Institute for Computer Science VII
Robotics and Telematics

Master's thesis

SatSim-REACSA - A Floating Platform for Simulation of Micro-Gravity With Three Degrees of Freedom

Finding and Following Optimal Trajectories With an Overactuated System

Anton Bredenbeck

February 2022

First reviewer:	Prof. Dr. Andreas Nüchter
Second reviewer:	Prof. Dr. Miguel Angel Olivares-Mendez
Advisor:	Martin Zwick & Dorit Borrmann

Acknowledgements

Many people have played a role in my being in the position that I am in today. These people have supported me in various ways, whether in the academic framework teaching and advising me or in a personal setting. I want to take this opportunity to thank those people.

First, I'd like to thank all the people involved in creating this master thesis. I want to thank the Automation & Robotics Group at the European Space Agency (ESA) for hosting me for the duration of my Master thesis and for providing all the infrastructure and hardware used in this work. Further, I'd like to thank my supervisors, Martin Zwick (ESA) and Prof. Dr. Miguel Angel Olivares Mendez (University Luxembourg), for helping with mechanical/structural problems and providing insightful feedback. I want to extend a special thank you to my friend Shubham Vyas for assisting in the experiments, numerous fruitful conversations about software, control architecture, god and the world, and the many coffees had. Thanks also to Willem Suter for assisting in the experiments and maintaining the software after my departure.

Of course, I'd also like to extend a huge thank you to my supervisors from the Julius-Maximilians-University: Dr. Dorit Borrmann and Prof. Dr. Andreas Nüchter. Thank you for supervising me during this theses, and moreover, thank you for challenging me and providing excellent opportunities for me throughout my entire academic career.

Thanks also to my colleagues and friends Jasper Zevering and Tim Lakemann for proofreading.

On the personal side, I'd like to thank my parents for their unwavering support throughout my entire studies and for enabling me to do the things I love, my wonderful girlfriend, Anja, for always supporting me and being my best friend no matter the distance. Thanks also to all the other friends who tagged along for the ride.

Thank you all very much!

Abstract

The recent increase in yearly spacecraft launches and the high number of planned launches have raised questions on maintaining accessibility to space for all interested parties. A key to sustaining the future of space-flight is the ability to service malfunctioning - and actively remove dysfunctional spacecraft from orbit. However, human missions, while proven effective in the past, carry immense cost and risk and thus call for autonomous, robotic mission concepts.

Robotic systems for satellite servicing and Active Space Debris Removal (ASDR) are the subject of ongoing research and require stringent testing and validation before launch. Testing those robotic systems by itself is already a substantial challenge as it is complicated to emulate the microgravity environment in which these systems must operate. At the cost of reducing the Degrees of Freedom (DoF) to the planar case, air-bearing platforms floating on a flat-floor can duplicate this environment at good accuracy and for reasonable test durations.

One such facility is the Orbital Robotics and GNC Lab (ORGL) at ESTEC with the European Space Agency (ESA). At this facility, a floating platform of approximately 225 kg, equipped with eight solenoid-valve-based thrusters and a Reaction-Wheel (RW) represents a realistic satellite setup. Pressurized air tanks onboard supply propellant to the thrusters and the air-bearings. To prolong test duration, it is therefore desirable to minimize the thruster usage to increase the testing time when controlling this system. The main contributions of this work are twofold: for one this work introduces a trajectory planner that finds optimal trajectories connecting two arbitrary states (Pose, Twist, and RW velocity), which minimize force exerted by the thrusters. Secondly, it proposes a trajectory tracking controller, based on a Time-Varying Linear Quadratic Regulator (TVLQR) formulation that follows arbitrary physically feasible trajectories. The trajectory tracking controller further uses a $\Sigma\Delta$-Modulator to modulate the continuous force demand onto the binary thrusters. Finally, a Kalman Filter (KF) estimates the current state, which the controller uses for feedback.

This work introduces a simulated model of the overall system that includes measurement errors and the unevenness of the flat-floor. In this simulation, the maximum average euclidean error and angular error are observed to be 16 cm and 5° respectively. Further, to assess the generalizability to different initial poses on the flat-floor a Monte-Carlo simulation is carried out. Here the controller achieves a 100% success rate of finding and following trajectories to the origin from arbitrary initial poses.

Further, this work evaluates the controller using the existing system. The trajectories are still followed well on the real system but experience a significant drop in precision compared to the simulation. The maximum average euclidean distance to the desired trajectory doubles and the average difference in orientation increases by an order of magnitude. Simple trajectories, such as straight lines, were observed to be followed better using this proposed control method, than complex ones (such as a semi-circle). The drop in performance from simulation to the real system is attributed to a lack of precise system identification and unmodelled disturbances. In addition, the RW saturates much faster than anticipated by the controller, which deprecates the pointing accuracy.

Overall this work designed and implemented the first controller of one of the heaviest floating platforms in Europe and created a software framework that will form the basis for future controller development within the ORGL.

Zusammenfassung

Die jüngste Zunahme der jährlichen Starts von Raumfahrzeugen und die große Zahl geplanter Starts haben die Frage aufgeworfen, wie der Zugang zum Weltraum für alle interessierten Kreise aufrechterhalten werden kann. Ein Schlüssel für die Zukunft der Raumfahrt ist die Fähigkeit, nicht funktionierende Raumfahrzeuge zu warten und sie aktiv aus der Umlaufbahn zu entfernen. Menschliche Missionen haben sich in der Vergangenheit zwar bewährt, sind jedoch mit immensen Kosten und Risiken verbunden, so dass autonome, robotergestützte Missionskonzepte erforderlich sind.

Robotersysteme für die Wartung und Instandhaltung von Satelliten sind Gegenstand laufender Forschungs und müssen vor dem Start streng getestet und validiert werden. Schon das Testen dieser Robotersysteme ist eine große Herausforderung, da es kompliziert ist, die Mikrogravitationsumgebung zu emulieren, in der diese Systeme arbeiten müssen. Für den Preis, dass die Degrees of Freedom (DoF) auf den planaren Fall reduziert werden, können luftgelagerte Plattformen, die auf einem ebenen Boden schweben, diese Umgebung mit guter Genauigkeit und für eine angemessene Testdauer duplizieren.

Eine solche Einrichtung ist das Orbital Robotics and GNC Lab (ORGL) am ESTEC der European Space Agency (ESA). In dieser Anlage stellt eine schwebende Plattform mit einem Gewicht von etwa 225 kg, die mit acht magnetventilbasierten Düsen und einem Reaction-Wheel (RW) ausgestattet ist, eine realistische Satellitenanordnung dar. Drucklufttanks an Bord versorgen die Triebwerke und die Luftlager mit Treibstoff. Daher ist es wünschenswert, die Nutzung der Triebwerke zu minimieren, um die Testzeit für die Steuerung dieses Systems zu erhöhen. Die Hauptbeiträge dieser Arbeit sind zweierlei: Zum einen wird ein Trajektorienplaner vorgestellt, der optimale Trajektorien zwischen zwei beliebigen Zuständen (Pose, Geschwindigkeit und RW-Drehgeschwindigkeit) findet, die die von den Düsen ausgeübte Kraft minimieren. Zweitens wird ein Trajektorienverfolgungsregler vorgeschlagen, der auf einer Time-Varying Linear Quadratic Regulator (TVLQR)-Formulierung basiert und beliebigen physikalisch machbaren Trajektorien folgt. Der Trajektorienverfolgungsregler verwendet außerdem einen $\Sigma\Delta$-Modulator, um die kontinuierliche Kraftanforderung an die binären Triebwerke zu modulieren. Schließlich schätzt ein Kalman Filter (KF) den aktuellen Zustand, den der Regler zur Rückkopplung verwendet.

In dieser Arbeit wird ein Simulation des Gesamtsystems vorgestellt, welches Messfehler und die Unebenheiten des flachen Bodens berücksichtigt. Vorerst wird die Steuerung in dieser Simulation auf verschiedenen von Hand erstellten Trajektorien evaluiert. Der Roboter folgt den Trajektorien in allen Fällen gut und erreicht für alle getesteten Trajektorien mindestens einen durchschnittlichen euklidischen Abstand zur Trajektorie von 16 cm und mindestens einen durchschnittlichen Winkelfehler von 5°. Die geringfügigen Abweichungen werden auf die Bodenunebenheiten zurückgeführt. In einer Monte-Carlo-Simulation, bei der der Roboter in einer beliebigen Ausgangsposition gestartet wird und eine Flugbahn zum Ursprung angefordert wird, erreicht der Regler eine Erfolgsquote von 100 %.

Darüber hinaus wird in dieser Arbeit die Steuerung anhand des bestehenden Systems evaluiert. Die Trajektorien werden auf dem realen System immer noch gut verfolgt, aber die Präzision nimmt im Vergleich zur Simulation deutlich ab. Sowohl der durchschnittliche euklidische Abstand zur gewünschten Trajektorie als auch die durchschnittliche Abweichung der Orientierung steigen von einigen Zentimetern/Grad auf mehrere zehn Zentimeter/Grad. Es wurde beobachtet,

dass einfache Bahnen, wie z. B. gerade Linien, mit dieser vorgeschlagenen Kontrollmethode besser verfolgt werden als komplexe Bahnen (wie z. B. ein Halbkreis). Der Leistungsabfall von der Simulation zum realen System wird auf eine ungenaue Systemidentifikation und nicht modellierte Störungen zurückgeführt. Außerdem sättigt der RW deutlich schneller als vom Regler erwartet, was die Ausrichtungsgenauigkeit beeinträchtigt.

Insgesamt wurde im Rahmen dieser Arbeit der erste Regler für eine der schwersten schwebenden Plattformen in Europa entworfen und implementiert und ein Software-Framework geschaffen, das die Grundlage für die künftige Reglerentwicklung innerhalb des ORGL bilden wird.

Contents

List of Figures

List of Tables

Acronyms

ACROBAT Air Cushion Robotic Platform.

ASDR Active Space Debris Removal.

AWGN Additive White Gaussian Noise.

BLDC Brushless Direct-Current.

DoF Degrees of Freedom.

DRE Differential Riccati Equation.

ESA European Space Agency.

GEO Geostationary Orbit.

GNC Guidance, Navigation and Control.

HIL hardware-in-the-loop.

HJB Hamilton-Jacobi-Bellman.

IMU Inertial Measurement Unit.

KF Kalman Filter.

LEO Low Earth Orbit.

LQR Linear Quadratic Regulator.

MINLP Mixed-Integer Non-Linear Programming.

MoCap Motion-Capture.

MoI Moment of Inertia.

MPC Model Predictive Control.

NLP Non-Linear Programming.

OBC On-Board Computer.

ORGL Orbital Robotics and GNC Lab.

PWM Pulse-Width-Modulation.

PWPF Pulse-Width Pulse-Frequency.

RECAP Reaction Control Autonomy Platform.

RW Reaction-Wheel.

SATSIM Satellite Simulator.

SDF Simulation Description Format.

SO(2) Special-Orthogonal Group of Order 2.

TVLQR Time-Varying Linear Quadratic Regulator.

Chapter 1

Introduction

Space debris is widely recognized as a significant challenge to all future activities in space [20–25]. Already in 1978, Kessler et al. [26] noted that the debris forming from satellite collision could lead to an artificial asteroid belt in popular orbits such as the Low Earth Orbit (LEO) and Geostationary Orbit (GEO). This belt could severely limit human space activities. In the extreme case, it will entirely prohibit the possibility for safe spaceflight in desirable orbits. Becoming known as the "Kessler-Syndrome", the described effect gained traction in recent years for multiple reasons. Firstly, today a significant portion of debris is caused by human activity. Some of the biggest contributions to human-made space debris are the following events:

- The 2007 anti-satellite missile test by the Chinese, where they destroyed one of their weather satellites producing about 3500 particles within 1 cm to 10 cm [27]
- The 2009 collision of the Iridium 33 and Cosmos 2251 that produces about 2300 pieces in the same range [28]
- The very recent, Russian anti-satellite test on 15th of November 2021 [29]. First approximations put the number of large debris pieces at 1500.

Planned large constellations, such as SpaceX Starlink, are suspected to strongly influence the space debris situation in the future, possibly endangering space sustainability, as indicated by Bastida Virgili et al. [30]. Despite there being stringent requirements on de-orbiting satellites, many members of the community (cf. [31–33]) argue that these are not sufficient to avoid the Kessler-Syndrome mentioned above and campaign for Active Space Debris Removal (ASDR).

Although being a relevant topic with major interest from academic circles and industry, a definitive solution does not exist. At the end of 2019, the European space agency (ESA) commissioned the world's first mission to remove an object from space: Clearspace-1. At the time of writing, ESA plans the launch for 2025. Given the current progress, a lot of challenges remain unsolved. Among those are the development of a versatile capture system, advanced Guidance, Navigation and Control (GNC) and target attitude estimation, and advanced sensors for ranging [34].

Very similar challenges are still open for the problem of on-orbit servicing of satellites. While also reducing space debris by extending the lifetime of satellites, the same factor also makes spacecraft launches more financially viable, which justifies the interest by the space industry.

Given the proximity of the servicing/capturing device and the respective client, the missions also carry above-average risk. Hence, the flawless operation must be guaranteed as good as possible prior to launch. For this purpose, ground-test facilities that provide system-level evaluation are necessary. The servicing/capturing system will eventually operate in a zero-g environment, while all ground facilities operate within the earth's gravity. Hence, precisely replicating the entire operating domain proves very difficult. Specific facilities are needed to simulate the microgravity environment in which the system operates.

Common approaches for simulating micro-gravity are air-bearing platforms, cable suspensions, neutral buoyancy, free-fall, magnetic suspensions, large rotating wheels, and hardware-in-the-loop (HIL) simulations [35]. Cable suspensions can only provide gravity compensation along one axis and are therefore unsuitable for robotic testing. Neutral buoyancy in water, while being immensely helpful in astronaut training, does not provide a great testing ground for robots since the viscous damping of the fluid is a very significant deviation from the actual operating environment. More practically, many robots cannot be submerged in water as it would damage the components and neutral buoyancy in air requires huge, impractical gas reservoirs. Free-fall testing, such as during a parabolic flight, provides micro-gravity in three dimensions which otherwise is hard to replicate; however, it only allows for periods of approximately 30 s [36] which is too short for most robotic testing. Lastly, magnetic suspensions are only able to support small forces and torque, hence cannot provide testing at the whole system level, and HIL simulations are only as good as the simulation used as the base. Since there are always model inaccuracies, it is difficult to guarantee the correctness of the simulation. As the last suitable option, air-bearing platforms have turned out to be the most popular type of testing facility in academia and industry [37].

One of these facilities is the Orbital Robotics and GNC Lab (ORGL) at ESTEC, depicted in Figure 1.1. Its main component is a flat surface on which a payload can float using air bearings and hence has three Degrees of Freedom (DoF); two translational and one rotational DoF. This flat floor enables testing of systems in micro-gravity for a longer duration at the cost of reducing the system to a lower number of DoF. One of the testing platforms provides a realistic actuator assembly similar to those found on a satellite. It is equipped with several cold gas thrusters and a Reaction-Wheel (RW). It is interesting to control the system along some desired trajectory to a final state for testing purposes. Among other things, these trajectories may represent the motion of a capturing target or a secondary satellite to which a primary satellite must maintain a relative pose. The deciding factor for the test duration is the amount of gold gas stored onboard, which is the propellant to keep the platform floating. However, the same tanks also provide propellant to the thrusters that control the system's position and orientation. Therefore, a controller that prolongs the tests, and consequently the operational lifetime in space, must use the thrusters propellant-optimally along some trajectory from any arbitrary starting point to some final destination.

This thesis proposes a controller that first finds optimal trajectories between two points in state-space and then uses traditional optimal control methods to follow the desired trajectory. For this, optimality is defined with respect to a cost function specified by the user, e.g., minimal propellant consumption.

The main contributions of this work are twofold: first, this work introduces a trajectory planner that finds optimal trajectories that connect two arbitrary states while minimizing the

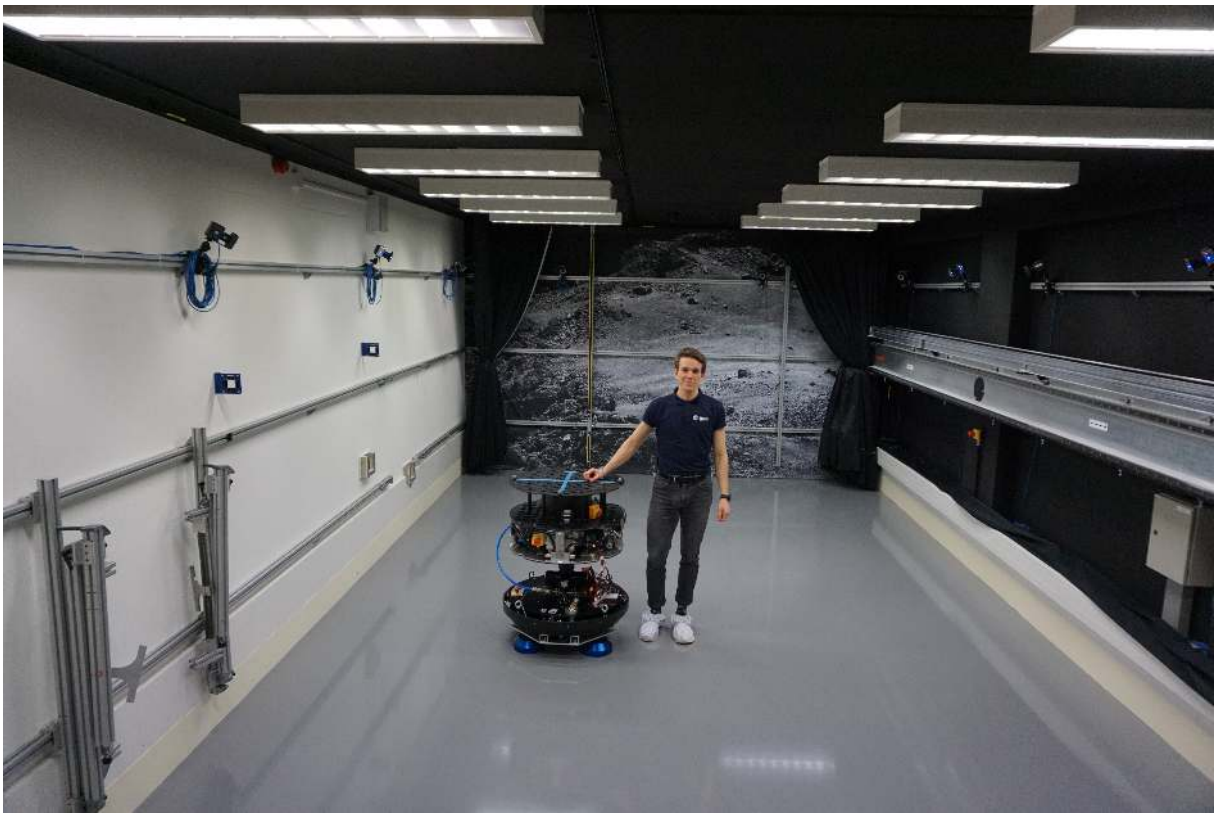


Figure 1.1: The author with the testing platform on the flat-floor of the ORGL at ESTEC. A flat ground on which an air-bearing platform can float. The testing platform consists of three subsystems: the floating base ACROBAT, the thruster and tank assembly SATSIM, and the RW REACSA, which are stacked onto each other. The flat-floor is within the field-of-view of a Motion-Capture (MoCap) system that provides pose measurements to some previously defined world reference frame.

force exerted by the thrusters. Secondly, it proposes a trajectory following controller that follows the above-mentioned optimal and all other physically feasible trajectories.

The rest of the document is structured as follows: Chapter 2 gives an overview of some facilities similar to the ORGL and common control approaches for satellite pose control. After introducing the system model, including the steps taken for system identification in Chapter 3 the control architecture is introduced by Chapter 4. Finally, an evaluation in simulation (Chapter 5) and on the real system (Chapter 6) are carried out before giving an overall summary and conclusion in Chapter 7.

