Control* 9.1 (Mar. 2021), pp. 363–378. ISSN: 2195-268X. DOI: `10.1007/s40435-020-00638-7`. URL: `https://link.springer.com/10.1007/s40435-020-00638-7`.

[65] Jinkun Liu. *Sliding mode control using MATLAB*. 2017, pp. 1–332. ISBN: 9780128025758.

[66] Midhun Augustine. *Into the sliding modes*. February. 2019. DOI: `10.13140/RG.2.2.17045.37603`.

[67] Igor Boiko et al. “Analysis of Chattering in Systems With Second-Order Sliding Modes”. In: *IEEE Transactions on Automatic Control* 52.11 (Nov. 2007), pp. 2085–2102. ISSN: 0018-9286. DOI: `10.1109/TAC.2007.908319`. URL: `http://ieeexplore.ieee.org/document/4380494/`.

[68] Vadim I. Utkin, Alex S. Poznyak, and Patricio Ordaz. “Adaptive super-twist control with minimal chattering effect”. In: *IEEE Conference on Decision and Control and European Control Conference*. IEEE, Dec. 2011, pp. 7009–7014. ISBN: 978-1-61284-801-3. DOI: `10.1109/CDC.2011.6160720`. URL: `http://ieeexplore.ieee.org/document/6160720/`.

[69] Yuri Shtessel et al. *Sliding Mode Control and Observation.* Control Engineering. New York, NY: Springer New York, 2014, pp. 1–356. ISBN: 978-0-8176-4892-3. DOI: `10.1007/978-0-8176-4893-0`. URL: `http://link.springer.com/10.1007/978-0-8176-4893-0`.

[70] Moritz Diehl et al. “Fast Direct Multiple Shooting Algorithms for Optimal Robot Control”. In: *Fast Motions in Biomechanics and Robotics.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 65–93. DOI: `10.1007/978-3-540-36119-0_4`. URL: `http://link.springer.com/10.1007/978-3-540-36119-0{\_}4`.

[71] Guido Sanchez et al. “MPC for nonlinear systems: A comparative review of discretization methods”. In: *2017 XVII Workshop on Information Processing and Control (RPIC).* Vol. 2017-Janua. 2. IEEE, Sept. 2017, pp. 1–6. ISBN: 978-987-544-754-7. DOI: `10.23919/RPIC.2017.8214333`. URL: `http://ieeexplore.ieee.org/document/8214333/`.

[72] Emil Fresk and George Nikolakopoulos. “A generalized Frame Adaptive MPC for the low-level control of UAVs”. In: *2018 European Control Conference (ECC).* 644128. IEEE, June 2018, pp. 1815–1820. ISBN: 978-3-9524-2698-2. DOI: `10.23919/ECC.2018.8550210`. URL: `https://ieeexplore.ieee.org/document/8550210/`.

[73] Boris Houska, Hans Joachim Ferreau, and Moritz Diehl. “ACADO toolkit-An open-source framework for automatic control and dynamic optimization”. In: *Optimal Control Applications and Methods* 32.3 (May 2011), pp. 298–312. ISSN: 01432087. DOI: `10.1002/oca.939`. URL: `https://onlinelibrary.wiley.com/doi/10.1002/oca.939`.

[74] Robin Verschueren et al. “acados—a modular open-source framework for fast embedded optimal control”. In: *Mathematical Programming Computation* 14.1 (Mar. 2022), pp. 147–183. ISSN: 1867-2949. DOI: `10.1007/s12532-021-00208-8`. arXiv: `1910.13753`. URL: `https://link.springer.com/10.1007/s12532-021-00208-8`.

[75] Pantelis Sopasakis, Emil Fresk, and Panagiotis Patrinos. “OpEn: Code Generation for Embedded Nonconvex Optimization”. In: *IFAC-PapersOnLine* 53.2 (2020), pp. 6548–

6554. ISSN: 24058963. DOI: 10.1016/j.ifacol.2020.12.071. arXiv: 2003.00292. URL: https://linkinghub.elsevier.com/retrieve/pii/S240589632030327X.

[76] Joel A. E. Andersson et al. “CasADi: a software framework for nonlinear optimization and optimal control”. In: *Mathematical Programming Computation* 11.1 (Mar. 2019), pp. 1–36. ISSN: 1867-2949. DOI: 10.1007/s12532-018-0139-4. URL: http://link.springer.com/10.1007/s12532-018-0139-4.

[77] Mark A. Hinton et al. “Advanced explosive ordnance disposal robotic system (AEODRS): A common architecture revolution”. In: *Johns Hopkins APL Technical Digest (Applied Physics Laboratory)* 30.3 (2011), pp. 256–266. ISSN: 02705214.