Chapter 2

State of the Art

Given the interest of academia and industry in orbital robotics, numerous labs worldwide investigating free-floating robotics exist. This Chapter introduces the relevant work on two categories: air-bearing-based test facilities and control algorithms for free-floating robotic systems.

2.1 Air-bearing-based Test Facilities

Air-bearing-based testing platforms are widely used in spacecraft testing and development. Even in the very early stages of space technology these platforms were used for validation of spacecraft attitude control systems. Already in 1967, the National Aeronautics and Space Administration (NASA) had developed a system capable of two-dimensional, free floating movement of a payload up to approximately 90 kg. Ever since those systems have become significantly more advanced, in particular in terms of control and autonomy [35]. Among the earliest successfully used systems for robotic applications in space is the Experimental Free-Floating Robot Satellite simulator (EFFORTS) at the Tohoku University. By demonstrating the dynamics and control of the later successfully flown ETS-VII robotic arm it led the way for many systems that followed [38].

A description of some of today's "State-Of-The-Art" facilities and previous installations follows in this Section.

2.1.1 Air Bearing Floor NASA

The Air Bearing Floor (ABF) at the Johnson Space Center in Houston is an approximately $21\,\mathrm{m} \times 30\,\mathrm{m}$ epoxy floor, making it the largest of its kind. It is manufactured to such a precision that the floor has an average deviation of $6.35\mathrm{e}{-3}\,\mathrm{mm}$ and is level up to $0.25\,\frac{\mathrm{mm}}{\mathrm{m}}$. Platforms on the floor are usually operated via an air-hose but can also be operated with tanks. The platform is shown in Figure 2.1. Its size and precision make it one of the pristine facilities in the world enabling vast testing capabilities [39].

2.1.2 Flight Robotics Laboratory (FRL) Marshall Space Flight Center

A second facility used by NASA is the Flight Robotics Laboratory (FRL) at the Marshall Space Flight Center [40]. Similar to the ABF it has a large epoxy floor ($13.41\,\mathrm{m} \times 26.21\,\mathrm{m}$) on which

Figure 2.1: Left: the Air Bearing Platform at the Johnson Space Center in Houston. Courtesy of NASA. The black area is the epoxy flat-floor. In the front, the image depicts a human-operated hovering platform [1]. Right: The Formation Control Testbed (FCT) at JPL [2].

three Degrees of Freedom (DoF) free-floating motion is achieved using air-bearings. In addition to the two air-bearing platforms, one 8-DoF overhead gantry provides a 225 kg payload capability for simulating relative motion with respect to a fixed target on the facility floor.

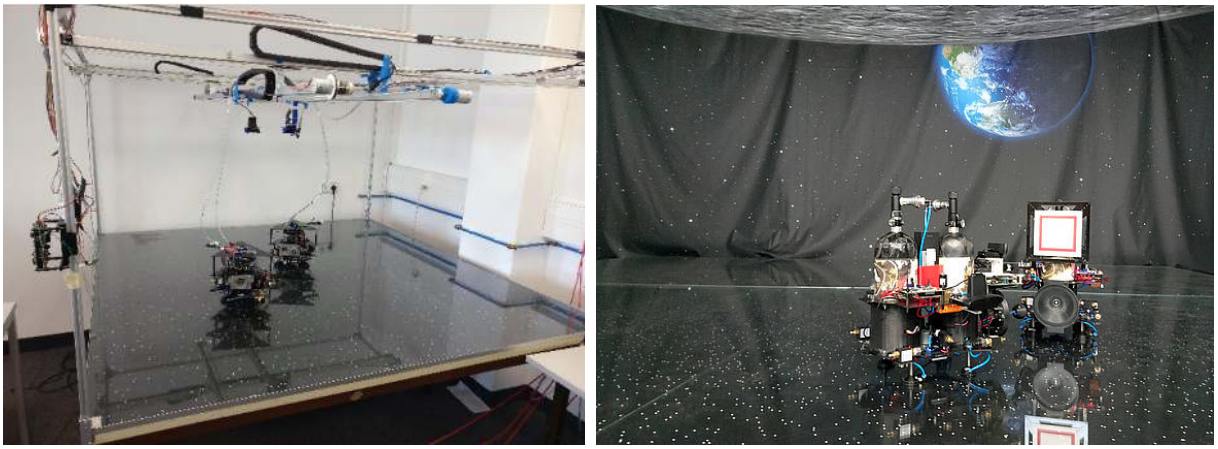
2.1.3 Formation Control Testbed (FCT) Jet Propulsion Laboratory

Another NASA facility is located at the Jet Propulsion Laboratory. The Formation Control Testbed (FCT) is mostly used for Guidance, Navigation and Control (GNC) development and testing [41]. It uses two air-bearing platforms on a flat-floor ($7.3\,\mathrm{m} \times 8.5\,\mathrm{m}$) as depicted in figure 2.1. Mounted on an erected platform is a hemispherical air bearing such that the overall system provides six DoF. For actuation, the system is equipped with 16 thrusters, a Reaction-Wheel (RW), inertial sensors, and processing capabilities. However, an additional payload is not anticipated. Further, the flatness of the floor is in the order of $(150 \pm 50)\,\mu\mathrm{rad}$ of maximal slope.

2.1.4 Air Bearing Based Platforms Würzburg

At the chair of Aerospace Computer Science at the Julius-Maximilians University Würzburg two different air bearing based testing platforms are developed.

The smaller of the two is a $2\,\mathrm{m} \times 2\,\mathrm{m}$ glass plate on top of a table surrounded by a mechanical structure as shown in Figure 2.2a. The enclosing mechanical structure contains two bi-axial steering systems which both carry a camera and an air-supply connector. This enables two agents to be on the platform at the same time, both fed with supply air from directly above to minimize disturbing forces from the umbilical. Steering is done by the mentioned camera and a simple visual servoing algorithm. The main advantage of this system is therefore the independence of an air tank, thus enabling unlimited testing duration. Further, since the agents do not require onboard containers they can be designed rather small and therefore use the limited size of the facility to its fullest. Each agent uses three pressure release nozzles arranged in a triangular constellation as steering thruster actuators and one RW to control its orientation



(a) **(b)**

Figure 2.2: The two air bearing based testing platforms at the chair of aerospace computer science at the University of Würzburg. The smaller robotic platform with the enclosing mechanical tracking structure (left) and the larger (right). Courtesy of Redah [3, 4]

while being mounted on a flat circular air bearing. The agents carry a camera and an onboard computer which is directed at the photo underneath the glass. This represents a star catalog that is equivalent to what an actual satellite would be measuring in orbit. Therefore, attitude determination and control based on start-trackers can be tested on this platform [4].

The larger of the two, which is seen in Figure 2.2b, is constructed in a similar way with a large glass plate of size $5.40\,\mathrm{m} \times 4.40\,\mathrm{m}$. The main difference is that no mechanical structure is implemented but the system is based on agents carrying tanks of pressurized air. The agents for this platform carry the same actuators and sensors as the smaller platforms but are have a bigger footprint to fit the air pressurized air containers [3].

Overall the system provides versatile testing capabilities, in particular the possibility for tests over a longer time frame with smaller objects the constant air-supply system is very intriguing. However, the choice of glass does bring its limitations with respect to the flatness of the overall area as well as the maximum payload weight of systems tested. Especially, for heavy payloads the glass might flex with the supporting structure underneath, causing local minima which would then cause deviations from the desired floating behavior. In a worst-case scenario, the glass might even be damaged by the weight of the payload. Further, the size of the area (small and large system) limits the maneuvers that can be executed.

2.1.5 Zero-G Lab Luxembourg

The Zero-G Lab, located at the University of Luxembourg is one of the most recent flat-floor facilities. As depicted in figure 2.3, it consists of a $3\,\mathrm{m} \times 5\,\mathrm{m}$ flat-floor, a floating platform, and two robotic arms mounted on rails. Using a Motion-Capture (MoCap) system ground truth for pose measurements is provided [5, 42].

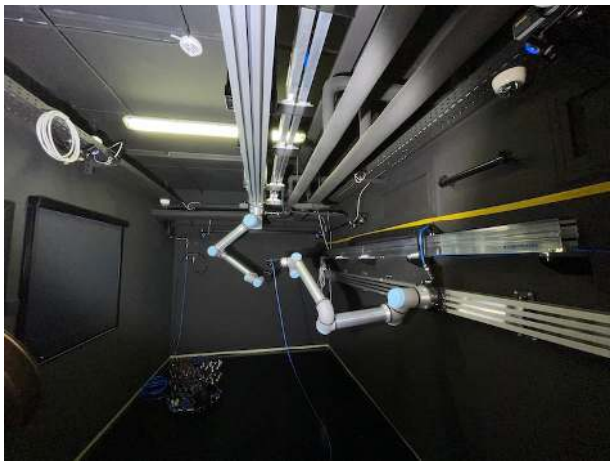


Figure 2.3: The Zero-G -Lab at the University of Luxembourg courtesy of SpaceR [5]

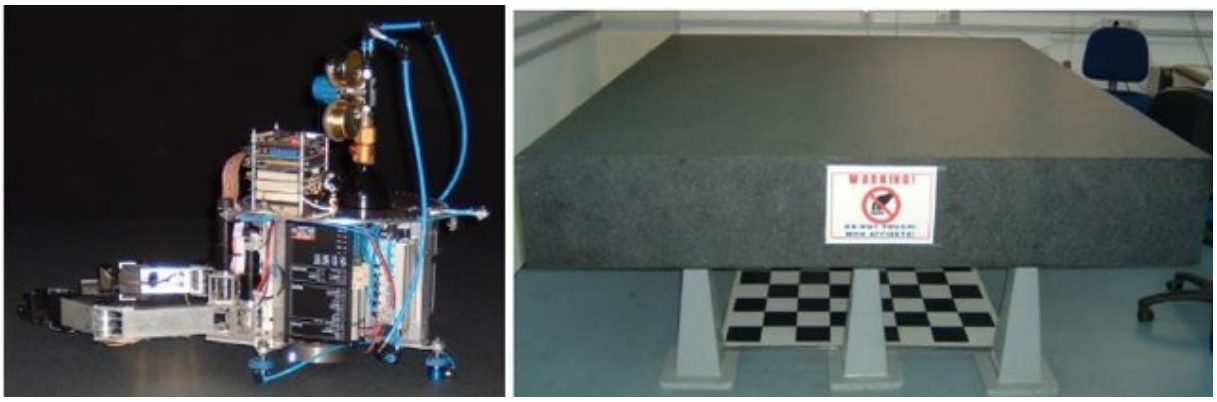


Figure 2.4: The test-bed at NTUA. The floating platform (left) and the granite table (right). Courtesy of Papadopoulos et al. [6].

2.1.6 The NTUA Space Robot Simulator

At the National Technical University of Athens (NTUA) Papadopoulos et al. developed a planar test-bed already in 2008 [6]. In particular, they develope a combination of a hard-and a software simulation. While the software simulation solved the differential equations of system dynamics and visualized them, the hardware is used to validate previous simulations. For the hardware setup, Papadopoulos et al. choose a granite table ($2.2\,\mathrm{m} \times 1.8\,\mathrm{m}$) that fulfills the required characteristics (maximum slope of $0.17\,\frac{\mathrm{mm}}{\mathrm{m}}$) and a floating platform. Figure 2.4 shows the table and the floating platform

The floating platform is equipped with three pairs of counter-facing thrusters as well as a reaction wheel. The air-bearings that enable the floating behavior are operated using the onboard CO_2 tanks which also propellant the thrusters. Control of thrust is done using on-off solenoid valves that are controlled via an On-Board Computer (OBC). In particular, the authors operate the thrusters using Pulse-Width-Modulation (PWM), which allows the thrusters to

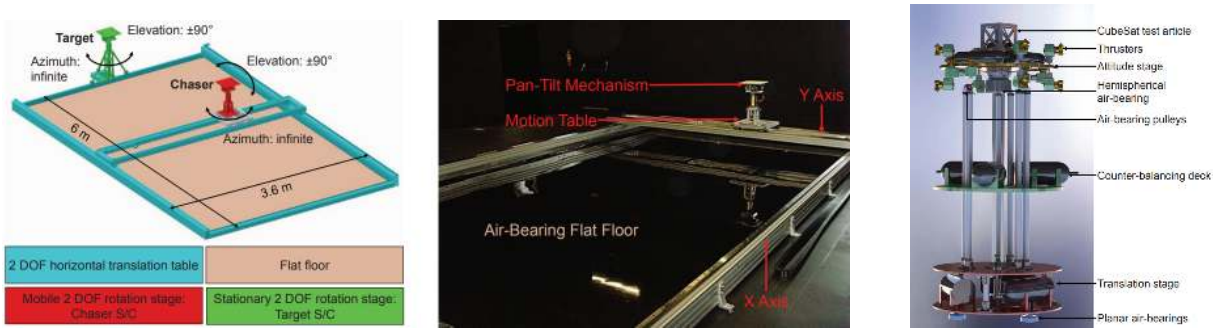


Figure 2.5: Left: The ORION test-bed. Right: DAWN-ABV, the floating platform. Courtesy of Wilde et al. [7].

provide different effective thrusts. Hereby, the authors optimize the PWM frequency to have minimal dead-band and saturation regions. Further, the authors study the tradeoff between activating the thrusters and the reaction-wheel with respect to performance and propellant efficiency. They propose a non-general approach of two different operating modes. One is more propellant efficient, since no thrusters counteract each other, while the second achieves the desired position significantly faster while being also less propellant efficient. For position control, the pose of the system is estimated using a camera mounted above the table, that tracks three LEDs attached to the floating platform.

The authors show that the simulation is representative of the hardware and especially both are representative of a free-floating system in space. Nonetheless, the small area of the table significantly limits the size and weight of possible payloads that can be tested.

2.1.7 ORION at the Florida Institute of Technology

The Orbital Robotic Interaction, On-orbit servicing, and Navigation (ORION) at the Florida Institute of Technology is one of the more recently developed flat-floor facilities [7]. In contrast to many of the other facilities, it contains multiple different mechanisms to simulate motion. For one it entails a stationary "target" mounted on a pan-tilt unit that can be used for approach simulations. Further, the classic 6 m × 3.6 m epoxy flat-floor is surrounded by a rail system providing translational motion to another pan-tilt unit powered by stepper motors, thus providing a possibly moving target with four DoF. And lastly, an air-bearing-based floating platform allows for motion on the flat-floor. As shown in Figure 2.5 this allows a system under test to aim for the target which moves at a precisely defined velocity.

The accompanying floating platform developed (as shown on the right in Figure 2.5) is an erected hemispherical air-bearing mounted on top planar air bearings. Additionally, vertical linear motion is enabled by having the attitude stage on the hemispherical air-bearing counter-balanced through a set of air-bearing pulleys, thus providing full six DoF (whereby some degrees are limited to maximal and minimal values). The payload capabilities are 10 kg and the test duration is approximately 30 min. The lab is further equipped with a sunlight simulator and obtains ground truth measurements from a MoCap system. Overall this is one of the most varied facilities in its capabilities providing a very solid foundation for development and validation of

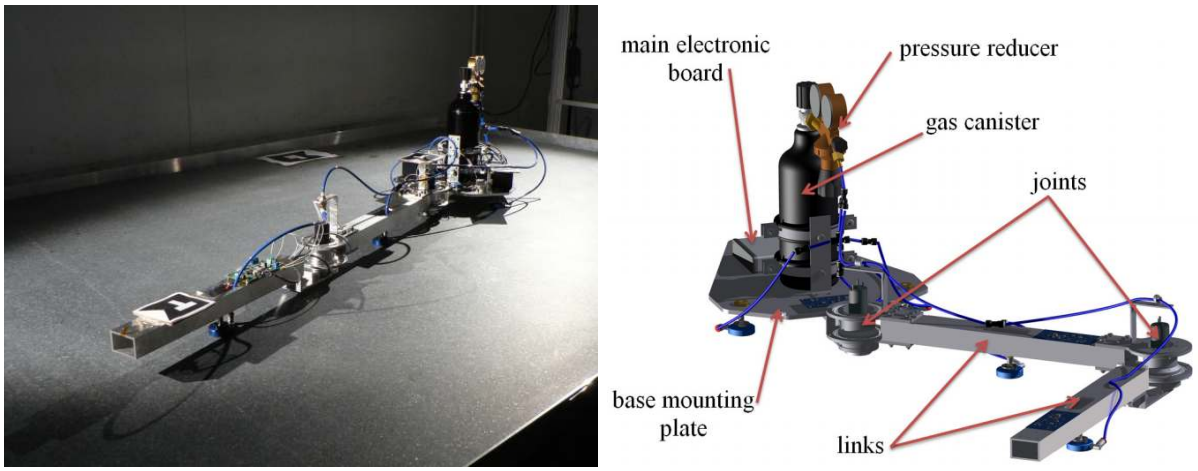


Figure 2.6: The Planar Air-Bearing Microgravity Simulator at CPK-PAN. Photo of the setup (left) and schematic (right); courtesy of Rybus et al. [8].

rendezvous, proximity operations, and capture of spacecraft at small scales.

2.1.8 Planar Air-Bearing Microgravity Simulator CPK-PAN

The planar test-bed at the Space Research Centre of the Polish Academy of Sciences (CPK-PAN) is presented in [8]. Especially with space manipulators in mind, the system consists of a two DoF manipulator which is mounted on a platform supported by air-bearings. Further, all links are also supported by air-bearings allowing them two move friction-less over the flat surface as well. As a flat surface, the authors chose a granite table, which is $2\,\mathrm{m} \times 3\,\mathrm{m}$ in size, for its low roughness and flatness properties, exhibiting a maximal slope of $1\,\frac{\mathrm{mm}}{\mathrm{m}}$. Figure 2.6 shows the setup of the system.

The floating platform uses pressurized air stored in the onboard tank which is distributed along the links to each individual air-bearing to maintain the frictionless motion. For pose estimation the authors propose a visual system that localizes predefined markers along the robot arm to determine the full state.

In their evaluation, the overall system works very well demonstrating a free-floating robotic system. The dynamics of the base are shown to be representative of that of a similar system in space. However, the lack of actuators on the base does not allow for full simulation of a controlled base including a robotic arm.

2.1.9 Test Environment for Applications of Multiple Spacecraft (TEAMS)

At the DLR Bremen a 5 DoF test facility exists [9, 43]. They achieve 5 DoF by mounting a spherical air bearing on top of a floating platform that moves on one of two $4\,\mathrm{m} \times 2.5\,\mathrm{m}$ granite tables, which are each leveled to within less than $0.01\,\frac{\mathrm{mm}}{\mathrm{m}}$. Each body on top of the spherical air-bearings is equipped with common satellite sensors and actuators, i.e. Inertial Measurement Unit (IMU), RWs, cold gas thrusters, and an on-board computer for control.

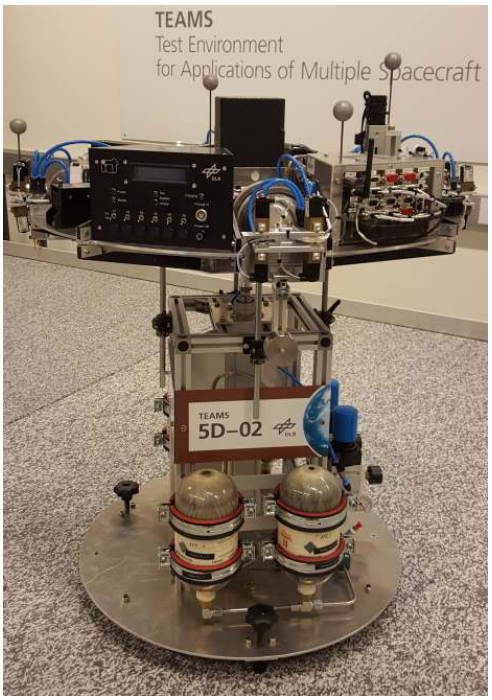


Figure 2.7: The TEAMS vehicle at DLR Bremen. Courtesy of Wehrman et al. [9].

Overall the test facility provides wide testing possibilities for algorithms in all aspects of GNC. However, the system is not designed to carry additional payloads and thus cannot be used to test different hardware components and actuator configurations.

2.1.10 Orbital Robotics and GNC Lab ESTEC

The Orbital Robotics and GNC Lab (ORGL) at ESTEC consists of two main parts: A flat floor for two dimensional free-floating systems and a pair of robotic arms mounted on rails along the side and the roof of the lab. While the robotic arms provide possibilities for evaluation of GNC payloads on various trajectories, the flat floor provides test grounds for multiple air-bearing-based floating platforms. The floor is 4.75 m × 8.78 m in size and has a maximum deviation of 0.7 mm in height [11]. It lies entirely within the field of view of a MoCap system that provides pose measurements. Multiple floating platforms for free-floating dynamics are present in the ORGL. Two of which are Mecanum wheel-based robotic platforms that carry a small floating plate themselves. Controlling the platform to keep the floating plate at its center allows for free-floating movement in a large area for small payloads. The other two platforms are air-bearing platforms for the flat-floor. The smaller of the two, MANTIS (Maneuverable Testbed for In-orbit Simulation), can carry approximately 50 kg [11]. The larger of the two, ACROBAT (Air Cushion Robotic Platform), has a payload capacity of 121 kg [10]. Figure 2.8 shows all the test platforms.

For ACROBAT two upgrade stacks are available. The first, called SatSim (SATellite SIMulator) weighs 45 kg and provides tanks for 4 kg of 300 bar compressed breathing air and eight

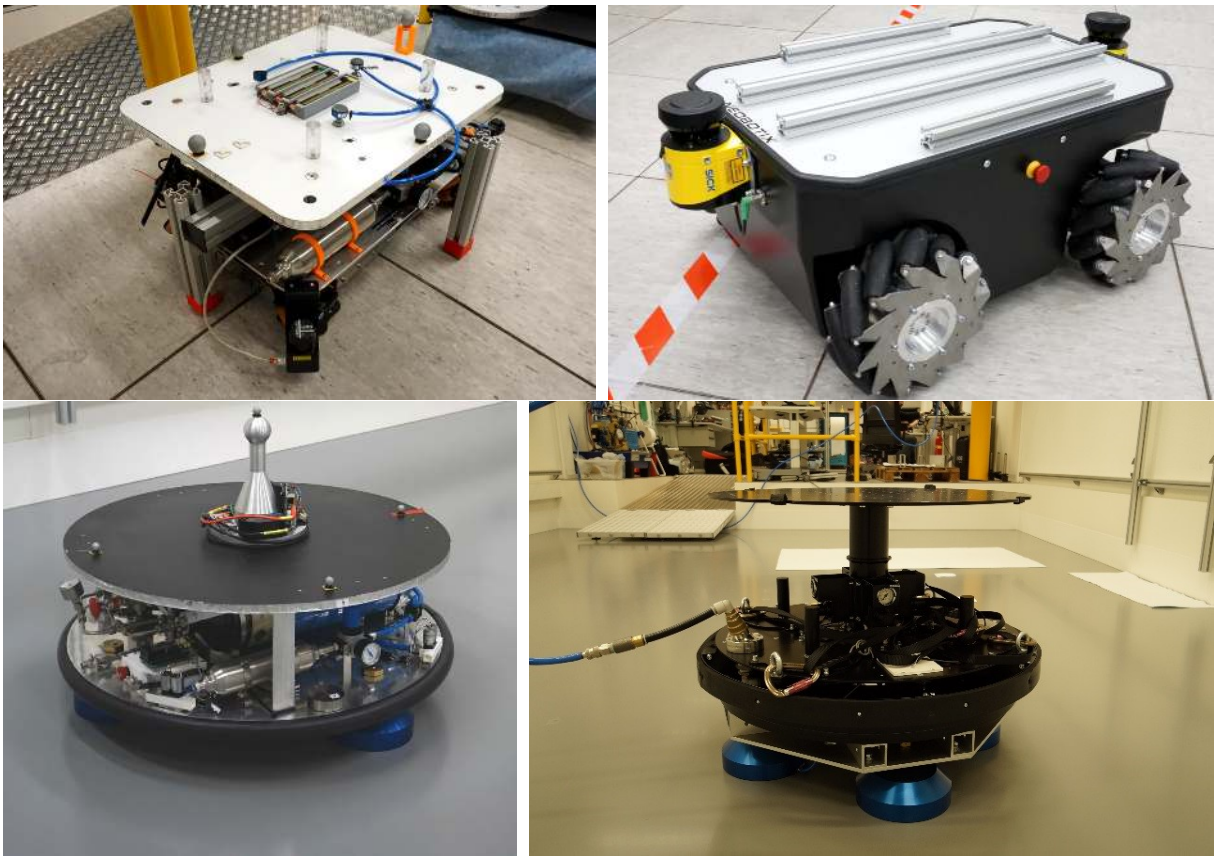


Figure 2.8: The test platforms for free-floating dynamics at the ORGL. Top: the two Mecanum-wheel-based platforms. Bottom: The smaller air-bearing platform MANTIS on the left, and the larger floating platform ACROBAT on the right. Courtesy of Zwick and Kolvenbach [10, 11]

thrusters operating at a regulated lower pressure [10]. The second stack, called RECAP (Reaction Control Autonomy Platform), weighs 17.67 kg and adds an RW to the overall system.

The combination of ACROBAT and both upgrade stacks is the central system used for this work. Before this work, there was no coordinated controller that allowed for position and orientation control of the entire system.

2.1.11 Summary

The above-summarized laboratories represent some of the current state-of-the-art air-bearing base testing platforms. Table 2.1 summarizes their properties. All of them use pressurized gas to provide free-floating dynamics, while they differ in overall DoF simulated, actuators provided, and payload capacity. In particular, the ORGL is the only platform in Europe that provides additional payload capabilities while also providing a realistic actuator assembly.

The goal of this work is to provide a control architecture for this system. Testing new technologies mounted on the controlled platform allows the system under test to not take the base platform trajectory control into account, yielding faster development times.

Laboratory	Size	Flatness	Realistic Actuator Assembly	Additional Payload
ABF	21 m × 30 m	0.25 $\frac{\text{mm}}{\text{m}}$	Used for testing of full systems - Yes	Yes
FRL	13.41 m × 26.21 m	–	Used for testing of full systems - Yes	Yes
FCT	7.3 m × 8.5 m	0.15 $\frac{\text{mm}}{\text{m}}$	Two fully equipped floating platforms - Yes	No
Würzburg	2 m × 2 m 5.4 m × 4.4 m	–	Multiple small fully equipped platforms - Yes	No
Zero-G Lab	3 m × 5 m	–	No	No
NTUA	2.2 m × 1.8 m	0.17 $\frac{\text{mm}}{\text{m}}$	One small platform - Yes	No
ORION	6 m × 3.6 m	3.5 $\frac{\text{mm}}{\text{m}}$	Multiple platforms - Yes	Yes, either floating or on the additional gantry
CPK-PAN	2 m × 3 m	1 $\frac{\text{mm}}{\text{m}}$	One satellite mock-up and robotic manipulator	No
TEAMS	4 m × 2.5 m	0.01 $\frac{\text{mm}}{\text{m}}$	Two satellite mock-ups	No
ORGL	4.75 m × 8.78 m	0.7 $\frac{\text{mm}}{\text{m}}$	Multiple floating mock-ups and Mecanum-wheel based platforms	Yes

Table 2.1: A comparison in size, flatness, and payload capabilities of the previously introduced laboratories using air-bearing based platforms. A cell not being filled in implies this information is not publicly available.

2.2 Position and Attitude Control

The systems floating on top of the previously mentioned flat-floors are equipped with different actuators and abide by different control laws. In general, different control approaches must be considered, given the respective constraints and desired performance of the given system. The following section gives an overview of some recent control approaches that are used for control of spacecraft and spacecraft-like systems.

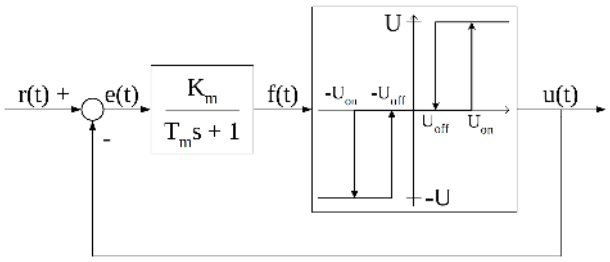


Figure 2.9: Block diagram of the PWPF-modulator. The current output u is subtracted from the continuous input r and the error e is accumulated using a first-order filter. The accumulated signal is fed to a Schmitt trigger. A Schmitt trigger is a comparator with hysteresis, i.e. once a higher threshold (U_{on}) is reached the output is set to the highest value and once the input falls below a lower threshold (U_{off}) it is set to the lowest possible value. In between the thresholds it maintains its current value. Courtesy of Krovel [12].

2.2.1 Traditional Control Methods

In traditional control methods for spacecraft often times computational simplicity and robustness are prioritized. Generally, the control laws are designed independently of the actuators in terms of desired control torques and forces acting on the body frame. Once those are found, the respective actuators are activated to best match these control torques and forces. The most common actuators are RWs and thrusters. The classic introduction to spacecraft control [44] proposes a feedback control law in form of a proportional-differential control-law

$$\mathbf{u}(t) = k_p \mathbf{e}(t) + k_d \dot{\mathbf{e}}(t) \tag{2.1}$$

where $\mathbf{e}(t)$ is the error between current and set value,k_p and k_d are the proportional and differential gains, and $\mathbf{u}(t)$ is the resulting control value. For torques and RWs, the resulting value can directly be realized on the actuator. If thrusters are used, the desired continuous value must be modulated onto the thrusters which are usually binary. This means they only allow for the states on and off. A popular technique for this is Pulse-Width Pulse-Frequency (PWPF) modulation. It uses dynamic on and off times of the thrusters to match the average force demanded. Figure 2.9 shows the block diagram of this modulator. The controller gains are then tuned heuristically to fulfill the system requirements.

The main advantage of this system is its simplicity. This allows for straightforward stability guarantees and uncomplicated realization in hard-and software, even with limited computing power.

However, these traditional control methods often rely on heuristics for gain tuning and cannot provide guarantees of optimality. Nowadays, the computing power of satellites has significantly increased [45] such that more complex control strategies are feasible. Thus, it is feasible to minimize some operator-specified cost functions, guaranteeing optimality explicitly.

2.2.2 Control for Continuous Actuators

Next to thrusters and RWs, other actuators that are commonly used in satellite attitude control are momentum wheels and magnet-torquers [46, 47]. Momentum wheels are similar to reaction

wheels in that they use the inertia of a spinning mass. However, they only spin at some constant velocity to provide gyroscopic stability along the rotation axis. Thus, they only provide a passive stabilization of the spacecraft. Magnet-torquers, on the other hand, are multiaxial, similar to some RW arrangements. Basically, they are electromagnets that interact with the Earth's magnetic field to provide torque onto the spacecraft orthogonal to the field lines. This implies that one rotational axis of the spacecraft – the axis aligned with the field lines – cannot be controlled. Nonetheless, they are prevalent for small satellites such as CubeSats (small satellites that fit within multiples of a standardized $10\,\text{cm} \times 10\,\text{cm} \times 10\,\text{cm}$ cube), as their size and power requirements fit into the limited frame.

In the mission MOVE-II [48] – a CubeSat developed by a student consortium at the Technical University Munich – the only actuator on board is a magnet-torquer. The authors of [48] introduce an "Extended Linear Quadratic Regulator (LQR)" control approach. Hereby they linearise the system about two different operating points, one in positive rotational velocity and one in negative. They employ a switching strategy subjected to hysteresis to switch to the respective controller depending on the current state. Overall they are able to achieve high pointing accuracy of fewer than five degrees using only magnet-torquers.

The LQR is a great way to utilize increased computing power on modern satellites and also doesn't rely on heuristic gain tuning as it explicitly minimizes a cost function. However, the classic LQR can only be used to control the system to the desired set-point and doesn't handle trajectory tracking without additional modifications. Moreover, it doesn't take actuator limitations into account.

Therefore, for this work optimal control methods provide a promising approach. However, they should be augmented to add capabilities to follow trajectories and take the actuator and state constraints explicitly into account.

Chapter 3

System Model

The platform this work deals with, is the combination of three previously existing platforms: Air Cushion Robotic Platform (ACROBAT), Satellite Simulator (SATSIM), and Reaction Control Autonomy Platform (RECAP). The platforms are a stack of modular components that provide different functionality:

- ACROBAT provides the base of the platform. Using three air bearings it enables the micro-gravity behavior on the flat ground.
- SATSIM provides tanks with air that can be used for the ACROBAT air bearing as well as the thrusters that SATSIM uses to provide thrust to the stack.
- RECAP provides a reaction wheel that can be used for yaw control.

The three systems and the proposed configuration are shown in Figures 3.2 and 3.1. Their mass and size properties are shown in Table 3.1. In this Chapter, the overall system, consisting of the three modules above is characterized. In particular, this chapter derives a model for the thrust by an individual thruster, the torque provided by the Reaction-Wheel (RW), and the motion model, given those forces and torques.

Subsystem	Mass	Height	Radius
ACROBAT	154 kg	62.5 cm	35 cm
SATSIM	50 kg	20 cm	35 cm
RECAP	17.67 kg	20 cm	35 cm
Σ	221.67 kg	102.5 cm	–

Table 3.1: Mass and size properties of the subsystems and the overall sum

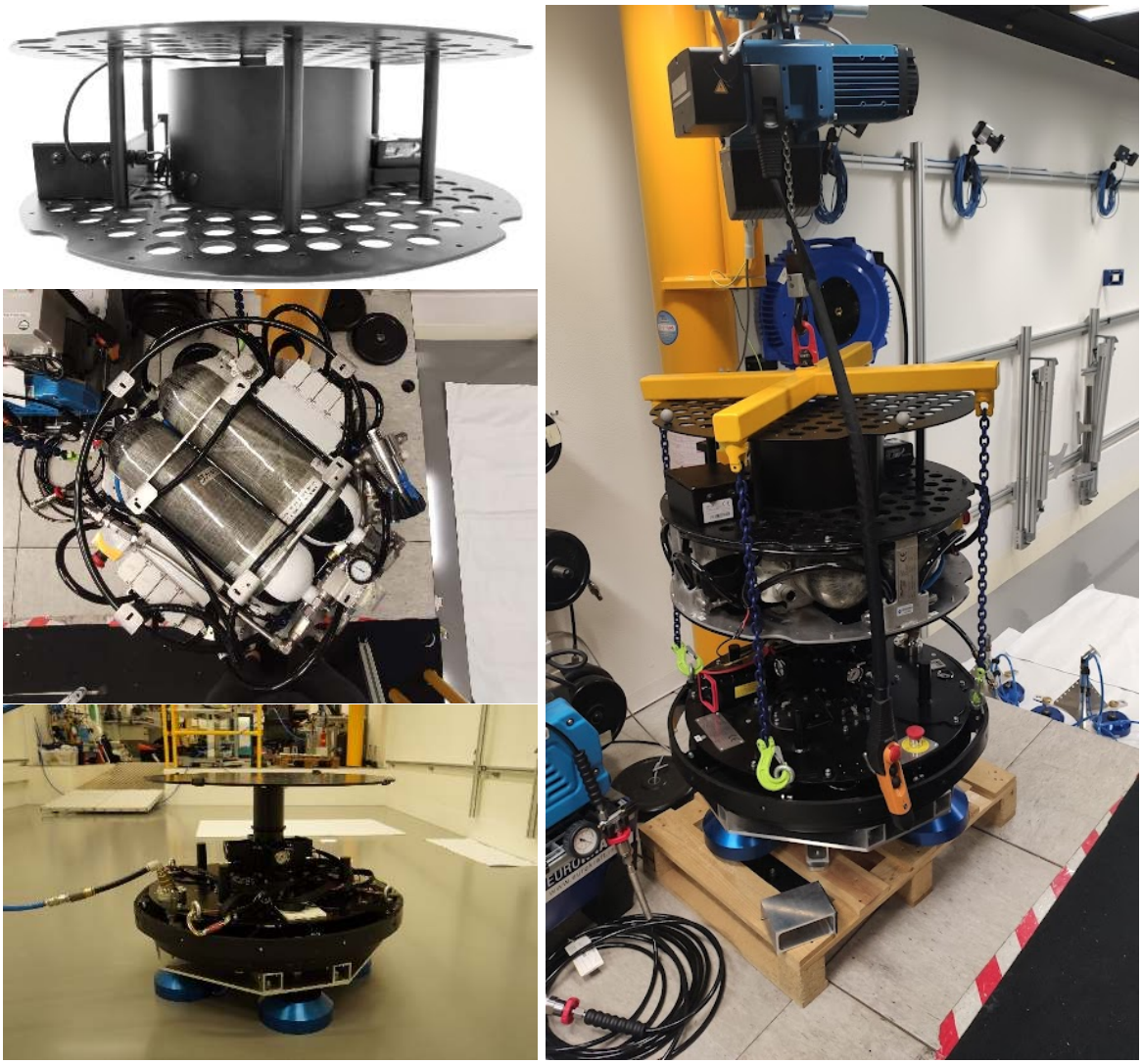


Figure 3.1: Images of the subsystems: Top Left: RECAP, Middle Left: SATSIM without its top plate, Bottom Left: ACROBAT Right: All components mounted onto each other.

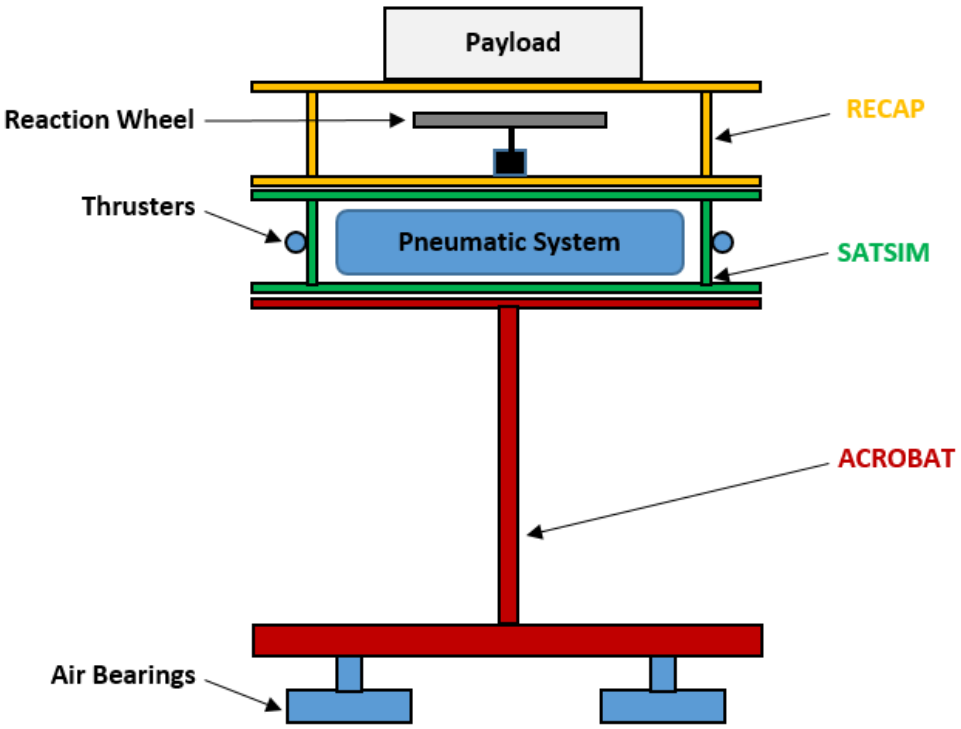


Figure 3.2: A schematic of the proposed platform setup to form the overall system. Each platform is connected via the module interface plate. All systems are connected to the same power supply and microcontroller.

3.1 Thruster Model

The propulsion system SATSIM carries four counter-facing pairs of thrusters that are arranged symmetrically around the circumference of the platform. To model the effect the actuators have on the overall system the forces they produce must be determined. This section, derives the nominal force an individual thruster produces when used. Eventually, the thrusters are assumed to behave ideally in that they provide nominal force when opened and no force when closed. The following section makes the argument that this is a reasonable approximation.