[78] Jose Luis Sanchez-Lopez et al. “A Multi-Layered Component-Based Approach for the Development of Aerial Robotic Systems: The Aerostack Framework”. In: *Journal of Intelligent & Robotic Systems* 88.2-4 (Dec. 2017), pp. 683–709. ISSN: 0921-0296. DOI: 10.1007/s10846-017-0551-4. URL: http://link.springer.com/10.1007/s10846-017-0551-4.

[79] Hangjin Wu et al. “Intelligent Explosive Ordnance Disposal UAV System Based on Manipulator and Real-Time Object Detection”. In: *2021 4th International Conference on Intelligent Robotics and Control Engineering (IRCE)*. 3. IEEE, Sept. 2021, pp. 61–65. ISBN: 978-1-6654-1348-0. DOI: 10.1109/IRCE53649.2021.9570974. URL: https://ieeexplore.ieee.org/document/9570974/.

[80] Tyler C. Looney et al. “Air-Releasable Soft Robots for Explosive Ordnance Disposal”. In: *2022 IEEE 5th International Conference on Soft Robotics, RoboSoft 2022*. IEEE, Apr. 2022, pp. 687–692. ISBN: 9781665408288. DOI: 10.1109/RoboSoft54090.2022.9762219. URL: https://ieeexplore.ieee.org/document/9762219/.

[81] Jiwei Fan et al. “Design and Implementation of Intelligent EOD System Based on Six-Rotor UAV”. In: *Drones* 5.4 (Dec. 2021), p. 146. ISSN: 2504-446X. DOI: 10.3390/drones5040146. URL: https://link.springer.com/10.1007/978-981-16-4258-6{_}144.

[82] Thomas Lens, Jürgen Kunz, and Oskar von Stryk. "Dynamic Modeling of the 4 DoF BioRob Series Elastic Robot Arm for Simulation and Control". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 6472 LNAI. Simpar. 2010, pp. 411–422. ISBN: 3642173187. DOI: `10.1007/978-3-642-17319-6_38`. URL: `http://link.springer.com/10.1007/978-3-642-17319-6{\_}38`.

[83] Serket Quintanar-Guzman et al. "Lightweight robotic arm actuated by shape memory alloy (SMA) wires". In: *2016 8th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*. IEEE, June 2016, pp. 1–6. ISBN: 978-1-5090-2047-8. DOI: `10.1109/ECAI.2016.7861065`. URL: `http://ieeexplore.ieee.org/document/7861065/`.

[84] Matteo Laffranchi, Nikos Tsagarakis, and D.G. Caldwell. "A compact compliant actuator (CompAct&#x2122;) with variable physical damping". In: *2011 IEEE International Conference on Robotics and Automation*. IEEE, May 2011, pp. 4644–4650. ISBN: 978-1-61284-386-5. DOI: `10.1109/ICRA.2011.5979915`. URL: `http://ieeexplore.ieee.org/document/5979915/`.

[85] Vincenzo Lippiello, Bruno Siciliano, and Luigi Villani. "Eye-in-Hand/Eye-to-Hand Multi-Camera Visual Servoing". In: *Proceedings of the 44th IEEE Conference on Decision and Control*. Vol. 2005. IEEE, 2005, pp. 5354–5359. ISBN: 0-7803-9567-0. DOI: `10.1109/CDC.2005.1583013`. URL: `http://ieeexplore.ieee.org/document/1583013/`.

[86] Hoseong Seo, Suseong Kim, and H. Jin Kim. "Aerial grasping of cylindrical object using visual servoing based on stochastic model predictive control". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2017, pp. 6362–6368. ISBN: 978-1-5090-4633-1. DOI: `10.1109/ICRA.2017.7989751`. URL: `http://ieeexplore.ieee.org/document/7989751/`.

[87] Lishan Lin et al. "Autonomous Vision-Based Aerial Grasping for Rotorcraft Unmanned Aerial Vehicles". In: *Sensors* 19.15 (Aug. 2019), p. 3410. ISSN: 1424-8220. DOI: 10.3390/s19153410. URL: https://www.mdpi.com/1424-8220/19/15/3410.

[88] Daniel Mellinger et al. "Design, modeling, estimation and control for aerial grasping and manipulation". In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, Sept. 2011, pp. 2668–2673. ISBN: 978-1-61284-456-5. DOI: 10.1109/IROS.2011.6048556. URL: https://ieeexplore.ieee.org/document/6048556.

[89] A. Suarez et al. "Lightweight and human-size dual arm aerial manipulator". In: *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, June 2017, pp. 1778–1784. ISBN: 978-1-5090-4495-5. DOI: 10.1109/ICUAS.2017.7991357. URL: http://ieeexplore.ieee.org/document/7991357/.