3.1.1 Hardware Components

The pneumatic system consists of a high and low pressure system. The high-pressure system is largely custom built according to the previous commissioning at the European Space Agency (ESA) and serves the purpose of regulating the pressure from the two Dräger 6.8 L, 300 bar carbon composite Type 3 cylinder [49] to a lower pressure of less than 8 bar. The pressure regulator is a Tescom 44-1300 Series [50] pressure regulator valve and connects to the low-pressure system. Here eight SMC solenoid valves of the type [51] are controlled by a RaspberryPi 3B+ through its GPIO pins connected to a power board, and powered by a 3800 mA h, 6 S Lithium Polymer battery by Conrad Energy with a nominal voltage of 22.2 V. The valve outlets

are connected to Silvent laval nozzles [52] via various pipes and connectors. Figure B.1 in appendix B shows the pneumatic plan of the thruster system.

3.1.2 Valve Type

A cold gas thruster, in essence, is the combination of a tank holding pressurized gas, a regulating valve, and a nozzle through which the gas escapes. Two different approaches are possible for the control valves: continuous and on/off valves. Continuous valves enable the user to control the flow rate of the cold gas and thus can regulate the resulting thrust. The on/off valves, as the name implies, can only be turned on or off, i.e. provide full or no thrust. While the continuous valves provide more functionality, they are also heavier, more expensive, and are less robust to very high pressure [53]. Given their advantages, solenoid-valve based on/off thrusters equip the system.

The phases of the thrusters is then characterized by three time durations:

- τ_{op} := The duration it takes for the valve to open.
- τ_{ac} := The duration the valve is fully open.
- τ_{cl} := The duration it takes for the valve to close.

It must then hold that the overall time τ is:

$$\tau = \tau_{op} + \tau_{ac} + \tau_{cl} \tag{3.1}$$

3.1.3 Thrust

For cold gas thrusters one can assume that the flow of the gas is laminar and therefore abides by the rocket thrust equation [54]. The force enacted by a thruster on the system, also denoted thrust $F(t)$, at some time t throughout the opening and closing process is then modeled as:

$$F(t) = \dot{m}(t)v_e + A_e(t)(p_e - p_a) = A_e(t)\left[\rho v_e^2 + (p_e - p_a)\right] \tag{3.2}$$

where $\dot{m}(t)$ is the mass flow, v_e is the effective exhaust velocity, $A_e(t)$ is the effective exhaust area, p_e is the pressure at the valve and p_a is the ambient pressure. It is assumed that, because the duration τ is small, only the exhaust area and consequently the mass flow is dependent on time, since the valve opens and closes, exposing different areas between zero and the maximal exhaust area $A_{e,max}$. If the duration τ that the valve is open is less than the sum of opening and closing time the valve never fully opens.

In the case of an ideal valve for which $\tau_{op} = \tau_{cl} \to 0$ the force is constant over the time the valve is open and takes the value of the nominal force $\bar{f}$:

$$F_{ideal}(t) = f_{max} =: \bar{f} \tag{3.3}$$

For the present system, which is running on pressurized breathing air, the gas and thruster characteristics are summarized in table 3.2. Where the nozzle exhaust velocity v_e is computed

T	293 K (Room Temperature)	$A_{e,max}$	1.644e−5 m^2
M	0.2896 kg mol^{-1}	ρ	1.2 kg/m^3
γ	1.4	v_e	158.47 m s^{-1}
$p_{ratio} = \frac{p_e}{p}$	$\frac{100\,kPa}{700\,kPa} = 0.1438$	$\Delta p = (p_e - p_a)$	600 kPa

Table 3.2: Gas (breathing air) and thruster characteristics

via:

$$v_e = \sqrt{\frac{TR}{M} \cdot \frac{2\gamma}{\gamma - 1} \cdot \left[1 - \left(\frac{p_e}{p}\right)^{\frac{\gamma-1}{\gamma}}\right]} \tag{3.4}$$

with T the absolute temperature of the inlet gas, R the universal gas law constant, M the gas' molecular mass, γ the isentropic expansion factor, p_e the absolute pressure of the exhaust gas at the nozzle exit and p the absolute pressure of the inlet gas as introduced in [55]. From this, the nominal force is computed using equation (3.2):

$$\bar{f} \approx 10.36\,\text{N} \tag{3.5}$$

However, given that the valves are non-ideal, a time dependence of the force remains for small opening times. In particular, some observations can be made: For $\tau >> \tau_{op} + \tau_{cl}$ the opening and closing time is negligible, as their contribution to the overall impulse is small in contrast to the opening time. Thus imposing a minimal pulse duration such that $\tau >> \tau_{op} + \tau_{cl}$ simplifies the model at the cost of forfeiting a lower minimum impulse. The present thrusters open and close within 10 ms, thus τ needs to be an order of magnitude larger, i.e. $\tau = 100$ ms. Given that the mass of the overall system is in a different order of magnitude than the force exerted by the thrusters (cf. Table 3.1 and equation (3.5)) the precision obtained with the minimal pulse is considered sufficient. Throughout the rest of this work, the thrusters are therefore assumed to be ideal.

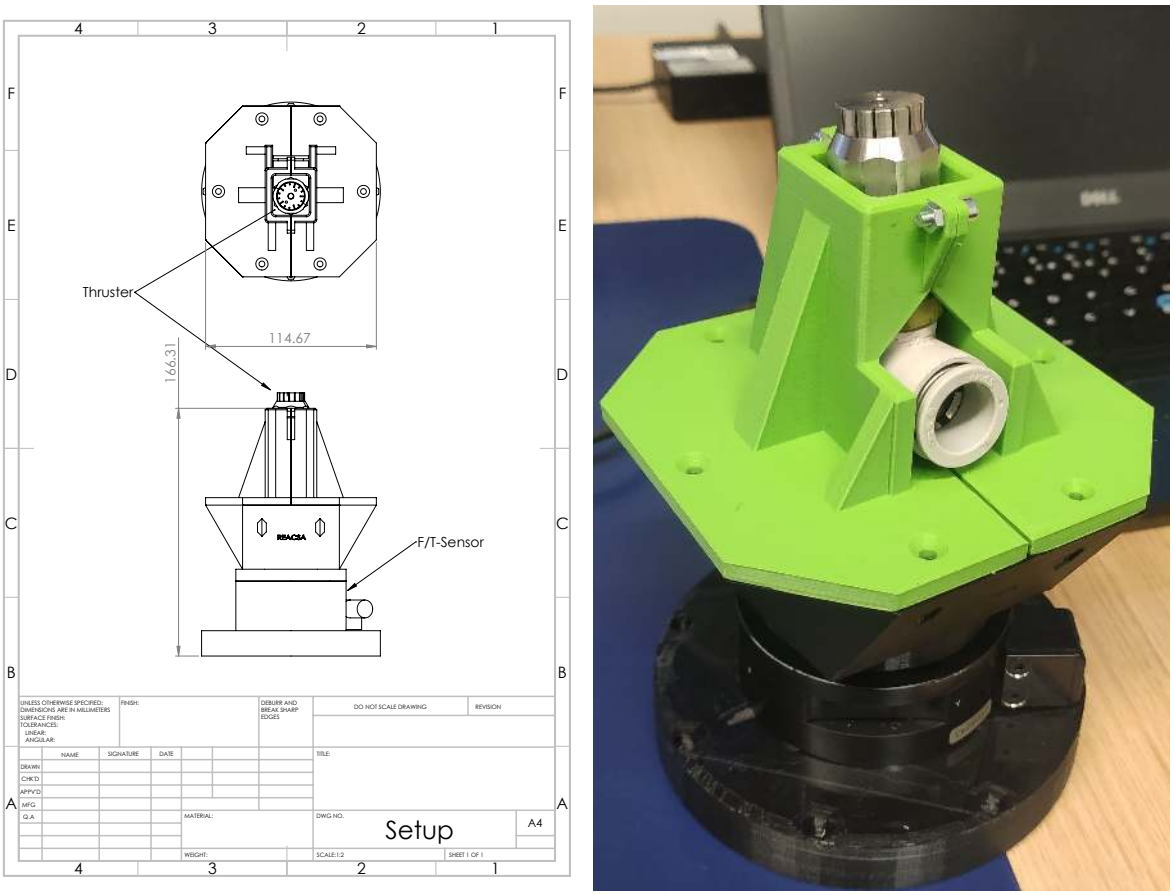


Figure 3.3: Thruster test stand. Technical drawing (left) and 3D-Printed stand (right) using the ATI Gamma Force Torque sensor FT5243 [13]. The force-torque sensor is mounted on the base. On top, an enclosure for the thruster is mounted (in green). The air inlet is routed to the side such that the thruster expels the air to the top and thus pushes down onto the force-torque sensor.

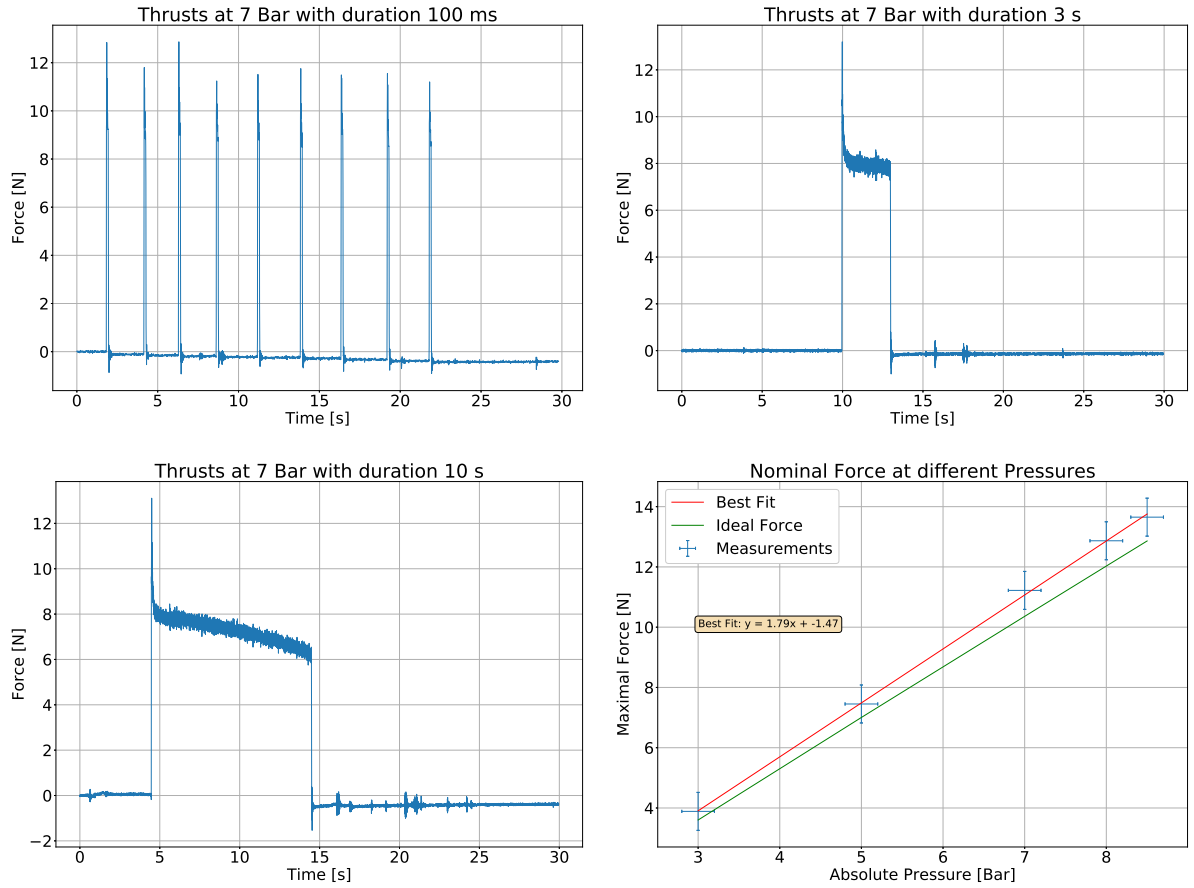


Figure 3.4: Top row and bottom left: Pulses of different thrust durations (100 ms, 3 s, 10 s) at operating pressure 7 bar. Bottom Right: Linear fit for maximal force for pulses of length 0.1 s at different operating pressure and ideal nominal force according to the introduced model. Note that the pressure is given in absolute terms, i.e. to get the pressure difference, one must subtract atmospheric pressure.

To validate this model, the thruster is put into a test stand (shown in Figure 3.3), and the exerted force is measured for different firing durations and operating pressures. Thereby is the force-torque sensor mounted at the base of the platform and the thruster pointing up in a fixture on top. Figure 3.4 shows the norm of the resulting force responses. For all durations, the maximal force peak is reached immediately after firing. The maximal value stays approximately consistent (± 1 N) for a number of firings. However, for longer firing durations (3 s) a decrease by 4.5 N to a lower, initially constant value occurs. For even longer firing (10 s), the force further decreases linearly. As implied by the model, the peak force also linearly depends on the operating pressure and matches quite closely to the theory introduced in section 3.1. The error bars along the pressure stem from the analog gauge's precision and the error bars in the force axis are the standard deviation over twenty repeated measurements of the same pulse. This indicates that our model fits the actual thrusters quite well. While there are some deviations from the predicted nominal values, these are attributed to the imprecision in the independent

variables, especially the pressure gauge that provides a precision of ±0.2 bar.

3.1.4 Shared Valve-Bank

Two sets of four thrusters each share one bank, i.e share one air-flow. Therefore, when firing two or more thrusters of the same bank simultaneously the pressure drops and therefore also the thrust deviates from the nominal value. To ensure this deviation is not significant we repeated the previous experiment while firing two thrusters from the same bank and from opposing banks instead of only one. Note that due to the valves exhibiting leakage at the previously used 7 bar operating pressure, the pressure is reduced to 6 bar for this experiment. The resulting force profiles are shown in Figure 3.5. While some decrease of approximately 1 N from the nominal force at the reduced pressure is visible this is an order of magnitude less than the nominal force and thus is considered negligible. However, this effect is indeed more pronounced if three or even more thrusters are firing simultaneously for an extended duration. As this case of three or more thrusters firing is not necessary for particular movements (and not likely in the control structure introduced in this thesis) this case will not be handled explicitly.

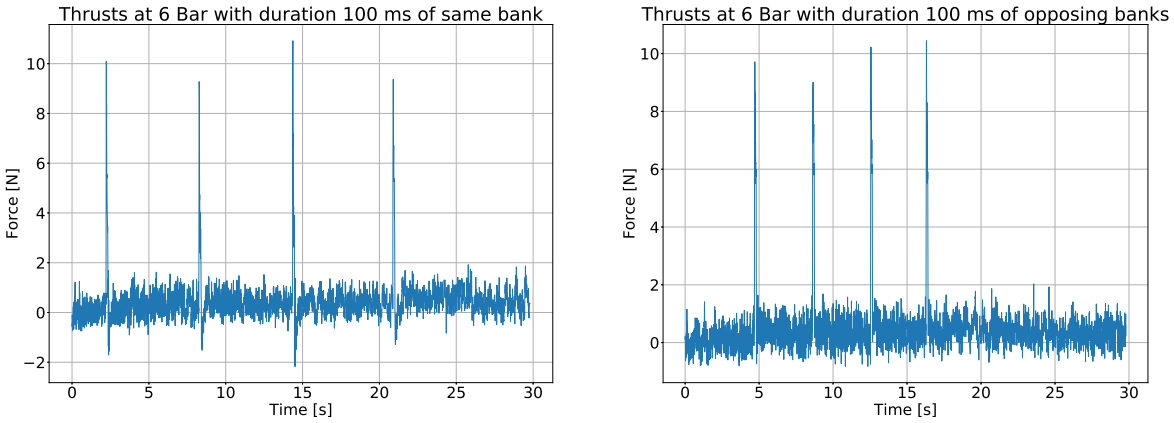


Figure 3.5: Left: Force profile of a single thruster while firing two thrusters of the same bank simultaneously for 100 ms at an operating pressure of 6 bar. Right: Force profile of a single thruster while firing two thrusters of opposing banks simultaneously for 100 ms at an operating pressure of 6 bar.

3.2 Reaction Wheel Model

3.2.1 Hardware Components

The motor used is a Nanotec DB80M048030-ENM05J Brushless Direct-Current (BLDC)-motor [56]. Its properties are summarized in table 3.3. The maximal RPM of the RW is further limited to 500 RPM since the motor is also loaded with the RW disk.

Rated Power	534 W
Rated Torque	1.7 N m
Maximal RPM	3000 RPM

Table 3.3: Properties of the Nanotec DB80M048030-ENM05J motor.

Component	Moment of Inertia
Reaction Wheel	$0.047\,\mathrm{kgm}^2$
ACROBAT + SATSIM	$11.506\,\mathrm{kgm}^2$
RECAP	$0.67\,\mathrm{kgm}^2$

Table 3.4: MoI of the different sub-components

3.2.2 Dynamic Model

In general, RW work by exploiting the conservation of angular momentum: A system on which no external forces or torques are acting will maintain its current angular momentum [57]. That way, when accelerating the reaction wheel, the overall system will perform a rotation about the same axis in the opposite direction of rotation, related via their moments of inertia I_{RW} and I_S respectively:

$$I_{RW} \cdot \omega_{RW} = -I_S \cdot \omega_S \tag{3.6}$$

Where ω_{XX} are the angular velocities. The RW provided by RECAP is manufactured from steel and consists of an inner disk and an outer ring. It is attached to a BLDC motor and encased in protective housing. Table 3.4 shows the Moment of Inertia (MoI) provided by the RW compared to the overall system MoI.

The motor provides an interface that allows for velocity control, i.e. the user commands some velocity and an underlying hardware controller attempts to control the input voltage to the motor such that the commanded velocity ω_{RW} is executed as well as possible. For each torque τ, a controller wants to impose on the system the respective velocity command $\omega_{RW,new}$ is computed by numerically integrating the torque command from the current velocity as follows:

$$\omega_{RW,new} = \omega_{RW} + \frac{\Delta t}{I_{RW}}\tau \tag{3.7}$$

where Δt is the interval between two velocity commands. The underlying schematic of the motor is shown in figure 3.6, however, since for this work only the provided torque is of interest, the motor will simply be assumed to be able to follow those velocity commands.

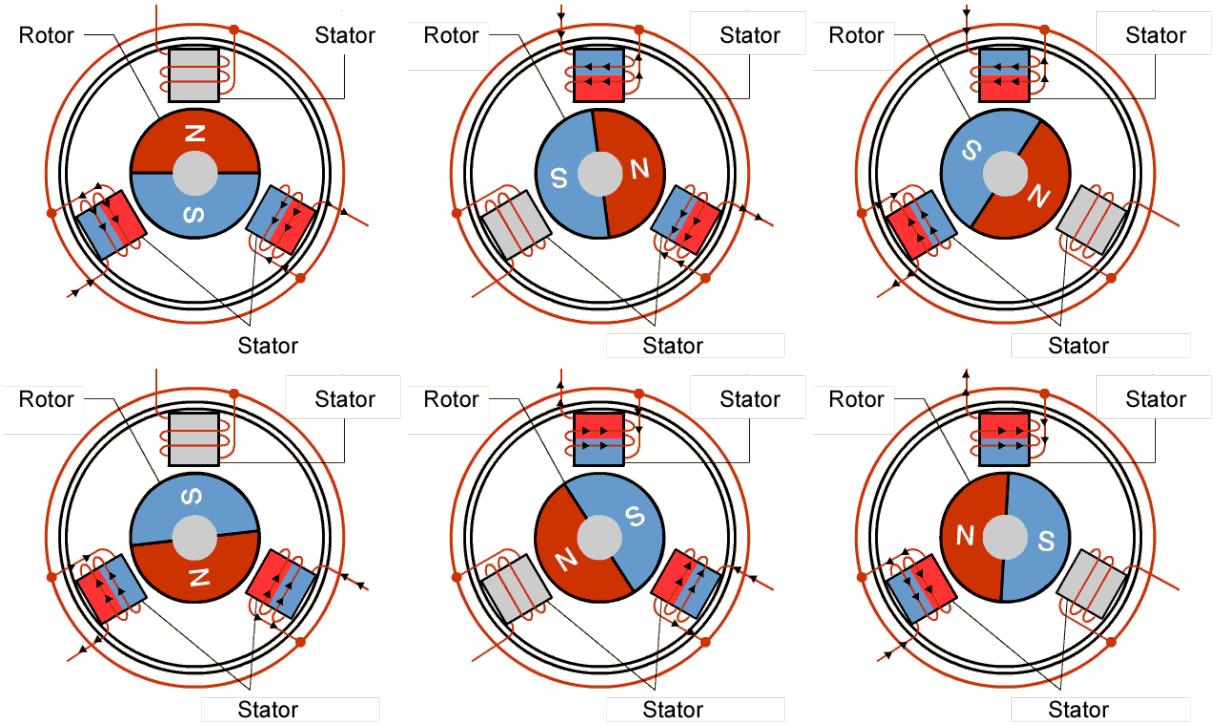


Figure 3.6: Schematic of a BLDC-motor. The motor is controlled via the coils by activating them in three phases which are shown from left to right and top to bottom. Courtesy of [14]. The rotor is a permanent magnet while stationary coils are arranged in a configuration around it (Pictured: three coils that function as stators). The current in the coils is controlled such that in each phase the magnetic fields created by the coils repel and attract the perma-magnet such that it rotates at the desired velocity.

Under this assumption, the following state-equation fully describes the orientation of the overall system θ and the RW velocity ω_{RW} :

$$\begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \\ \dot{\omega}_{RW} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \theta \\ \dot{\theta} \\ \omega_{RW} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{-1}{I_b} \\ \frac{1}{I_w} \end{bmatrix} \tau \tag{3.8}$$

3.3 Gliding Model

3.3.1 Hardware Components

To allow the system to glide over the flat floor, ACROBAT uses three 200 mm air-bearings [58] which are supplied with compressed air at 4.8 bar to provide five microns lift for up to 777 kg.

3.3.2 Dynamic Model

Using the models derived in the previous sections the overall motion model is defined as follows. First, the labels for all actuators is defined as in Figure 3.7. In particular, the thrusters, and

their produced forces, are numbered from zero to seven in a counterclockwise direction. The torque on the RW created by the motor τ and its rotational velocity ω_{RW} is also defined in the mathematically positive direction. Consequently, the resulting torque on the overall system acts in the mathematically negative rotational direction. The position of the system is defined relative to some world coordinate system and the orientation θ is defined relative to that x-axis. The world coordinate system is arbitrarily placed somewhere on the flat-floor for each experiment but is generally located close to the geometric center of the flat-floor.

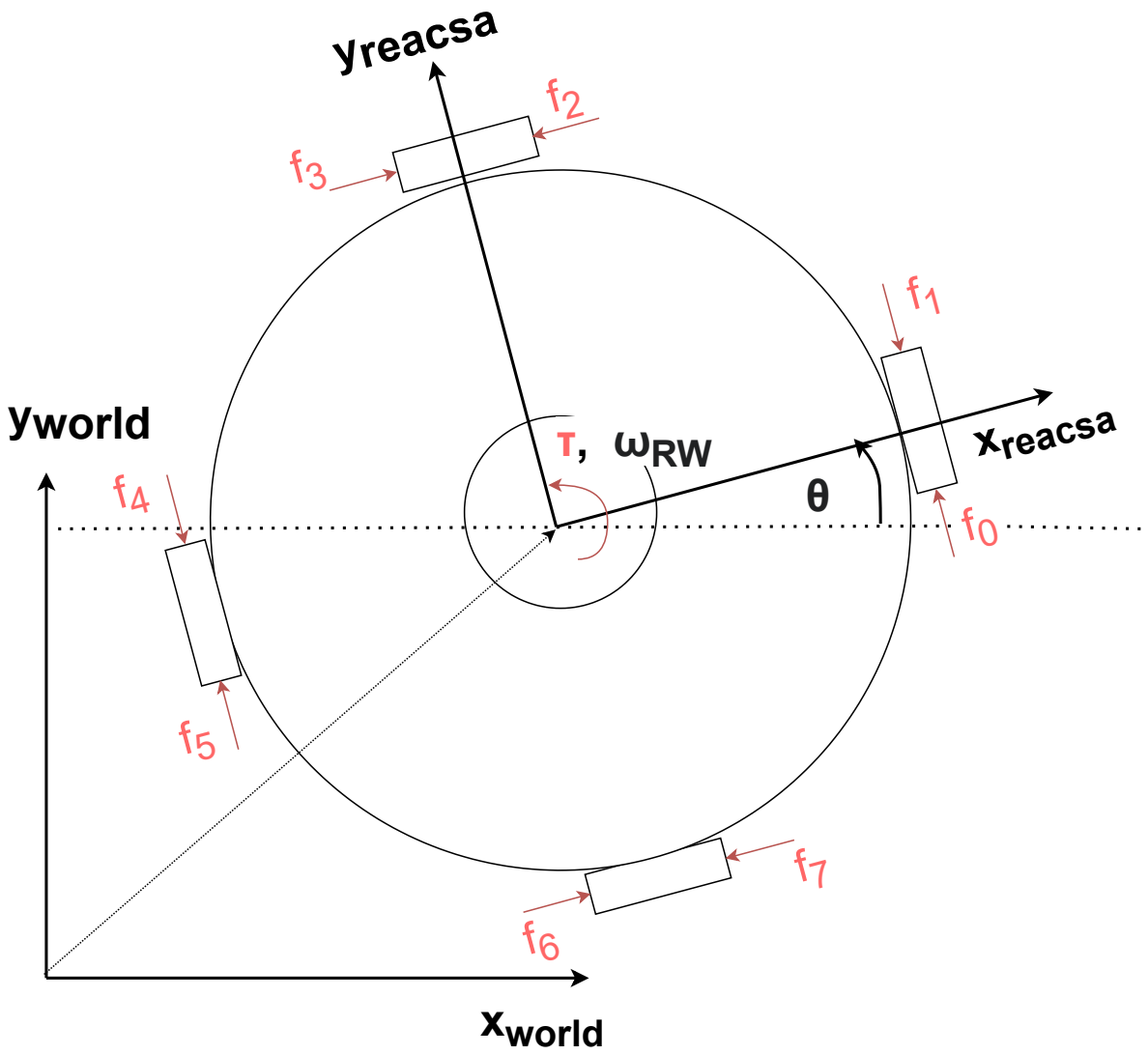


Figure 3.7: Definitions of coordinate systems and actuators in the REACSA system. The thrusters and their respective exerted force are numbered in the mathematically positive direction starting from the x-axis of the system. The torque on the RW and the RW velocity are defined in the positive direction. The overall system pose is defined with respect to some world coordinate frame where the orientation is defined as the angle between the world coordinate system and local coordinate system x-axes.

Using the state and control vectors:

$$\mathbf{x} = \begin{bmatrix} x & y & \theta & \dot{x} & \dot{y} & \dot{\theta} & \omega_{RW} \end{bmatrix}^T \tag{3.9}$$

$$\mathbf{u} = \begin{bmatrix} \tau & f_0 & f_1 & f_2 & f_3 & f_4 & f_5 & f_6 & f_7 \end{bmatrix}^T \tag{3.10}$$

the resulting continuous state-equation $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u})$ is:

$$\mathbf{f}(\mathbf{x}, \mathbf{u}) = \begin{bmatrix} \mathbf{0}^{3\times3} & \mathbf{I}^{3\times3} & 0 \\ & \mathbf{0}^{4\times7} & \end{bmatrix} \mathbf{x} + \begin{bmatrix} & & & & \mathbf{0}^{3\times7} & & & & \\ 0 & \frac{-s_\theta}{m} & \frac{s_\theta}{m} & \frac{-c_\theta}{m} & \frac{c_\theta}{m} & \frac{s_\theta}{m} & \frac{-s_\theta}{m} & \frac{c_\theta}{m} & \frac{-c_\theta}{m} \\ 0 & \frac{c_\theta}{m} & \frac{-c_\theta}{m} & \frac{-s_\theta}{m} & \frac{s_\theta}{m} & \frac{-c_\theta}{m} & \frac{c_\theta}{m} & \frac{s_\theta}{m} & \frac{-s_\theta}{m} \\ \frac{-1}{I_b} & \frac{r}{I_b} & \frac{-r}{I_b} & \frac{r}{I_b} & \frac{-r}{I_b} & \frac{r}{I_b} & \frac{-r}{I_b} & \frac{r}{I_b} & \frac{-r}{I_b} \\ \frac{1}{I_w} & & & & \mathbf{0}^{1\times6} & & & & \end{bmatrix} \mathbf{u} \tag{3.11}$$

where s_θ and c_θ denote the sin and cos of the respective angle, m is the system mass, I_w and I_b are the MoI of the reaction wheel and the overall system respectively. Since the force is produced by the solenoid valves described in the previous section it holds that

$$f_i \in \{0, \bar{f}\} \; \forall i \in [0, 7] \; . \tag{3.12}$$

However, since the measurements are only available at discrete times, the above system is also represented as a discrete-time state-space model. From the linear, time-varying state-space model the respective discrete-time representation is derived as follows (cf. [59]):

$$\dot{\mathbf{x}} = \mathbf{A}(t)\mathbf{x}(t) + \mathbf{B}(t)\mathbf{u}(t) \tag{3.13}$$

Which has the solution:

$$\mathbf{x}(t) = e^{\mathbf{A}(t_0)(t-t_0)}x(t_0) + \int_{t_0}^{t} e^{\mathbf{A}(\tau)(t-\tau)}\mathbf{B}(\tau)\mathbf{u}(\tau)d\tau \tag{3.14}$$

Applying this solution to one specific control interval, i.e let $t_0 = t_k$ and $t = t_{k+1}$

$$\mathbf{x}_{k+1} = e^{\mathbf{A}_k(t_{k+1}-t_k)}x_k + \int_{t_k}^{t_{k+1}} e^{\mathbf{A}(\tau)(t_{k+1}-\tau)}\mathbf{B}(\tau)\mathbf{u}(\tau)d\tau \tag{3.15}$$

Assuming a constant control frequency the difference in time is denoted as $T = t_{k+1} - t_k$. Further, applying the zero-order hold assumption, i.e. assuming constant control input and state matrices over one interval and applying the substitution $\tau' = t_{k+1} - \tau$ it follows:

$$\mathbf{x}_{k+1} = e^{\mathbf{A}_k T}x_k + \int_0^T e^{\mathbf{A}_k\tau}d\tau\mathbf{B}_k\mathbf{u}_k \tag{3.16}$$

Hence the discrete system matrices are found via:

$$\mathbf{A}_d = e^{\mathbf{A}T} \qquad \mathbf{B}_d = \int_0^T e^{\mathbf{A}\tau}d\tau\mathbf{B} \tag{3.17}$$

Applying this to the continuous dynamics in equation (3.11) yields the discrete state-space model in the form $\mathbf{x}_{k+1} = \mathbf{f}_d(\mathbf{x}_k, \mathbf{u}_k)$ as follows:

$$
\begin{aligned}
\mathbf{x}_{k+1} = &\begin{bmatrix}
1 & 0 & 0 & T & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & T & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & T & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix} \mathbf{x}_k \\
+ &\begin{bmatrix}
0 & -\frac{T^2}{2}\frac{s_\theta}{m} & \frac{T^2}{2}\frac{s_\theta}{m} & -\frac{T^2}{2}\frac{c_\theta}{m} & \frac{T^2}{2}\frac{c_\theta}{m} & \frac{T^2}{2}\frac{s_\theta}{m} & -\frac{T^2}{2}\frac{s_\theta}{m} & \frac{T^2}{2}\frac{c_\theta}{m} & -\frac{T^2}{2}\frac{c_\theta}{m} \\
0 & \frac{T^2}{2}\frac{c_\theta}{m} & -\frac{T^2}{2}\frac{c_\theta}{m} & -\frac{T^2}{2}\frac{s_\theta}{m} & \frac{T^2}{2}\frac{s_\theta}{m} & -\frac{T^2}{2}\frac{c_\theta}{m} & \frac{T^2}{2}\frac{c_\theta}{m} & \frac{T^2}{2}\frac{s_\theta}{m} & -\frac{T^2}{2}\frac{s_\theta}{m} \\
-\frac{T^2}{2}\frac{1}{Ib} & \frac{T^2}{2}\frac{r}{I_b} & -\frac{T^2}{2}\frac{r}{I_b} & \frac{T^2}{2}\frac{r}{I_b} & -\frac{T^2}{2}\frac{r}{I_b} & \frac{T^2}{2}\frac{r}{I_b} & -\frac{T^2}{2}\frac{r}{I_b} & \frac{T^2}{2}\frac{r}{I_b} & -\frac{T^2}{2}\frac{r}{I_b} \\
0 & -T\frac{s_\theta}{m} & T\frac{s_\theta}{m} & -T\frac{c_\theta}{m} & T\frac{c_\theta}{m} & T\frac{s_\theta}{m} & -T\frac{s_\theta}{m} & T\frac{c_\theta}{m} & -T\frac{c_\theta}{m} \\
0 & T\frac{c_\theta}{m} & -T\frac{c_\theta}{m} & -T\frac{s_\theta}{m} & T\frac{s_\theta}{m} & -T\frac{c_\theta}{m} & T\frac{c_\theta}{m} & T\frac{s_\theta}{m} & -T\frac{s_\theta}{m} \\
-T\frac{1}{Ib} & T\frac{r}{I_b} & -T\frac{r}{I_b} & T\frac{r}{I_b} & -T\frac{r}{I_b} & T\frac{r}{I_b} & -T\frac{r}{I_b} & T\frac{r}{I_b} & -T\frac{r}{I_b} \\
\frac{1}{I_w} & & & & \mathbf{0}^{1\times 6} & & & &
\end{bmatrix} \mathbf{u}_k
\end{aligned}
\tag{3.18}
$$

Chapter 4

Controller

The controller should be capable of performing the best possible action to steer the robot from some initial state to some final state. To be able to specify boundary conditions and to make sure the overall followed trajectory is optimal and not only one instantaneous action, the controller is split into two segments: A *Trajectory Planner* that pre-computes the optimal state-trajectory and control and a *Trajectory Follower* that computes the control values online, which are required to steer the robot onto the path.

4.1 Overview

To give the reader a high-level overview of the entire control architecture, figure 4.1 shows a block diagram. The system is split into two main modules: the trajectory planner and the trajectory tracker. The trajectory finder computes an optimal trajectory (according to some cost function) a priori, given the constraints specified by the user. The trajectory tracker then tries to follow the trajectory. For this, it consists of three sub-modules: the continuous feedback controller, the modulator, and the observer. The feedback controller computes a continuous desired force using Time-Varying Linear Quadratic Regulator (TVLQR), which would be required to optimally follow the trajectory. The modulator then chooses opening times for the on/off thrusters to best match the desired continuous force using a $\Sigma\Delta$-Modulation scheme. These control actions are forwarded to the system which then acts according to its dynamics. The observer uses the most recent measurement available at the sensors and the commanded control input to optimally estimate the current system state. Modules and sub-modules are interchangeable in this architecture, allowing for the employment of different modulators, observers or feedback controllers. In particular, the trajectory follower is capable of following not only the previously computed optimal trajectory but any admissible (physically possible) trajectory.

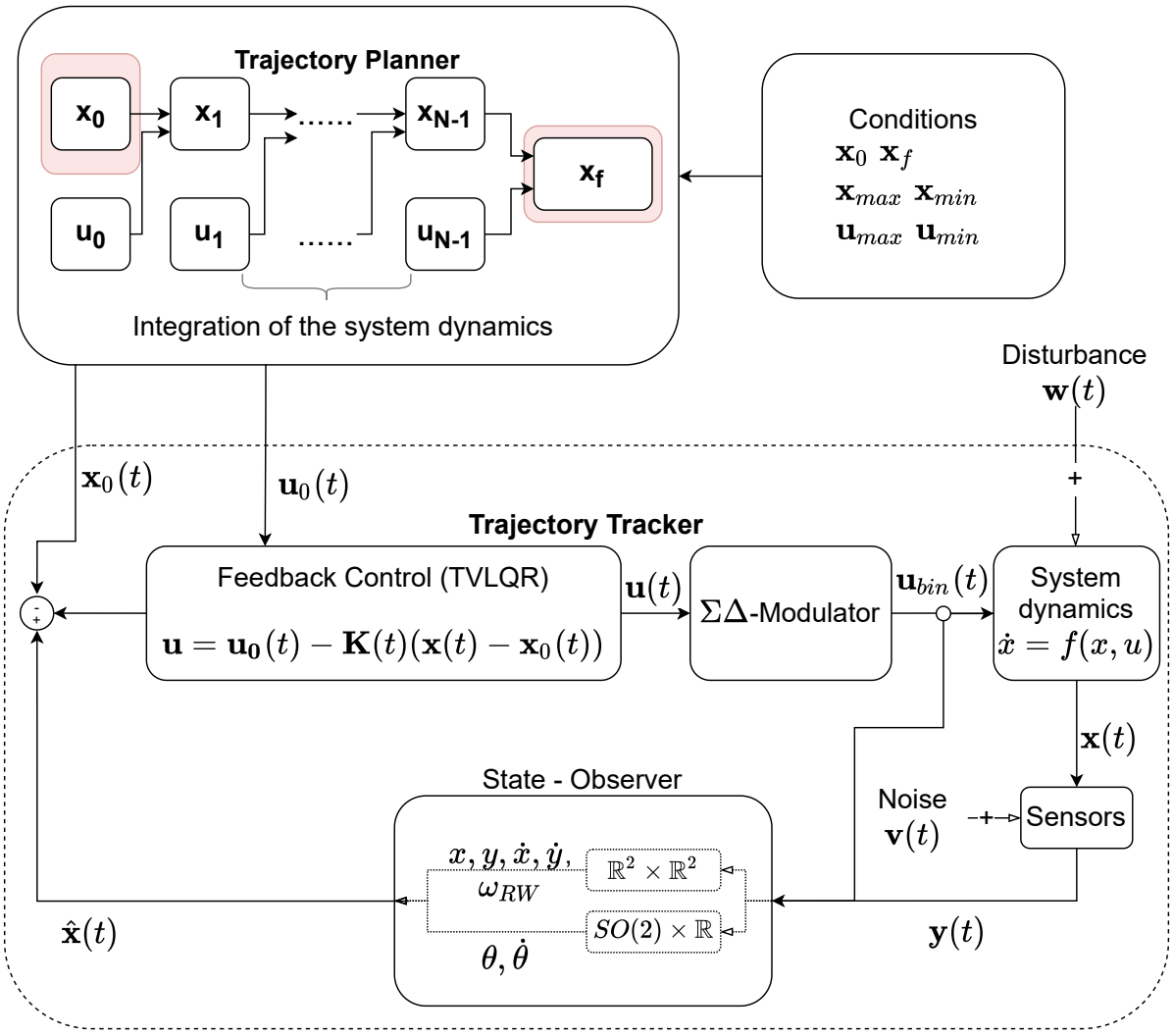


Figure 4.1: Block diagram of the overall control architecture. The user provides conditions such as the initial and final state as well as the state and control limitations. From those conditions, the trajectory planner pre-computes an optimal trajectory to follow. The trajectory is then followed by a feedback controller that computes continuous control inputs which are partially modulated onto the thruster. A state observer uses the most recent measurements, and the commanded control input to estimate the current state and feed the estimation back to the controller as state feedback.

4.2 Trajectory Planner: Optimizing over the Discrete States and Control Inputs

Chapter 3 introduces the system model in the form of a non-linear state-space model such as

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) \tag{4.1}$$
$$\mathbf{y} = \mathbf{o}(\mathbf{x}) \tag{4.2}$$

For any realistic system the state variables, as well as the control input, are limited. These limits can be expressed as follows:

$$\mathbf{h}(\mathbf{x}) = \mathbf{0} \tag{4.3}$$
$$\mathbf{g}(\mathbf{x}) \leq \mathbf{0} \tag{4.4}$$
$$\mathbf{q}(\mathbf{u}) \leq \mathbf{0} \tag{4.5}$$

Thereby the functions $\mathbf{h}$ and $\mathbf{g}$ express the equality and inequality constraints with respect to the state. The function $\mathbf{q}$ encodes the limits of the control.