[90] Christopher Korpela, Matko Orsag, and Paul Oh. "Towards valve turning using a dual-arm aerial manipulator". In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, Sept. 2014, pp. 3411–3416. ISBN: 978-1-4799-6934-0. DOI: 10.1109/IROS.2014.6943037. URL: http://ieeexplore.ieee.org/document/6943037/.

[91] Anush Manukyan et al. "Deep Reinforcement Learning-based Continuous Control for Multicopter Systems". In: *2019 6th International Conference on Control, Decision and Information Technologies (CoDIT)*. IEEE, Apr. 2019, pp. 1876–1881. ISBN: 978-1-7281-0521-5. DOI: 10.1109/CoDIT.2019.8820368. URL: https://ieeexplore.ieee.org/document/8820368/.

[92] Oussama Khatib Bruno Siciliano. *Springer Handbook of Robotics*. Ed. by Bruno Siciliano and Oussama Khatib. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. ISBN: 978-3-540-23957-4. DOI: 10.1007/978-3-540-30301-5. URL: http://link.springer.com/10.1007/978-3-540-30301-5.

[93] A.E. Jimenez-Cano et al. "Control of an aerial robot with multi-link arm for assembly tasks". In: *2013 IEEE International Conference on Robotics and Automation.*

IEEE, May 2013, pp. 4916–4921. ISBN: 978-1-4673-5643-5. DOI: 10.1109/ICRA.2013.6631279. URL: http://ieeexplore.ieee.org/document/6631279/.

[94] Ran Jiao et al. “Control of Quadrotor Equipped with a Two DOF Robotic Arm”. In: *2018 3rd International Conference on Advanced Robotics and Mechatronics (ICARM)*. IEEE, July 2018, pp. 437–442. ISBN: 978-1-5386-7066-8. DOI: 10.1109/ICARM.2018.8610770. URL: https://ieeexplore.ieee.org/document/8610770/.

[95] H. Abaunza et al. “Quadrotor aerial manipulator based on dual quaternions”. In: *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, June 2016, pp. 152–161. ISBN: 978-1-4673-9334-8. DOI: 10.1109/ICUAS.2016.7502589. URL: http://ieeexplore.ieee.org/document/7502589/.

[96] Alessandro Tassora. “An optimized Lagrangian multiplier approach for interactive multibody simulation in kinematic and dynamical digital prototyping”. In: *International Symposium on Computer Simulation in Biomechanics*. 2001, pp. 4–6.

[97] Seyyed Ali Emami and Afshin Banazadeh. “Simultaneous trajectory tracking and aerial manipulation using a multi-stage model predictive control”. In: *Aerospace Science and Technology* 112 (May 2021), p. 106573. ISSN: 12709638. DOI: 10.1016/j.ast.2021.106573. URL: https://linkinghub.elsevier.com/retrieve/pii/S1270963821000845.

[98] H. Abaunza et al. “Dual Quaternion Modeling and Control of a Quad-rotor Aerial Manipulator”. In: *Journal of Intelligent & Robotic Systems* 88.2-4 (Dec. 2017), pp. 267–283. ISSN: 0921-0296. DOI: 10.1007/s10846-017-0519-4. URL: http://link.springer.com/10.1007/s10846-017-0519-4.

[99] Matko Orsag, Christopher Korpela, and Paul Oh. “Modeling and Control of MM-UAV: Mobile Manipulating Unmanned Aerial Vehicle”. In: *Journal of Intelligent & Robotic Systems* 69.1-4 (Jan. 2013), pp. 227–240. ISSN: 0921-0296. DOI: 10.1007/s10846-012-9723-4. URL: http://link.springer.com/10.1007/s10846-012-9723-4.

[100] Mohamed Fanni and Ahmed Khalifa. "A New 6-DOF Quadrotor Manipulation System: Design, Kinematics, Dynamics, and Control". In: *IEEE/ASME Transactions on Mechatronics* 22.3 (June 2017), pp. 1315–1326. ISSN: 1083-4435. DOI: 10.1109/TMECH.2017.2681179. URL: http://ieeexplore.ieee.org/document/7875412/.

[101] Tiehua Wang et al. "Dynamic hybrid position/force control for the quadrotor with a multi-degree-of-freedom manipulator". In: *Artificial Life and Robotics* 24.3 (Sept. 2019), pp. 378–389. ISSN: 1433-5298. DOI: 10.1007/s10015-019-00534-0. URL: http://link.springer.com/10.1007/s10015-019-00534-0.

[102] J.A. Acosta, M.I. Sanchez, and A. Ollero. "Robust control of underactuated Aerial Manipulators via IDA-PBC". In: *53rd IEEE Conference on Decision and Control.* Vol. 2015-Febru. February. Los Angeles, CA, USA: IEEE, Dec. 2014, pp. 673–678. ISBN: 978-1-4673-6090-6. DOI: 10.1109/CDC.2014.7039459. URL: http://ieeexplore.ieee.org/document/7039459/.

[103] Zain Anwar Ali and Xinde Li. "Controlling of an Under-Actuated Quadrotor UAV Equipped With a Manipulator". In: *IEEE Access* 8 (2020), pp. 34664–34674. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.2974581. URL: https://ieeexplore.ieee.org/document/9000827/.

[104] Jiacheng Liang et al. "Low-Complexity Prescribed Performance Control for Unmanned Aerial Manipulator Robot System Under Model Uncertainty and Unknown Disturbances". In: *IEEE Transactions on Industrial Informatics* 18.7 (July 2022), pp. 4632–4641. ISSN: 1551-3203. DOI: 10.1109/TII.2021.3117262. URL: https://ieeexplore.ieee.org/document/9565382/.

[105] ROS. *URDF XML Specifications.* URL: http://wiki.ros.org/urdf/XML (visited on Dec. 10, 2020).

[106] Richard M. Murray, Zexiang Li, and S. Shankar Sastry. *A Mathematical Introduction to Robotic Manipulation.* 1st editio. Boca Raton: CRC Press, Dec. 2017, pp. 1–456. ISBN: 9781315136370. DOI: 10.1201/9781315136370. URL: https://www.taylorfrancis.com/books/9781351469791.

[107] John J. Craig. *Introduction to Robotics; Mechanics and Control*. 3rd ed. Upper Saddle River: Pearson Education, May 2005, p. 400. ISBN: 978-0201543612.

[108] Bruno Siciliano et al. “Robotics: modelling, planning and control”. In: *Choice Reviews Online*. Advanced Textbooks in Control and Signal Processing 46.11 (July 2009), pp. 46–6226–46–6226. ISSN: 0009-4978. DOI: `10.5860/CHOICE.46-6226`. URL: `http://link.springer.com/10.1007/978-1-84628-642-1`.

[109] J. Baumgarte. “Stabilization of constraints and integrals of motion in dynamical systems”. In: *Computer Methods in Applied Mechanics and Engineering* 1.1 (June 1972), pp. 1–16. ISSN: 00457825. DOI: `10.1016/0045-7825(72)90018-7`. URL: `https://linkinghub.elsevier.com/retrieve/pii/0045782572900187`.

[110] Firdaus E. Udwadia and Aaron D. Schutte. “An Alternative Derivation of the Quaternion Equations of Motion for Rigid-Body Rotational Dynamics”. In: *Journal of Applied Mechanics* 77.4 (July 2010), pp. 1–4. ISSN: 0021-8936. DOI: `10.1115/1.4000917`. URL: `https://asmedigitalcollection.asme.org/appliedmechanics/article/doi/10.1115/1.4000917/459518/An-Alternative-Derivation-of-the-Quaternion`.

[111] Aaron Schutte and Firdaus Udwadia. “New Approach to the Modeling of Complex Multibody Dynamical Systems”. In: *Journal of Applied Mechanics* 78.2 (Mar. 2011). ISSN: 0021-8936. DOI: `10.1115/1.4002329`. URL: `https://asmedigitalcollection.asme.org/appliedmechanics/article/doi/10.1115/1.4002329/476121/New-Approach-to-the-Modeling-of-Complex-Multibody`.

[112] Magnus Bjerkeng and Kristin Y. Pettersen. “A new Coriolis matrix factorization”. In: *2012 IEEE International Conference on Robotics and Automation*. IEEE, May 2012, pp. 4974–4979. ISBN: 978-1-4673-1405-3. DOI: `10.1109/ICRA.2012.6224820`. URL: `http://ieeexplore.ieee.org/document/6224820/`.

[113] Denis Kotarski et al. “Experimental Identification and Characterization of Multirotor UAV Propulsion”. In: *Journal of Physics: Conference Series* 870.1 (July 2017), p. 012003. ISSN: 1742-6588. DOI: `10.1088/1742-6596/870/1/012003`. URL: `https://iopscience.iop.org/article/10.1088/1742-6596/870/1/012003`.

[114] Myunggon Yoon. “Experimental identification of thrust dynamics for a multirotor helicopter”. In: *International Journal of Engineering Research and Technology* 4.11 (2015), pp. 206–209. URL: `https://www.ijert.org/experimental-identification-of-thrust-dynamics-for-a-multi-rotor-helicopter`.

[115] Sugjoon Yoon, Robert M. Howe, and Donald T. Greenwood. “Constraint violation stabilization using gradient feedback in constrained dynamics simulation”. In: *Journal of Guidance, Control, and Dynamics* 15.6 (Nov. 1992), pp. 1467–1474. ISSN: 0731-5090. DOI: `10.2514/3.11410`. URL: `https://arc.aiaa.org/doi/10.2514/3.11410`.