Given these limitations of the system, one can define a cost function J associated with a trajectory. This cost function J often takes the following form:

$$J = \phi(t_f, \mathbf{x}(t_f)) + \int_{t_0}^{t_f} L(\mathbf{x}(t), \mathbf{u}(t), t)dt \tag{4.6}$$

where ϕ describes the cost imposed on the final state $\mathbf{x}_f$ at the final time t_f and L describes the cost of all states and control actions along the entire trajectory. The distinction between all states and the final state is useful since most trajectories are executed to reach some final. This allows setting the cost for missing the final state independently from all other states along the way.

Any trajectory between two states $\mathbf{x}_{init}$ and $\mathbf{x}_{final}$, that satisfies the constraints (including the system dynamics) is called an *admissible* trajectory. Finding the admissible trajectory that minimizes the cost function J is the process of trajectory optimization.

4.2.1 Cost Function

The cost function J is the main factor that determines the shape of the resulting trajectory. The two qualitative criteria that should be optimized are propellant efficiency and trajectory duration. Given the actuator limits, there is a minimal time in which the system can reach the final state. This trajectory is called time-optimal. On the other side, if the desired finishing time is infinite the most propellant-efficient action is to do nothing and simply wait.
In order to find a trade-off between both criteria, the approach is as follows:

1. Find the time-optimal trajectory using the cost function $J = t_f$. In addition, it is enforced that $t_f \geq 0$. This will result in a non-propellant-efficient, "bang-bang" - controller. However, it will also provide a lower bound for the time that the overall trajectory takes.

2. Define a desired final time that is a result of the multiplication of a buffer factor α with the time-optimal final time:

$$t_{f,des} = \alpha \cdot t_f^* \tag{4.7}$$

The buffer factor is chosen heuristically to be $\alpha = 12$ to provide slow movement with less than $0.025\,\mathrm{m\,s^{-1}}$.

3. Find the trajectory that minimizes the propellant usage (or maximizes the tank status at the end of the trajectory) which finishes at the desired final time. As seen in chapter 3 the force exerted by the thrusters is approximately proportional to the propellant used. Therefore a reasonably proxy for minimal-propellant is minimal-force. The cost function is then:

$$J = \sum_{k=1}^{N} \mathbf{u}_k \mathbf{R} \mathbf{u}_k^T \tag{4.8}$$

where $\mathbf{R}$ is a diagonal matrix that repeats the respective weights for each control value. By choosing a large weight for the thrusters and a small (or even zero) weight for the reaction wheel the actuation of the thrusters is minimized for the resulting trajectory.

There are many approaches to finding the *optimal* trajectory. One such approach is Direct Collocation, which is explained in the following subsection.

4.2.2 Direct Collocation

The key aspect of direct collocation for trajectory optimization is to discretize the trajectory in time at N instances of time t_k, denoted as knot points in the following. Each state and control variable at each knot point is a decision variable within an optimization problem. Then, the solution to the said problem is a trajectory consisting of N states and control values. Further, all integral components (usually in the cost function but also possible in the constraints) are approximated via sums. In particular, we also transfer the system dynamics from the differential form to the integral form in order allow for the same approximation as such:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) \Rightarrow \int_{t_k}^{t_{k+1}} \dot{\mathbf{x}} = \int_{t_k}^{t_{k+1}} \mathbf{f}(\mathbf{x}, \mathbf{u}) \tag{4.9}$$

In the following the explicit dependence of the dynamics on the state and the input is omitted for readability. To approximate the integrals to a higher-order precision the Hermite–Simpson collocation [60] is used. Thereby the function is approximated using quadratic splines. This adds the additional benefit that the result has a continuous first derivative. To derive the integral approximation one finds a quadratic function $v(t)$ that matches the original function $w(t)$ at three points, the start of the interval, the midpoint, and the end:

$$v(t) = At + Bt^2 + Ct^3 \tag{4.10}$$

with (4.11)

$$v(t_0) = w(t_0),\ v\left(\frac{t_f - t_0}{2}\right) = w\left(\frac{t_f - t_0}{2}\right),\ v(t_f) = w(t_f) \tag{4.12}$$

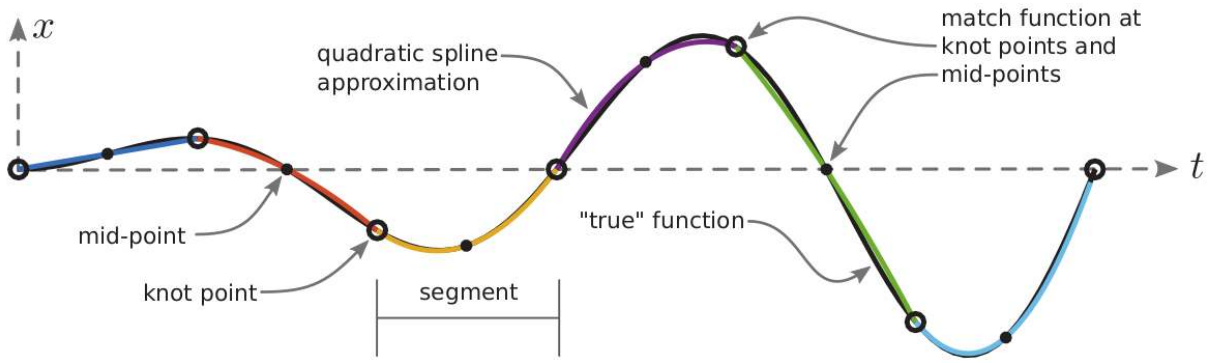


Figure 4.2: Visualization of the Hermite-Simpson Collocation Method. Empty circles are knot points, filled circles are the respective mid-points. Courtesy of Kelly [15]

Solving this system of linear equations yields the coefficients A, B, C:

$$A = w(t_0) \tag{4.13}$$

$$B = \frac{-3w(t_0) + 4w((t_f - t_0)/2) - w(t_f)}{t_f - t_0} \tag{4.14}$$

$$C = \frac{2w(t_0) - 4w((t_f - t_0)/2) + 2w(t_f)}{t_f - t_0} \tag{4.15}$$

Thus the integral over one spline-segment S of length h is approximated as:

$$\int_S w(\tau)d\tau \approx \int_0^h v(\tau)d\tau = \left[Ah + \frac{1}{2}Bh^2 + \frac{1}{3}Ch^3\right] = \frac{h}{6}\left(w(0) + 4w\left(\frac{h}{2}\right) + w(h)\right) \tag{4.16}$$

Some integral of some arbitrary function is then approximated via the sum over all integrals of all spline segments within the interval:

$$\int_{t_0}^{t_F} w(\tau)d\tau = \sum_{k=0}^{N-1} \frac{\Delta t}{6}(w_k + 4w_{k+1/2} + w_{k+1}), \quad \text{with } \Delta t = \frac{t_F - t_0}{N} \tag{4.17}$$

where $w_{k+1/2}$ denotes the midpoint between two knot points. Figure 4.2 visualizes the Hermite-Simpson collocation on an arbitrary one-dimensional function in time.

Applying this approximation to equation (4.9) yields:

$$\mathbf{x}_{k+1} - \mathbf{x}_k = \frac{\Delta t}{6}(\mathbf{f}_k + 4\mathbf{f}_{k+1/2} + \mathbf{f}_{k+1}) \tag{4.18}$$

This equation must hold for all pairs of subsequent knot points. However, since the system dynamics at the midpoint are a function of the state and the control at the midpoint this must be computed explicitly. Assuming a cubic state (from integrating the quadratic dynamic approximation from before) an additional condition is derived by solving the linear equation to find the cubic spline connecting the two states and evaluating it at the midpoint:

$$\mathbf{x}_{k+1/2} = \frac{1}{2}(\mathbf{x}_k + \mathbf{x}_{k+1}) + \frac{\Delta t}{8}(\mathbf{f}_k - \mathbf{f}_{k+1}) \tag{4.19}$$

Lastly, in the particular present case, a number of actuators are binary, i.e. they can only be switched on or off and remain in either state for the full duration of the segment. Thus the last conditions are:

$$\mathbf{u}_k = \begin{bmatrix} \mathbf{u}_{k,contiuous} \\ \mathbf{u}_{k,binary} \end{bmatrix} \Rightarrow \mathbf{u}_{k,binary} \in \{0, f_{nom}\}^n \tag{4.20}$$

where n is the number of control variables that are binary.

This finally fully describes the discretized trajectory. Hence, now the problem of finding the optimal trajectory from some initial to some final state can be formulated:

$$\begin{aligned}
&\min_{\mathbf{x},\mathbf{u}} \left\{ J = \phi(t_f, \mathbf{x}(t_f)) + \int_{t_0}^{t_f} L(\mathbf{x}(t), \mathbf{u}(t), t) dt \right\} \\
&\text{s.t. } \mathbf{x}(0) = \mathbf{x}_{init}, \quad \mathbf{x}(t_f) = \mathbf{x}_{final} \\
&\quad \mathbf{h}(\mathbf{x}_k) = \mathbf{0} && \forall k \in [0, N-1] \\
&\quad \mathbf{g}(\mathbf{x}_k) \leq \mathbf{0} && \forall k \in [0, N-1] \\
&\quad \mathbf{q}(\mathbf{u}_{k,continuous}) \leq \mathbf{0} && \forall k \in [0, N-1] \\
&\quad \mathbf{x}_{k+1} - \mathbf{x}_k = \frac{\Delta t}{6}(\mathbf{f}_k + 4\mathbf{f}_{k+1/2} + \mathbf{f}_{k+1}) && \forall k \in [0, N-1] \\
&\quad \text{where } \mathbf{x}_{k+1/2} = \frac{1}{2}(\mathbf{x}_k + \mathbf{x}_{k+1}) + \frac{\Delta t}{8}(\mathbf{f}_k - \mathbf{f}_{k+1}) && \forall k \in [0, N-1] \\
&\quad \text{and } \mathbf{u}_{k+1/2,continuous} = \mathbf{u}_{k,continuous} && \forall k \in [0, N-1] \\
&\quad \mathbf{u}_{k,binary} \in \{0, f_{nom}\}^n && \forall k \in [0, N-1]
\end{aligned} \tag{4.21}$$

Because of the restriction on the actuators, the problem is of the class Mixed-Integer Non-Linear Programming (MINLP). These problems are increasingly difficult to solve in that they are NP-hard and combine challenges of handling non-linearities with a combinatorial explosion of integer variables [61]. In fact, those problems are considered computationally intractable in the general case and can only be solved in a reasonable amount of time for very small control horizons for a small number of dimensions in control and state space [62, 63]. By relaxing the binary condition on the respective control variables, the problem is reduced to a problem of the class Non-Linear Programming (NLP), which can be solved significantly faster. Now that the control variable is also considered continuous, the same interpolation at the collocation point needs to be enforced:

$$\mathbf{u}_{k+1/2} = \frac{1}{2}(\mathbf{u}_k + \mathbf{u}_{k+1}) \tag{4.22}$$

The resulting optimization problem, using the explicit constraints for the system and also drop-

ping the equality constraints that don't involve the initial and final state, is:

$$
\begin{aligned}
&\min_{\mathbf{x},\mathbf{u}} \left\{ J = \phi(t_f, \mathbf{x}(t_f)) + \int_{t_0}^{t_f} L(\mathbf{x}(t), \mathbf{u}(t), t) dt \right\} \\
&\text{s.t. } \mathbf{x}(0) = \mathbf{x}_{init}, \quad \mathbf{x}(t_f) = \mathbf{x}_{final} \\
&\qquad \begin{bmatrix} \mathbf{1} \\ -\mathbf{1} \end{bmatrix} \mathbf{x}_k + \begin{bmatrix} -\mathbf{x}_{max} \\ \mathbf{x}_{min} \end{bmatrix} \leq \mathbf{0} && \forall k \in [0, N-1] \\
&\qquad \begin{bmatrix} \mathbf{1} \\ -\mathbf{1} \end{bmatrix} \mathbf{u}_k + \begin{bmatrix} -\mathbf{u}_{max} \\ \mathbf{u}_{min} \end{bmatrix} \leq \mathbf{0} && \forall k \in [0, N-1] \\
&\qquad \mathbf{x}_{k+1} - \mathbf{x}_k = \frac{\Delta t}{6}(\mathbf{f}_k + 4\mathbf{f}_{k+1/2} + \mathbf{f}_{k+1}) && \forall k \in [0, N-1] \\
&\qquad \text{where } \mathbf{x}_{k+1/2} = \frac{1}{2}(\mathbf{x}_k + \mathbf{x}_{k+1}) + \frac{\Delta t}{8}(\mathbf{f}_k - \mathbf{f}_{k+1}) && \forall k \in [0, N-1] \\
&\qquad \text{and } \mathbf{u}_{k+1/2} = \frac{1}{2}(\mathbf{u}_k + \mathbf{u}_{k+1}) && \forall k \in [0, N-1]
\end{aligned}
\tag{4.23}
$$

Readily available, open source solvers, such as "IPOPT" [64] using some programming interface, such as "Drake" [19], find a solution to this problem in a reasonable amount of time. However, this relaxation yields trajectories that demand continuous control input that cannot be provided by the discrete or binary actuator. Therefore, any trajectory tracking controller needs to consider how to translate the desired control input into the discrete space. This is further discussed in subsection 4.3.

4.2.3 Example

Figure 4.3 shows an example trajectory connecting two states of the form

$$
\mathbf{x} = \begin{bmatrix} x & y & \theta & \dot{x} & \dot{y} & \dot{\theta} & \omega_{RW} \end{bmatrix} \tag{4.24}
$$

using this optimization approach. It clearly shows that for two states the trajectory consists of an acceleration, coasting, and breaking phase, which is energy optimal. Additionally, it shows that all state and control constraints are never exceeded, as enforced by the optimization problem.

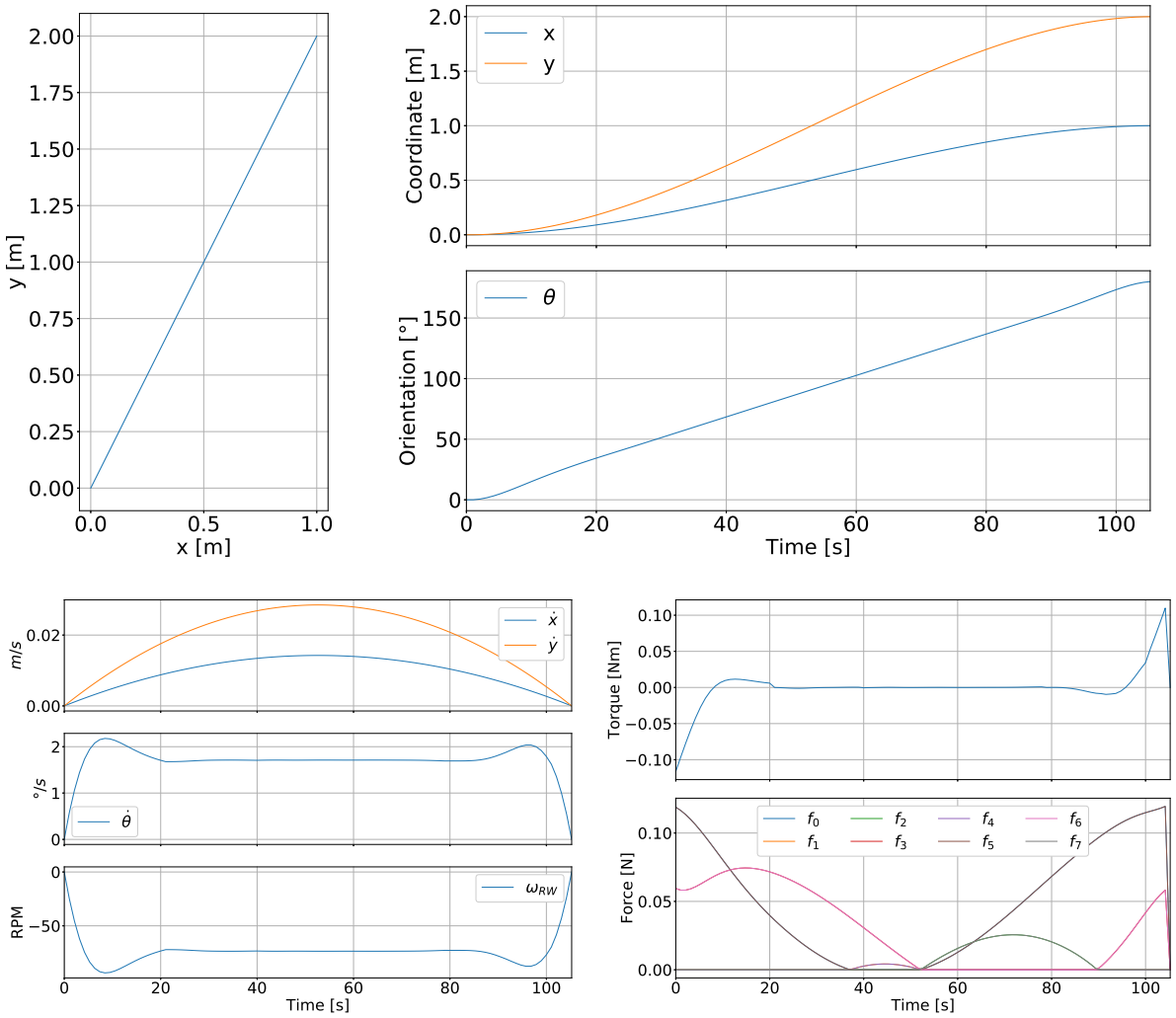


Figure 4.3: An example trajectory, resulting from the optimization scheme. It shows trajectory 1 from Table 4.1 using 100 discrete knot points.

4.2.4 Computational Burden

In the previous subsection solving an MINLP is explicitly excluded for its computational intractability. However, solving an NLP is computationally not trivial either. Since the trajectory is computed a-priori, no strict time requirements are imposed on the system. Yet, a reasonable computation time must be achieved for practicality. Figure 4.4 shows the computational time for different trajectories using different numbers of knot points. The different trajectories each connect two states outlined in Table 4.1.

For all trajectories, the computation time remains under ten seconds when using 100 knot points. Since the computational load grows exponentially the point of diminishing return is reached shortly after said 100 knot points as the computational load increases dramatically

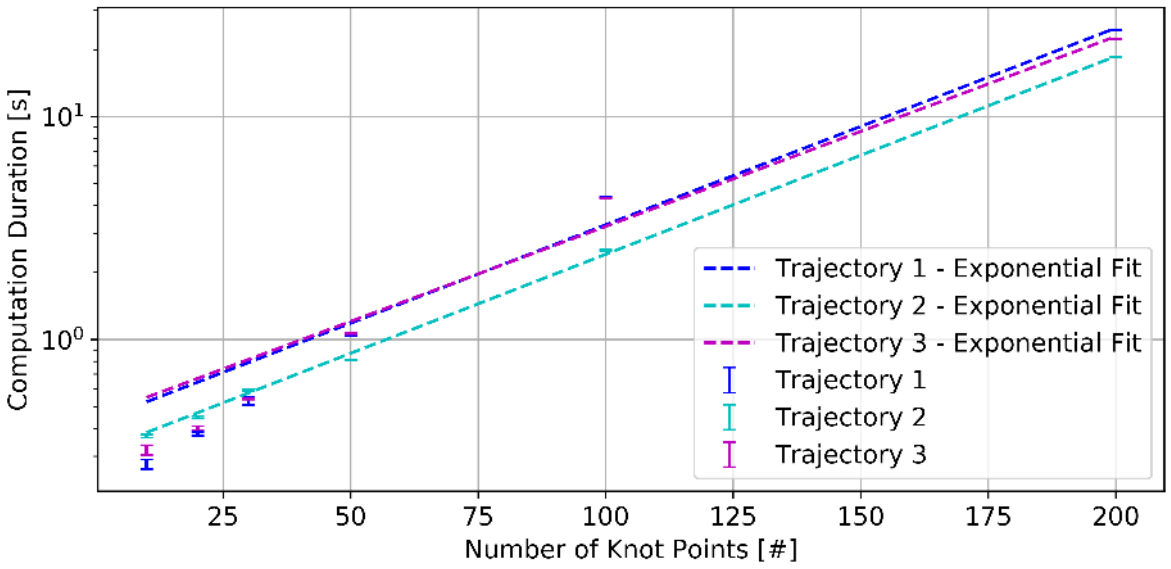


Figure 4.4: Time required to find optimal trajectories for different numbers of knot points. The trajectories correspond to the initial and final conditions in table 4.1. Each experiment (Trajectory and Number of knot points) is run 100 times. The plot shows the mean and standard deviation of each experiment.

while the gain in resolution no longer adds valuable information. Hence, in the rest of this work the trajectories are pre-computing using the same order of magnitude for N.