[116] David J. Braun and Michael Goldfarb. “Eliminating constraint drift in the numerical simulation of constrained dynamical systems”. In: *Computer Methods in Applied Mechanics and Engineering* 198.37-40 (Aug. 2009), pp. 3151–3160. ISSN: 00457825. DOI: `10.1016/j.cma.2009.05.013`. URL: `https://linkinghub.elsevier.com/retrieve/pii/S0045782509002011`.

[117] Karim Sherif, Karin Nachbagauer, and Wolfgang Steiner. “On the rotational equations of motion in rigid body dynamics when using Euler parameters”. In: *Nonlinear Dynamics* 81.1-2 (July 2015), pp. 343–352. ISSN: 0924-090X. DOI: `10.1007/s11071-015-1995-3`. URL: `http://link.springer.com/10.1007/s11071-015-1995-3`.

[118] Jiafeng Xu and Karl Henning Halse. “Dual Quaternion Variational Integrator for Rigid Body Dynamic Simulation”. In: *ArXiv* abs/1611.0.August (Nov. 2016). arXiv: `1611.00616`. URL: `http://arxiv.org/abs/1611.00616`.

[119] Erwin C and Yunfei B. *PyBullet, a Python module for physics simulation for games, robotics and machine learning.* 2021. URL: `https://pybullet.org/wordpress/`.

[120] Evan G. Hemingway and Oliver M. O’Reilly. “Perspectives on Euler angle singularities, gimbal lock, and the orthogonality of applied forces and applied moments”. In: *Multibody System Dynamics* 44.1 (Sept. 2018), pp. 31–56. ISSN: 1384-5640. DOI: `10.1007/s11044-018-9620-0`. URL: `http://link.springer.com/10.1007/s11044-018-9620-0`.

[121] Geneviève Miron, Benjamin Bédard, and Jean-Sébastien Plante. "Sleeved Bending Actuators for Soft Grippers: A Durable Solution for High Force-to-Weight Applications". In: *Actuators* 7.3 (July 2018), p. 40. ISSN: 2076-0825. DOI: `10.3390/act7030040`. URL: `http://www.mdpi.com/2076-0825/7/3/40`.

[122] Markus Lieret et al. "A lightweight, low-cost and self-diagnosing mechatronic jaw gripper for the aerial picking with unmanned aerial vehicles". In: *Procedia Manufacturing* 51 (2020), pp. 424–430. ISSN: 23519789. DOI: `10.1016/j.promfg.2020.10.060`. URL: `https://linkinghub.elsevier.com/retrieve/pii/S2351978920319156`.

[123] Chad C. Kessens et al. "Versatile aerial grasping using self-sealing suction". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. Vol. 2016-June. IEEE, May 2016, pp. 3249–3254. ISBN: 978-1-4673-8026-3. DOI: `10.1109/ICRA.2016.7487495`. URL: `http://ieeexplore.ieee.org/document/7487495/`.

[124] Usman A. Fiaz, M. Abdelkader, and Jeff S. Shamma. "An Intelligent Gripper Design for Autonomous Aerial Transport with Passive Magnetic Grasping and Dual-Impulsive Release". In: *2018 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*. Vol. 2018-July. IEEE, July 2018, pp. 1027–1032. ISBN: 978-1-5386-1854-7. DOI: `10.1109/AIM.2018.8452383`. URL: `https://ieeexplore.ieee.org/document/8452383/`.

[125] Raymond R. Ma, Lael U. Odhner, and Aaron M. Dollar. "A modular, open-source 3D printed underactuated hand". In: *2013 IEEE International Conference on Robotics and Automation*. IEEE, May 2013, pp. 2737–2743. ISBN: 978-1-4673-5643-5. DOI: `10.1109/ICRA.2013.6630954`. URL: `http://ieeexplore.ieee.org/document/6630954/`.

[126] Salvatore D'Avella, Paolo Tripicchio, and Carlo Alberto Avizzano. "A study on picking objects in cluttered environments: Exploiting depth features for a custom low-cost universal jamming gripper". In: *Robotics and Computer-Integrated Manufacturing* 63.October 2019 (June 2020), p. 101888. ISSN: 07365845. DOI: `10.1016/j.rcim.2019.101888`. URL: `https://linkinghub.elsevier.com/retrieve/pii/S0736584519307276`.

[127] Paul E.I. Pounds and Aaron M. Dollar. "Towards grasping with a helicopter platform: Landing accuracy and other challenges". In: *Proceedings of the 2010 Australasian Conference on Robotics and Automation, ACRA 2010* (2010).

[128] Edoardo Milana. "Soft robotics for infrastructure protection". In: *Frontiers in Robotics and AI* 9.November (Nov. 2022), pp. 1–7. ISSN: 2296-9144. DOI: `10.3389/frobt.2022.1026891`. URL: `https://www.frontiersin.org/articles/10.3389/frobt.2022.1026891/full`.