#	Initial Condition	Final Condition
Trajectory 1	$\begin{bmatrix}0 & 0 & 0 & 0 & 0 & 0 & 0\end{bmatrix}^T$	$\begin{bmatrix}1 & 2 & \pi & 0 & 0 & 0 & 0\end{bmatrix}^T$
Trajectory 2	$\begin{bmatrix}0 & 0 & 0 & 0 & 0 & 0 & 0\end{bmatrix}^T$	$\begin{bmatrix}0.2 & 3 & 0 & 0 & 0 & 0 & 250\ \text{RPM}\end{bmatrix}^T$
Trajectory 3	$\begin{bmatrix}1 & 1 & 0 & 0 & 0 & 0 & 0\end{bmatrix}^T$	$\begin{bmatrix}0 & 1 & -1 & 0 & 0 & 0 & 0\end{bmatrix}^T$

Table 4.1: Different initial and final conditions used to test the computational load of finding optimal trajectories in between.

4.3 Trajectory Follower

4.3.1 Time-Varying Feedback Controller

Once an optimal trajectory is computed (or any other desired trajectory of the same structure for that matter), the robot should find the controls that, given the current state, moves the robot along the desired trajectory. One could assume that simply enacting the control specified by the trajectory would be enough; however, since the system is subjected to unmodelled disturbances, this open-loop control is not sufficient.

One common approach for implementing a trajectory tracking feedback controller is that of a TVLQR – a version of the abundantly present Linear Quadratic Regulator (LQR). Let's first introduce the finite time-horizon stabilization scheme as introduced in [65]. A non-linear, time-varying system is linearised at some equilibrium point where it is to be stabilized at. This yields a time-varying dynamic system of the form:

$$\dot{\mathbf{x}} = \mathbf{A}(t)\mathbf{x} + \mathbf{B}(t)\mathbf{u} \tag{4.25}$$

The resulting controller shall minimize the finite-horizon cost-to-go function:

$$J = h(\mathbf{x}_f) + \int_0^{t_f} l(\mathbf{x}(t), \mathbf{u}(t)) = \mathbf{x}_f^T \mathbf{Q}_f \mathbf{x}_f + \int_0^{t_f} \mathbf{x}^T \mathbf{Q}\mathbf{x} + \mathbf{u}^T \mathbf{R}\mathbf{u} dt \tag{4.26}$$

with $\mathbf{Q}_f = \mathbf{Q}_f^T \succeq 0$, $\mathbf{Q} = \mathbf{Q}^T \succeq 0$ and $\mathbf{R} = \mathbf{R}^T \succ 0$. Any control policy $\mathbf{u}^*$ and its associated cost J^* is optimal if it satisfies the Hamilton-Jacobi-Bellman (HJB) equation:

$$0 = \min_u \left[l(\mathbf{x}, \mathbf{u}) + \frac{dJ^*}{dt} \right] = \min_u \left[l(\mathbf{x}, \mathbf{u}) + \frac{\partial J^*}{d\mathbf{x}} \mathbf{f}(\mathbf{x}, \mathbf{u}) + \frac{\partial J^*}{\partial t} \right] \tag{4.27}$$

$$\Rightarrow 0 = \min_u \left[\mathbf{x}^T \mathbf{Q}\mathbf{x} + \mathbf{u}^T \mathbf{R}\mathbf{u} + \frac{\partial J^*}{\partial \mathbf{x}} (\mathbf{A}(t)\mathbf{x} + \mathbf{B}(t)\mathbf{u}) + \frac{\partial J^*}{\partial t} \right] \tag{4.28}$$

The expression within the brackets is – by construction – positive definite in $\mathbf{u}$. Hence the minimum can be found by setting its derivative equal to zero:

$$0 = \frac{\partial}{\partial \mathbf{u}} \left[\mathbf{x}^T \mathbf{Q}\mathbf{x} + \mathbf{u}^T \mathbf{R}\mathbf{u} + \frac{\partial J^*}{\partial \mathbf{x}} (\mathbf{A}(t)\mathbf{x} + \mathbf{B}(t)\mathbf{u}) + \frac{\partial J^*}{\partial t} \right] = 2\mathbf{u}^T R + \frac{\partial J^*}{\partial \mathbf{x}} \mathbf{B}(t) \tag{4.29}$$

$$\Rightarrow \mathbf{u} = -\frac{1}{2} \mathbf{R}^{-1} \mathbf{B}^T(t) \frac{\partial J^{*T}}{\partial \mathbf{x}} \tag{4.30}$$

Sticking with the theme of quadratic forms we choose the cost function to be:

$$J^*(\mathbf{x}, t) = \mathbf{x}^T \mathbf{S}(t)\mathbf{x}, \quad \mathbf{S}(t) = \mathbf{S}^T(t) \succ 0 \tag{4.31}$$

$$\Rightarrow \frac{\partial J^*}{\partial \mathbf{x}} = 2\mathbf{x}^T \mathbf{S}(t) \qquad \frac{\partial J^*}{\partial t} = \mathbf{x}^T \dot{\mathbf{S}}(t)\mathbf{x} \tag{4.32}$$

Then it follows for the control input:

$$\mathbf{u} = -\mathbf{R}^{-1} \mathbf{B}^T(t) \mathbf{S}(t)\mathbf{x} = \mathbf{K}(t)\mathbf{x} \tag{4.33}$$

Plugging this back into the HJB (equation (4.28)) to find the conditions on $\mathbf{S}$:

$$0 = \mathbf{x}^T \left[\mathbf{Q} - \mathbf{S}(t)\mathbf{B}(t)\mathbf{R}^{-1}\mathbf{B}(t)^T\mathbf{S}(t) + \mathbf{S}(t)\mathbf{A}(t) + \mathbf{A}^T(t)\mathbf{S} + \dot{\mathbf{S}}(t) \right] \mathbf{x} \tag{4.34}$$

This must hold for all $\mathbf{x}$ hence we only consider the content of the bracket and arrive at:

$$\Rightarrow -\dot{\mathbf{S}}(t) = \mathbf{S}(t)\mathbf{A}(t) + \mathbf{A}^T(t)\mathbf{S} - \mathbf{S}(t)\mathbf{B}(t)\mathbf{R}^{-1}\mathbf{B}(t)^T\mathbf{S}(t) + \mathbf{Q} \tag{4.35}$$

And finally, to also satisfy the equality constraint at the final state it must hold:

$$\mathbf{S}(t_f) = \mathbf{Q_f} \tag{4.36}$$

From this the Differential Riccati Equation (DRE) (equation (4.35)) is solved by initializing $\mathbf{S}$ with its value at the final time and integrating it backwards in time to find the respective matrices $\mathbf{S}$ at different times t and from this the feedback gain matrix $\mathbf{K}$.

The main difference to extend this to the trajectory tracking case is that the time-varying system matrices are linearised about the nominal trajectory ($\mathbf{x}_0$ and $\mathbf{u}_0$) and, hence also, the coordinate system is defined as

$$\bar{\mathbf{x}} = \mathbf{x} - \mathbf{x}_0 \qquad\qquad \bar{\mathbf{u}} = \mathbf{u} - \mathbf{u}_0 \tag{4.37}$$

This results in the control law:

$$\boxed{\mathbf{u} = \mathbf{u_0}(t) + \mathbf{K}(t)(\mathbf{x} - \mathbf{x}_0(t))} \tag{4.38}$$

where $\mathbf{K}$ is computed as in equation (4.33). It should be noted, that solving the DRE backward in time, implies the controller must pre-compute the gain matrices for a trajectory beforehand yielding a slightly longer computation time in the initialization.

4.3.2 Modulator

Given that there are discrete actuators in the present system, one needs to answer the question of how to modulate the continuous control signal derived in the previous section. One common approach, that is particularly popular for microgravity simulation systems (cf. chapter 2), is Pulse-Width-Modulation (PWM). In PWM one defines a repeating set of intervals of which the actuator is active for some and non-active for the rest. E.g. if there were one hundred intervals in one second (which corresponds to a PWM frequency of 100 Hz), the actuator might be turned on for fifty of those intervals. The resulting impulse is equivalent to that of an actuator, that is active at all times but with half the propulsion force. The continuous force is therefore approximated, by fully activating the actuator a specific number of intervals.

This, however, can only approximate the continuous force at a number of discrete intervals, defined by the PWM frequency. In fact, the lower the PWM frequency, the fewer discrete levels are available. Additionally, high switching frequencies prove to be problematic for pneumatic systems, as they often resonate or have too high time delays. These can take the form of, e.g. a pressure regulator not following the demand immediately or a valve's opening and closing duration being too large. In combination, these two facts allow for only a very crude approximation

of the desired continuous force using PWM as high frequencies are not possible and thus only a few approximation levels are available.

An alternative approach introduced in [16] is using the technique of a $\Sigma\Delta$-Modulator[1], a technique commonly used in analog to digital modulation, to modulate the continuous force onto the binary actuators. The basic concept of a $\Sigma\Delta$-Modulator is to trigger a pulse as soon as the integrator error reaches some certain threshold. Figure 4.5 shows the block diagram of the $\Sigma\Delta$-Modulator, in the configuration for modulating on binary output. The procedure it follows is the following:

1. Sample the continuous signal $u(t)$ at some frequency f_{smpl} using sample and hold.

2. Compute the error $e(t)$ by taking the difference of the current output $y(t)$ (which is already modulated) to the desired value.

3. Weight the error with some gain $k_{\Sigma\Delta}$ depending on the desired sensitivity of the modulator. In this work, it is set to one.

4. Integrate the weighted error via $w_e(t) = \int_{t_0}^{t} k_{\Sigma\Delta} e(\tau) d\tau \approx \sum_{i=0}^{t} k_{\Sigma\Delta} e[i]$.

5. Once the integrator value surpasses some threshold, i.e. $w_e(t) > \epsilon$, trigger a pulse. The threshold should be chosen such that a single pulse resets the error integrator to zero, i.e. has the same area under the curve.

6. Feedback the current output value.

Figure 4.5 shows the block diagram of the modulator and an example modulation of a sinusoidal signal. One big advantage of the $\Sigma\Delta$-Modulator is that it is possible to sample the input at a different frequency as the output (and thus the feedback). By adjusting the sampling frequency of the continuous signal the integrator accumulates at some frequency. This is independent of the output read frequency which runs separately. Later in this work, this is exploited to run different actuators at different frequencies.

[1]There are competing names for this modulator: $\Sigma\Delta$-Modulator vs. $\Delta\Sigma$-Modulator. The source cited in this work uses the prior, hence so does this work. Further, the name originates from the fact that the modulator is *summing up* **differences**, therefore the order should be $\Sigma \rightarrow \Delta$.

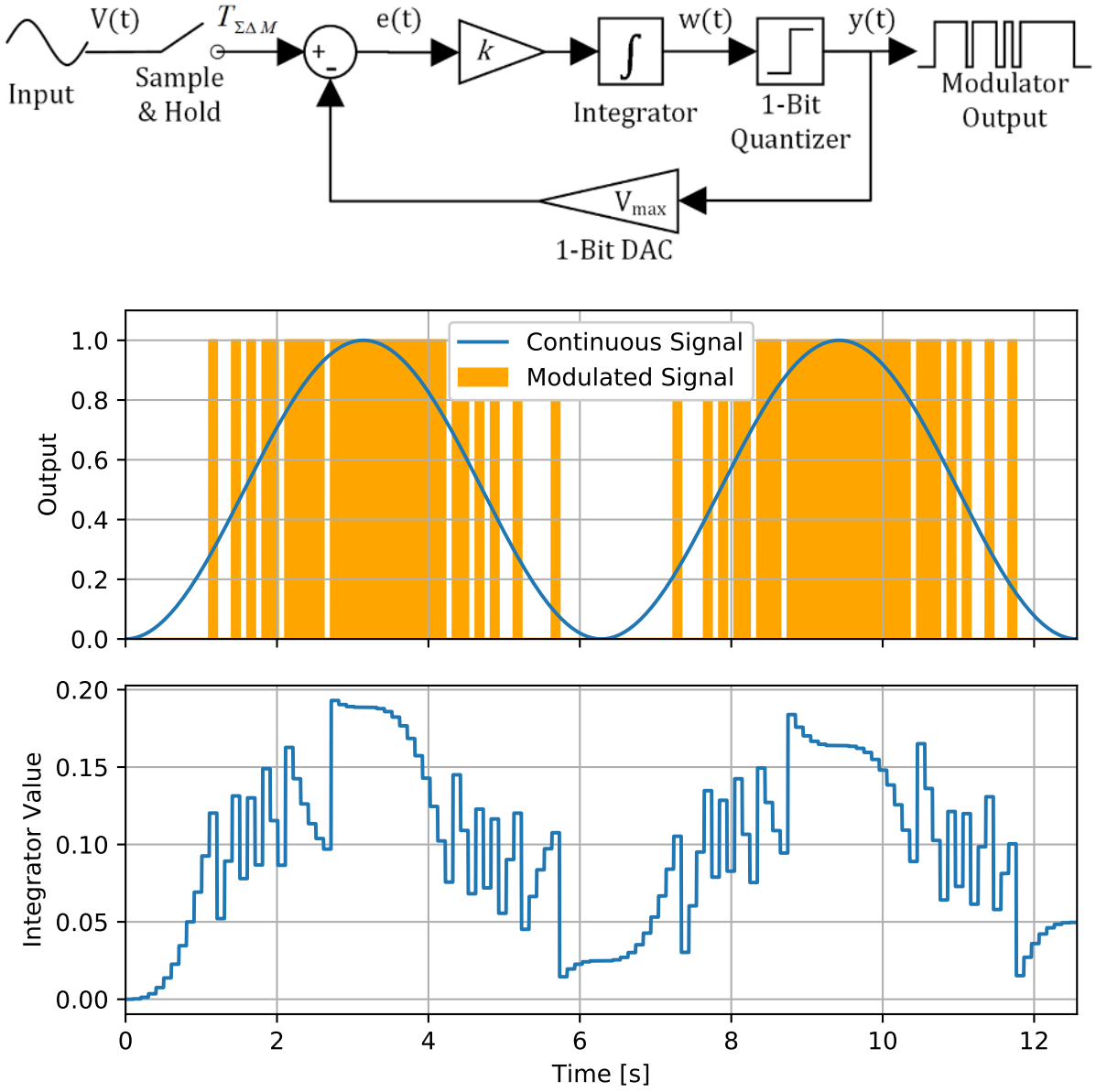


Figure 4.5: ΣΔ - Modulator. Block Diagram (top) Courtesy of Zappulla [16], sample modulation of a sinusoidal signal (middle) using $K = 1$, $f_{smpl} = 10\,\text{Hz}$ and $\epsilon = 0.1$ and the respective integrator value (bottom).

4.3.3 State-Estimation

In the previous sections it is assumed that the full state is available for full state feedback. This is not the case. The motion capture system provides only pose measurements of the system and no velocity. Literature shows that when numerically differentiating noisy pose data, even low

levels of noise lead to arbitrarily large deviations in the numerically computed derivative [66–68], making this option unsuitable.

Further, all available measurements are subjected to noise and thus are in need of some filtering process to improve their quality given some model of the underlying system. The gold standard for this filter since the sixties is the Kalman Filter (KF) [69]. By combining the knowledge of the underlying system and the measurements optimally (for linear systems), in the sense of a quadratic estimation error, the filter smooths the data and rejects harsh outliers that fall significantly outside the distribution of the specified measurement error.

The Classical Kalman Filter

In the linear case the KF [69] assumes a discrete state-space system subjected to system and measurement noise ($\mathbf{w}_k$ and $\mathbf{v}_k$) in the form:

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k + \mathbf{G}\mathbf{w}_k \tag{4.39}$$

$$\mathbf{y}_k = \mathbf{C}\mathbf{x}_k + \mathbf{v}_k \tag{4.40}$$

The KF requires the pair $(\mathbf{A}, \mathbf{C})$ to be fully observable, that all noise is additive white noise with zero mean, that all noise is uncorrelated over time (i.e. $E[\mathbf{w}_k\mathbf{w}_{k+1}^T] = E[\mathbf{v}_k\mathbf{v}_{k+1}^T] = 0 \ \ \forall k$), and uncorrelated measurement and system noise (i.e. $E[\mathbf{w}_k\mathbf{v}_k^T] = 0 \ \ \forall k$). The KF now aims to find an estimate of the current state $\hat{\mathbf{x}}_k$ such that the quadratic estimation error is minimized:

$$J_k = E[(\mathbf{x}_k - \hat{\mathbf{x}}_k)^T(\mathbf{x}_k - \hat{\mathbf{x}}_k)] \tag{4.41}$$

For this the KF proceeds in two steps: prediction and correction. During the prediction, the known system model is used to compute the expected next state given the current estimate and the commanded control input. The a-priori state estimate $\hat{\mathbf{x}}_{k+1}^-$ is then:

$$\boxed{\hat{\mathbf{x}}_k^- = \mathbf{A}\hat{\mathbf{x}}_{k-1} + \mathbf{B}\mathbf{u}_{k-1}} \tag{4.42}$$

and the corresponding expected measurement:

$$\hat{\mathbf{y}}_k = \mathbf{C}\hat{\mathbf{x}}_k^- \tag{4.43}$$

using this and the true measurement at the respective time stamp the correction step yields the estimate:

$$\boxed{\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + \mathbf{K}_k\left(\mathbf{y}_k - \mathbf{C}\hat{\mathbf{x}}_k^-\right)} \tag{4.44}$$

The task is now to find the optimal Kalman gain K that minimizes the quadratic estimation cost, i.e.:

$$K_k = \arg\min_{K_k} E[(\mathbf{x}_k - \hat{\mathbf{x}}_k)^T(\mathbf{x}_k - \hat{\mathbf{x}}_k)] \tag{4.45}$$

Since the function to be minimized is quadratic, by construction, the minimum is found by setting the derivative equal to zero:

$$0 = \frac{d}{dK_k}\left(E[(\mathbf{x}_k - \hat{\mathbf{x}}_k)^T(\mathbf{x}_k - \hat{\mathbf{x}}_k)]\right) \tag{4.46}$$

Ommiting the explicit dependence on the timestep k:

$$0 = E\left[\frac{d}{dK}\left((\mathbf{x} - \hat{\mathbf{x}})^T(\mathbf{x} - \hat{\mathbf{x}})\right)\right] \tag{4.47}$$

Using the a-prio estimation error $\mathbf{e}^- = \mathbf{x} - \hat{\mathbf{x}}^-$:

$$0 = E\left[2\left(\mathbf{KCe}^-\mathbf{e}^{-,T}\mathbf{C}^T + \mathbf{KCe}^{-,T}\mathbf{v}^T + \mathbf{Kve}^{-,T}\mathbf{C}^T + \mathbf{Kvv}^T - \mathbf{e}^-\mathbf{v}^T - \mathbf{e}^-\mathbf{e}^{-,T}\mathbf{C}^T\right)\right] \tag{4.48}$$

$$\begin{aligned}\Rightarrow 0 &= \mathbf{KC}E\left[\mathbf{e}^-\mathbf{e}^{-,T}\right]\mathbf{C}^T + \mathbf{KC}E\left[\mathbf{e}^{-,T}\mathbf{v}^T\right] + \mathbf{K}E\left[\mathbf{ve}^{-,T}\right]\mathbf{C}^T \\ &\quad + \mathbf{K}E\left[\mathbf{vv}^T\right] - E\left[\mathbf{e}^-\mathbf{v}^T\right] - E\left[\mathbf{e}^-\mathbf{e}^{-,T}\right]\mathbf{C}^T\end{aligned} \tag{4.49}$$