[129] Liam Kruse and Justin Bradley. "A Hybrid, Actively Compliant Manipulator/Gripper for Aerial Manipulation with a Multicopter". In: *2018 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE, Aug. 2018, pp. 1–8. ISBN: 978-1-5386-5572-6. DOI: `10.1109/SSRR.2018.8468651`. URL: `https://ieeexplore.ieee.org/document/8468651/`.

[130] F. Javier Garcia Rubiales et al. "Soft-Tentacle Gripper for Pipe Crawling to Inspect Industrial Facilities Using UAVs". In: *Sensors* 21.12 (June 2021), p. 4142. ISSN: 1424-8220. DOI: `10.3390/s21124142`. URL: `https://www.mdpi.com/1424-8220/21/12/4142`.

[131] Hideyuki Tsukagoshi and Yuichi Osada. "Soft Hybrid Suction Cup Capable of Sticking to Various Objects and Environments". In: *Actuators* 10.3 (Mar. 2021), p. 50. ISSN: 2076-0825. DOI: `10.3390/act10030050`. URL: `https://www.mdpi.com/2076-0825/10/3/50`.

[132] Dario Tscholl et al. "Flying Hydraulically Amplified Electrostatic Gripper System for Aerial Object Manipulation". In: *arXiv preprint* (May 2022). arXiv: `2205.13011`. URL: `http://arxiv.org/abs/2205.13011`.

[133] Eric Brown et al. "Universal robotic gripper based on the jamming of granular material". In: *Proceedings of the National Academy of Sciences* 107.44 (Nov. 2010), pp. 18809–18814. ISSN: 0027-8424. DOI: `10.1073/pnas.1003250107`. arXiv: `1009.4444`. URL: `https://pnas.org/doi/full/10.1073/pnas.1003250107`.

[134] Robert P. Behringer and Bulbul Chakraborty. “The physics of jamming for granular materials: a review”. In: *Reports on Progress in Physics* 82.1 (Jan. 2019), p. 012601. ISSN: 0034-4885. DOI: 10.1088/1361-6633/aadc3c. URL: https://iopscience.iop.org/article/10.1088/1361-6633/aadc3c.

[135] Takeshi Nishida, Yuki Okatani, and Kenjiro Tadakuma. “Development of Universal Robot Gripper Using MRα Fluid”. In: *International Journal of Humanoid Robotics* 13.04 (Dec. 2016), p. 1650017. ISSN: 0219-8436. DOI: 10.1142/S0219843616500171. URL: https://www.worldscientific.com/doi/abs/10.1142/S0219843616500171.

[136] Tatsuya Sakuma et al. “A Universal Gripper Using Optical Sensing to Acquire Tactile Information and Membrane Deformation”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Oct. 2018, pp. 1–9. ISBN: 978-1-5386-8094-0. DOI: 10.1109/IROS.2018.8593697. URL: https://ieeexplore.ieee.org/document/8593697.

[137] John R. Amend et al. “A Positive Pressure Universal Gripper Based on the Jamming of Granular Material”. In: *IEEE Transactions on Robotics* 28.2 (Apr. 2012), pp. 341–350. DOI: 10.1109/TRO.2011.2171093. arXiv: 1009.4444. URL: http://ieeexplore.ieee.org/document/6142115/.

[138] Jaimeen Kapadia and Mark Yim. “Design and performance of nubbed fluidizing jamming grippers”. In: *2012 IEEE International Conference on Robotics and Automation*. 3. IEEE, May 2012, pp. 5301–5306. ISBN: 978-1-4673-1405-3. DOI: 10.1109/ICRA.2012.6225111. URL: http://ieeexplore.ieee.org/document/6225111/.

[139] Julián M. Gómez–Paccapelo et al. “Effect of the granular material on the maximum holding force of a granular gripper”. In: *Granular Matter* 23.1 (Feb. 2021), p. 4. ISSN: 1434-5021. DOI: 10.1007/s10035-020-01069-z. arXiv: 2010.14992. URL: http://link.springer.com/10.1007/s10035-020-01069-z.

[140] Stephen Licht et al. “Universal jamming grippers for deep-sea manipulation”. In: *OCEANS 2016 MTS/IEEE Monterey*. IEEE, Sept. 2016, pp. 1–5. ISBN: 978-1-5090-

1537-5. DOI: 10.1109/OCEANS.2016.7761237. URL: https://ieeexplore.ieee.org/document/7761237/.

[141] Holger Götz et al. “Soft particles reinforce robotic grippers: robotic grippers based on granular jamming of soft particles”. In: *Granular Matter* 24.1 (Feb. 2022), p. 31. ISSN: 1434-5021. DOI: 10.1007/s10035-021-01193-4. arXiv: 2109.03356. URL: https://link.springer.com/10.1007/s10035-021-01193-4.

[142] Zhili Chen, Hamed Rahimi Nohooji, and Chee-Meng Chew. “Development of Topology Optimized Bending-Twisting Soft Finger”. In: *Journal of Mechanisms and Robotics* 14.5 (Oct. 2022), pp. 1–23. ISSN: 1942-4302. DOI: 10.1115/1.4053159. URL: https://asmedigitalcollection.asme.org/mechanismsrobotics/article/14/5/051003/1129058/Development-of-Topology-Optimized-Bending-Twisting.

[143] Hengjia Zhu et al. “Nonlinear dynamic model of air spring with a damper for vehicle ride comfort”. In: *Nonlinear Dynamics* 89.2 (July 2017), pp. 1545–1568. ISSN: 0924-090X. DOI: 10.1007/s11071-017-3535-9. URL: http://link.springer.com/10.1007/s11071-017-3535-9.

[144] Raghav Mishra et al. “Vibration Improves Performance in Granular Jamming Grippers”. In: *arXiv preprint* (Sept. 2021). arXiv: 2109.10496. URL: http://arxiv.org/abs/2109.10496.

[145] Sanjivani Shantaiya, Keshri Verma, and Kamal Mehta. “A survey on approaches of object detection”. In: *International Journal of Computer Applications* 65.18 (2013), pp. 975–8887.

[146] Zhengxia Zou et al. “Object Detection in 20 Years: A Survey”. In: (May 2019), pp. 1–39. arXiv: 1905.05055. URL: http://arxiv.org/abs/1905.05055.

[147] Licheng Jiao et al. “A Survey of Deep Learning-Based Object Detection”. In: *IEEE Access* 7 (2019), pp. 128837–128868. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2019.2939201. arXiv: 1907.09408. URL: https://ieeexplore.ieee.org/document/8825470/.

[148] Jeongho Shin et al. “Optical flow-based real-time object tracking using non-prior training active feature model”. In: *Real-Time Imaging* 11.3 (June 2005), pp. 204–218. ISSN: 10772014. DOI: 10.1016/j.rti.2005.03.006. URL: https://linkinghub.elsevier.com/retrieve/pii/S1077201405000215.

[149] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. “YOLOv4: Optimal Speed and Accuracy of Object Detection”. In: (Apr. 2020). arXiv: 2004.10934. URL: http://arxiv.org/abs/2004.10934.

[150] Davide Falanga et al. “PAMPC: Perception-Aware Model Predictive Control for Quadrotors”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Oct. 2018, pp. 1–8. ISBN: 978-1-5386-8094-0. DOI: 10.1109/IROS.2018.8593739. arXiv: 1804.04811. URL: https://ieeexplore.ieee.org/document/8593739/.

[151] Qiang Wang, An Zhang, and Hai Yang Sun. “MPC and SADE for UAV real-time path planning in 3D environment”. In: *Proceedings 2014 IEEE International Conference on Security, Pattern Analysis, and Cybernetics (SPAC)*. IEEE, Oct. 2014, pp. 130–133. ISBN: 978-1-4799-5353-0. DOI: 10.1109/SPAC.2014.6982672. URL: http://ieeexplore.ieee.org/document/6982672/.

[152] Manuel Castillo-Lopez et al. “A Real-Time Approach for Chance-Constrained Motion Planning With Dynamic Obstacles”. In: *IEEE Robotics and Automation Letters* 5.2 (Apr. 2020), pp. 3620–3625. ISSN: 2377-3766. DOI: 10.1109/LRA.2020.2975759. arXiv: 2001.08012. URL: https://ieeexplore.ieee.org/document/9006821/.

[153] Neville Hogan. “Impedance Control: An Approach to Manipulation”. In: *1984 American Control Conference*. IEEE, July 1984, pp. 304–313. DOI: 10.23919/ACC.1984.4788393. URL: https://ieeexplore.ieee.org/document/4788393/.

[154] E. Cataldi et al. “Impedance Control of an aerial-manipulator: Preliminary results”. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Vol. 2016-Novem. IEEE, Oct. 2016, pp. 3848–3853. ISBN: 978-1-5090-3762-9. DOI:

10.1109/IROS.2016.7759566. URL: http://ieeexplore.ieee.org/document/7759566/.

[155] Hubert Nguyen. *GPU Gems 3*. First. Addison-Wesley Professional, 2007, p. 1008. ISBN: 9780321545428.

[156] Lorenzo Stella et al. “A simple and efficient algorithm for nonlinear model predictive control”. In: *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*. Vol. 2018-Janua. IEEE, Dec. 2017, pp. 1939–1944. ISBN: 978-1-5090-2873-3. DOI: 10.1109/CDC.2017.8263933. arXiv: 1709.06487. URL: http://ieeexplore.ieee.org/document/8263933/.

[157] Max Schwenzer et al. “Review on model predictive control: an engineering perspective”. In: *The International Journal of Advanced Manufacturing Technology* 117.5-6 (Nov. 2021), pp. 1327–1349. DOI: 10.1007/s00170-021-07682-3. URL: https://link.springer.com/10.1007/s00170-021-07682-3.

[158] Takahiro Ikeda et al. “Wall contact by octo-rotor UAV with one DoF manipulator for bridge inspection”. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Vol. 2017-Septe. IEEE, Sept. 2017, pp. 5122–5127. ISBN: 978-1-5386-2682-5. DOI: 10.1109/IROS.2017.8206398. URL: http://ieeexplore.ieee.org/document/8206398/.

[159] Dimos Tzoumanikas et al. “Aerial Manipulation Using Hybrid Force and Position NMPC Applied to Aerial Writing”. In: *Robotics: Science and Systems XVI*. Robotics: Science and Systems Foundation, July 2020. ISBN: 978-0-9923747-6-1. DOI: 10.15607/RSS.2020.XVI.046. arXiv: 2006.02116. URL: http://www.roboticsproceedings.org/rss16/p046.pdf.

[160] Yon Ping Chen and Jeang Lin Chang. “Sliding-mode force control of manipulators”. In: *Proceedings of the National Science Council, Republic of China, Part A: Physical Science and Engineering* 23.2 (1999), pp. 281–288. ISSN: 02556588.

[161] Huanxin Luo, Rong Hu, and Hua Deng. “Force control of an underactuated prosthetic hand based on sliding mode with exponential reaching law”. In: *Proceedings of the 2016 International Conference on Advanced Electronic Science and Technology (AEST 2016)*. Aest. Paris, France: Atlantis Press, 2016, pp. 186–192. ISBN: 978-94-6252-257-2. DOI: `10.2991/aest-16.2016.24`. URL: `http://www.atlantis-press.com/php/paper-details.php?id=25864431`.

[162] Lorenz Meier, Dominik Honegger, and Marc Pollefeys. “PX4: A node-based multithreaded open source robotics framework for deeply embedded platforms”. In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. Vol. 2015-June. June. IEEE, May 2015, pp. 6235–6240. ISBN: 978-1-4799-6923-4. DOI: `10.1109/ICRA.2015.7140074`. URL: `http://ieeexplore.ieee.org/document/7140074/`.

[163] Anis Koubaa et al. “Micro Air Vehicle Link (MAVlink) in a Nutshell: A Survey”. In: *IEEE Access* 7 (2019), pp. 87658–87680. ISSN: 2169-3536. DOI: `10.1109/ACCESS.2019.2924410`. arXiv: `1906.10641`. URL: `https://ieeexplore.ieee.org/document/8743355/`.

[164] Open Source Robotics Foundation. *Robotic Operating system website*. 2015. URL: `http://www.ros.org`.

[165] S. Chen et al. “End-to-End UAV Simulation for Visual SLAM and Navigation”. In: (Dec. 2020). arXiv: `2012.00298`. URL: `http://arxiv.org/abs/2012.00298`.

[166] Helen Oleynikova et al. “Voxblox: Incremental 3D Euclidean Signed Distance Fields for on-board MAV planning”. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Vol. 2017-Septe. IEEE, Sept. 2017, pp. 1366–1373. ISBN: 978-1-5386-2682-5. DOI: `10.1109/IROS.2017.8202315`. arXiv: `1611.03631`. URL: `http://ieeexplore.ieee.org/document/8202315/`.

[167] Paul Kremer et al. “Cooperative localization of unmanned aerial vehicles in ROS — The Atlas node”. In: *2017 IEEE 15th International Conference on Industrial Informatics (INDIN)*. IEEE, July 2017, pp. 319–325. ISBN: 978-1-5386-0837-1. DOI: `10.1109/INDIN.2017.8104792`. URL: `http://ieeexplore.ieee.org/document/8104792/`.

[168] Luca Bergamini et al. “Deep learning-based method for vision-guided robotic grasping of unknown objects”. In: *Advanced Engineering Informatics* 44.January (Apr. 2020), p. 101052. ISSN: 14740346. DOI: 10.1016/j.aei.2020.101052. URL: https://linkinghub.elsevier.com/retrieve/pii/S1474034620300215.

[169] He Cao et al. “Unknown Object Grasping Based on Adaptive Dynamic Force Balance”. In: *Journal of Intelligent & Robotic Systems* 105.1 (May 2022), p. 10. ISSN: 0921-0296. DOI: 10.1007/s10846-021-01546-4. URL: https://link.springer.com/10.1007/s10846-021-01546-4.