Since it is the covariance of the measurement noise (per definition) $E\left[\mathbf{vv}^T\right] = \mathbf{R}$. Further the mixed terms can be shown to be zero since the system and measurement noise are also uncorrelated. The equation then simplifies to:

$$0 = \mathbf{KC}E\left[\mathbf{e}^-\mathbf{e}^{-,T}\right]\mathbf{C}^T + \mathbf{KR} - E\left[\mathbf{e}^-\mathbf{e}^{-,T}\right]\mathbf{C}^T \tag{4.50}$$

Inserting the covariance of the a-priori estimation as $\mathbf{P}^- = E\left[\mathbf{e}^-\mathbf{e}^{-,T}\right]$ and re-introducing the explicit dependence on the time-step yields

$$\mathbf{K}_k(\mathbf{CP}_k^-\mathbf{C}^T + \mathbf{R}) = \mathbf{P}_k^-\mathbf{C}^T \tag{4.51}$$

And finally solving for the optimal Kalman gain:

$$\boxed{\mathbf{K}_k = \mathbf{P}_k^-\mathbf{C}^T(\mathbf{CP}_k^-\mathbf{C}^T + \mathbf{R})^{-1}} \tag{4.52}$$

From this what is left to do, is find an expression for the a-priori covariance of the estimation and its a-posteriori correction. Using the a-priori estimation error $\mathbf{e}_{k+1}^- = (\mathbf{Ax}_k + \mathbf{Bu}_k + \mathbf{Gw}_k) - (\mathbf{A}\hat{\mathbf{x}}_k + \mathbf{Bu}_k) = \mathbf{Ae}_k + \mathbf{Gw}_k$ and the definition of covariance

$$\mathbf{P}_{k+1}^- = E\left[\mathbf{e}_{k+1}^-\mathbf{e}_{k+1}^{-,T}\right] \tag{4.53}$$

$$= E\left[\mathbf{Ae}_k\mathbf{e}_k^T\mathbf{A}^T + \mathbf{Ae}_k\mathbf{w}_k^T\mathbf{G}^T + \mathbf{Gw}_k\mathbf{e}_k^T\mathbf{A}^T + \mathbf{Gw}_k\mathbf{w}_k^T\mathbf{G}^T\right] \tag{4.54}$$

$$= \mathbf{A}E\left[\mathbf{e}_k\mathbf{e}_k^T\right]\mathbf{A}^T + \mathbf{A}E\left[\mathbf{e}_k\mathbf{w}_k^T\right]\mathbf{G}^T + \mathbf{G}E\left[\mathbf{w}_k\mathbf{e}_k^T\right]\mathbf{A}^T + \mathbf{G}E\left[\mathbf{w}_k\mathbf{w}_k^T\right]\mathbf{G}^T \tag{4.55}$$

As before the mixed terms within the expectation operator vanish and the quadratic terms are replaced by the respective covariances. This yields for the prediction of the covariance matrix:

$$\boxed{\mathbf{P}_{k+1}^- = \mathbf{AP}_k\mathbf{A}^T + \mathbf{GQG}^T} \tag{4.56}$$

Proceeding similarly for the correction:

$$\mathbf{e_k} = \mathbf{x}_k - \left[\hat{\mathbf{x}}_k^- + \mathbf{K}_k\left(\mathbf{y}_k - \mathbf{C}_k\hat{\mathbf{x}}_k^-\right)\right] = \mathbf{e}_k^- - \mathbf{K}_k\mathbf{C}\mathbf{e}_k^- + \mathbf{K}_k\mathbf{v}_k \tag{4.57}$$

$$\mathbf{P}_k = E\left[\mathbf{e}_k\mathbf{e}_k^T\right] = E\left[\left(\left(\mathbf{I} - \mathbf{K}_k\mathbf{C}\right)\mathbf{e}_k^- - \mathbf{K}\mathbf{v}\right)\left(\left(\mathbf{I} - \mathbf{K}_k\mathbf{C}\right)\mathbf{e}_k^- - \mathbf{K}\mathbf{v}\right)^T\right] \tag{4.58}$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{C})\mathbf{P}_k^-(\mathbf{I} - \mathbf{K}_k\mathbf{C})^T + \mathbf{K}_k\mathbf{R}\mathbf{K}_k \tag{4.59}$$

And inserting the optimal Kalman gain:

$$\boxed{\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{C})\mathbf{P}_k^-} \tag{4.60}$$

Using all highlighted equations and following the procedure outlined above (and visualized in figure 4.6) yields the optimal observer for a linear system. This result also extends to time-varying linear systems [69] such as the one present in this work. For this, at each time step the respective time-varying matrices $\mathbf{A}_k, \mathbf{B}_k, \mathbf{C}_k, \mathbf{G}_k, \mathbf{Q}_k$ and $\mathbf{R}_k$ are used [70].

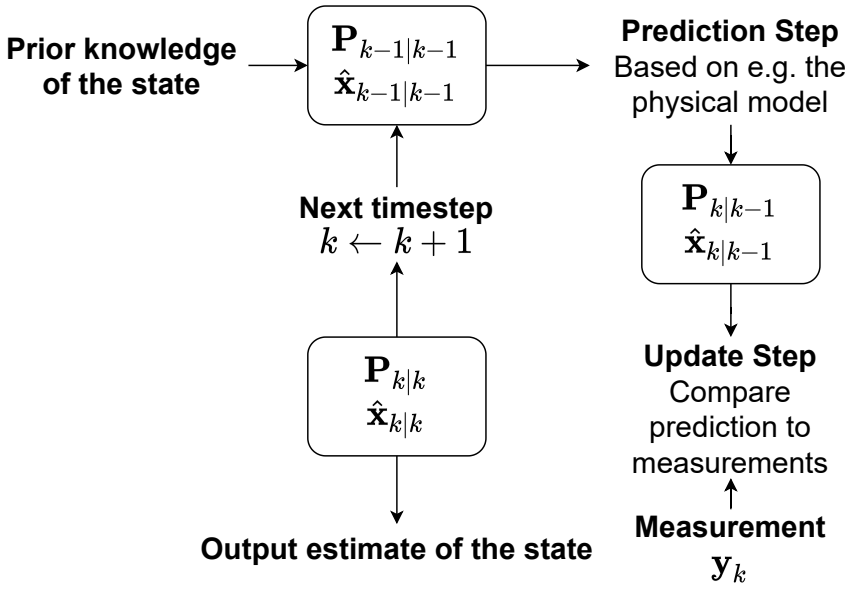


Figure 4.6: The basic concept of Kalman-Filtering. The equations outlined in this section correspond to either the prediction or correction step respectively. Based on [17]

In the present system, the classical KF is implemented as described above only for the linear position and velocity. However, since in the angular position the effect of wrapping orientation (i.e. $2\pi \Leftrightarrow 0$) is highly non-linear in euclidean space. This causes the prediction to be incorrect and the filter to fail at following the signal appropriately.

To avoid this effect, the orientation and angular velocity are estimated in a parallel KF that models the partial state-space as a Lie group in which this effect is linear.

The Kalman Filter over $\mathbf{SO(2)} \times \mathbb{R}$

A Lie group G is a group that is also a smooth manifold whose composition and inverse are smooth functions on the manifold G. Each Lie group G has an associated Lie algebra g which is an open neighborhood in the tangent space of G at the identity. The matrix exponential $\exp_G$ and matrix logarithm $\log_G$ establish a local diffeomorphism between Lie groups and Lie algebras:

$$\exp_G : G \to g, \qquad \log_G : g \to G \tag{4.61}$$

For example the two-dimensional euclidean space $\mathbb{R}^2$ is a Lie group with the composition μ and inverse i:

$$\mu_{\mathbb{R}^2} : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}^2, \mu_{\mathbb{R}^2}(\mathbf{x}, \mathbf{y}) = \mathbf{x} + \mathbf{y} \tag{4.62}$$

$$i_{\mathbb{R}^2} : \mathbb{R}^2 \to \mathbb{R}^2, i_{\mathbb{R}^2}(\mathbf{x}) = -\mathbf{x} \tag{4.63}$$

The associated Lie algebra is $g = \mathbb{R}^2$ which bridges a state on the Lie group $\mathbf{x} \in G$ with the matrix exponential and logarithm. Those operations, generally familiar to most maths practitioners, are used extensively in the previously derived structure of the Kalman filter. However, when estimating over a different Lie group, the respective operations must be used. The Lie group of all rotations in two dimensions is the Special-Orthogonal Group of Order 2 (SO(2)) in form of the rotation matrix:

$$R_\theta = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{4.64}$$

The Lie group that combines all rotations in two dimensions with the angular velocity is $SO(2) \times \mathbb{R}$, where the state is represented as:

$$\mathbf{x} = \begin{bmatrix} \mathbf{R}_\theta & \\ & \begin{bmatrix} 1 & \dot{\theta} \\ 0 & 1 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix}_{SO(2)\times\mathbb{R}} \tag{4.65}$$

It's composition and inverse are:

$$\begin{aligned} \mu_{SO(2)\times\mathbb{R}} :& (SO(2) \times \mathbb{R}) \times (SO(2) \times \mathbb{R}) \to SO(2) \times \mathbb{R}, \\ & \mu_{SO(2)\times\mathbb{R}}(\mathbf{x}_1, \mathbf{x}_2) = \begin{bmatrix} \mathbf{R}_1\mathbf{R}_2 & \\ & \begin{bmatrix} 1 & \dot{\theta}_1 + \dot{\theta}_2 \\ 0 & 1 \end{bmatrix} \end{bmatrix} \end{aligned} \tag{4.66}$$

$$i_{SO(2)\times\mathbb{R}} : SO(2) \times \mathbb{R} \to SO(2) \times \mathbb{R}, \mu_{SO(2)\times\mathbb{R}}(\mathbf{x}_1, \mathbf{x}_2) = \begin{bmatrix} \mathbf{R}^T & \\ & \begin{bmatrix} 1 & -\dot{\theta} \\ 0 & 1 \end{bmatrix} \end{bmatrix} \tag{4.67}$$

The associated Lie algebra is $so(2) \times \mathbb{R}$ and bridges the state on the Lie group to the vector $\begin{bmatrix}\theta & \dot{\theta}\end{bmatrix}^T$:

$$\begin{bmatrix}\theta \\ \dot{\theta}\end{bmatrix}_{SO(2)\times\mathbb{R}} = \begin{bmatrix}\begin{bmatrix}0 & -\theta \\ \theta & 0\end{bmatrix} & \\ & \begin{bmatrix}0 & \dot{\theta} \\ 0 & 0\end{bmatrix}\end{bmatrix}_{so(2)\times\mathbb{R}} \tag{4.68}$$

Finally, the link between the group and the link is given by the exponential and logarithmic mapping as follows:

$$\exp_{SO(2)\times\mathbb{R}}(\mathbf{x}) = \begin{bmatrix}\exp_{SO(2)}\left(\begin{bmatrix}0 & -\theta \\ \theta & 0\end{bmatrix}\right) \\ \dot{\theta}\end{bmatrix}, \quad \log_{SO(2)\times\mathbb{R}}(\mathbf{x}) = \begin{bmatrix}\log_{SO(2)}\left(\begin{bmatrix}0 & -\theta \\ \theta & 0\end{bmatrix}\right) \\ \dot{\theta}\end{bmatrix} \tag{4.69}$$

Given all necessary operations the prediction and correction step in the Kalman filter can now be expressed using the arithmetic on the Lie group as introduced by Markovic et al. in [71].

The prediction

$$\hat{\mathbf{x}}^-_{SO(2)\times\mathbb{R},k} = \hat{\mathbf{x}}_{SO(2)\times\mathbb{R},k-1} \exp_{SO(2)\times\mathbb{R}}\left(\begin{bmatrix}\begin{bmatrix}0 & -(T\dot{\theta}_{k-1} + \mathbf{B}\mathbf{u}_{k-1}) \\ T\dot{\theta}_{k-1} + \mathbf{B}\mathbf{u}_{k-1} & 0\end{bmatrix} & \\ & \begin{bmatrix}0 & \mathbf{B}\mathbf{u}_{k-1} \\ 0 & 0\end{bmatrix}\end{bmatrix}\right) \tag{4.70}$$

$$\mathbf{P}^-_{SO(2)\times\mathbb{R},k+1} = \mathbf{A}\mathbf{P}_{SO(2)\times\mathbb{R},k}\mathbf{A}^T + \mathbf{G}\mathbf{Q}\mathbf{G} \tag{4.71}$$

and the correction:

$$\mathbf{K}_k = \mathbf{P}_k\mathbf{C}^T(\mathbf{C}\mathbf{P}_k\mathbf{C}^T + \mathbf{R})^{-1} \tag{4.72}$$

$$\hat{\mathbf{x}}_{SO(2)\times\mathbb{R},k} = \hat{\mathbf{x}}^-_{SO(2)\times\mathbb{R},k} \exp_{SO(2)\times\mathbb{R}}\left(\mathbf{K}_k \log_{SO(2)\times\mathbb{R}}\left(\left[\hat{\mathbf{x}}^-_{SO(2)\times\mathbb{R},k}\right]^{-1}\mathbf{y}_{SO(2)\times\mathbb{R},k}\right)\right) \tag{4.73}$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}\mathbf{C})\mathbf{P}^-_k \tag{4.74}$$

One should note that only the prediction and correction steps directly related to the state differ from the classical KF, since only the state itself is on the Lie group. The covariance remains in the traditional space thus allowing for the same formulations of the Kalman gain and for the covariance's prediction and correction step.

4.4 Stability and Optimality

It is a well-known result that an LQR state-feedback controller is guaranteed to be stable and optimal with respect to the quadratic cost function [72]. This subsection intends to prove to the reader that the additional $\Sigma\Delta$-Modulator in the feedback-control loop does not compromise these properties.

4.4.1 Stability

Lets consider the simplistic example of a one-dimensional double-integrator with mass m and two binary actuators that enact a force in two opposing directions:

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 & 0 \\ 1/m & -1/m \end{bmatrix} u \quad \text{where } \mathbf{x} = \begin{bmatrix} q \\ \dot{q} \end{bmatrix} \quad \text{and } \mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{4.75}$$

Assuming zero-order hold the discrete state-space matrices can be obtained according to [59] via:

$$\mathbf{A} = \exp\left(\mathbf{A}_c T\right) \qquad \mathbf{B} = \int_0^T \exp\left(\mathbf{A}_c \alpha\right) d\alpha \mathbf{B}_c \tag{4.76}$$

Thus it follows for the discretized state-space system:

$$\mathbf{x}_{k+1} = \underbrace{\begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}}_{\mathbf{A}} \mathbf{x}_k + \underbrace{\begin{bmatrix} \frac{1}{2}\frac{T^2}{m} & -\frac{1}{2}\frac{T^2}{m} \\ \frac{T}{m} & -\frac{T}{m} \end{bmatrix}}_{\mathbf{B}} \mathbf{u}_k \tag{4.77}$$

When replacing the control input $\mathbf{u}$ with the LQR-feedback control law after quantization via the $\Sigma\Delta$-Modulator given the reference $\mathbf{w}$ it yields:

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{f}_{\Sigma\Delta}\left(\underbrace{\mathbf{K}\left[\mathbf{w}_k - \mathbf{x}_k\right]}_{:=\mathbf{f}_{LQR,k}}\right) \tag{4.78}$$

Where $\mathbf{f}_{\Sigma\Delta}$ is the vector-function that outputs either nominal force or zero depending on whether the respective modulator currently triggers a pulse, and its argument is the current control desired by the LQR-controller. Thus for each element i (corresponding to integrator i) follows the following function:

$$f^i_{\Sigma\Delta}(I^i) = \begin{cases} 0 & I^i \leq \epsilon \\ f_{nom} & I^i > \epsilon \end{cases} = f_{nom} \cdot \theta(I^i - \epsilon) \tag{4.79}$$

where I^i is the respective current integrator value and θ the step function. The integrator values at some timestep k are given by the expression:

$$\mathbf{I}_k = k_{\Sigma\Delta} T \sum_{j=0}^{k-1} \mathbf{f}_{LQR,j} - \mathbf{f}_{\Sigma\Delta,j} \tag{4.80}$$

or in the recursive form

$$\mathbf{I}_{k+1} = \mathbf{I}_k + k_{\Sigma\Delta}\frac{T}{m}(\mathbf{f}_{LQR,k} - \mathbf{f}_{\Sigma\Delta,k}) \tag{4.81}$$

Without loss of generality, we assume the reference state $\mathbf{w}$ to be the origin and express the modulator output in is full, discrete form:

$$\mathbf{I}_{k+1} = \mathbf{I}_k + k_{\Sigma\Delta}\frac{T}{m}(\mathbf{K}\mathbf{x}_k - f_{nom}\theta[\mathbf{I}_k - \epsilon]) =: \mathbf{f}_\mathbf{I}(\mathbf{I}_k) \tag{4.82}$$

Since a system controlled by LQR-feedback control is stable by itself, it only needs to be shown that the modulator doesn't introduce any instabilities. This is the case if the integrator value is not unstable. Intuitively this corresponds to the guarantee that the thrusters never get to a state, where they are firing indefinitely as the integrator value doesn't blow up to infinity. By analyzing the Jacobian of the discrete state-space model of the integrator (4.82) this can be determined:

$$J_I = \frac{\partial \mathbf{f}_I}{\partial \mathbf{I}} = \begin{bmatrix} 1 - k_{\Sigma\Delta}T\frac{f_{nom}}{m}\delta[I_k^1 - \epsilon] & 0 \\ 0 & 1 - k_{\Sigma\Delta}T\frac{f_{nom}}{m}\delta[I_k^2 - \epsilon] \end{bmatrix} \tag{4.83}$$

From this, the argument for stability goes as follows: For all times k where $\mathbf{I}_k \neq \epsilon$, the discrete Dirac-delta function vanishes, and the real part of the eigenvalue (in this case the matrix-diagonal elements) is equal to one i.e. borderline stable. This corresponds to the case that the thrusters are not firing, i.e. the system will simply remain on any trajectory it is currently on. For the times when $\mathbf{I}_k = \epsilon$, the discrete Dirac-delta function takes on the value one. Hence, for any combination of $k_{\Sigma\Delta}$, $\frac{f_{nom}}{m}$ and T such that

$$k_{\Sigma\Delta}T\frac{f_{nom}}{m} \overset{!}{<} 2 \tag{4.84}$$

the stability condition is satisfied. With $k_{\Sigma\Delta} = 1$ this implies that during one control interval the system can experience a change in velocity up to $2\,\mathrm{m\,s^{-1}}$ before becoming unstable. For the exemplary values $T = 0.1\,\mathrm{s}$, $m = 221\,\mathrm{kg}$ and $f_{nom} = 2 \cdot 10.36\,\mathrm{N}$ this is the case as $T\frac{f_{nom}}{m} = 0.0094\,\mathrm{m/s^2}$, providing a comfortable margin.

Therefore, since the LQR controller is stable and the modulator does not introduce instability, the example double integrator system with a modulator in the loop is stable. This generalizes to the presented system as it is a superposition of double integrator systems in x and y direction using multiple parallel thrusters that can be each modeled as one.

4.4.2 Optimality

The modulator in the loop does not compromise the optimality of the system if the integral over the desired continuous force over some time duration is equal to the integral of the modulated force, i.e.

$$\int_{t_0}^{t_1} f(t)dt \overset{!}{=} \int_{t_0}^{t_1} f_{\Sigma\Delta}(t)dt \tag{4.85}$$

This condition for optimality holds, since the function is optimal and therefore, so must be its integral. But since we map f and $f_{\Sigma\Delta}$ to $[0, 1]$ in terms of no- and maximal force, the actuation time of the $\Sigma\Delta$-Modulator is optimal if its integral is equal to the optimal one. It is assumed

that t_0 and t_1 correspond to time instances that are multiples of the minimum pulse duration of the thruster. Since the threshold that triggers a pulse is chosen such that a single pulse equalizes the integral of the input and the modulator output, the integrals are equal immediately after a pulse. But since t_0 and t_1 are multiples of the minimum pulse duration, t_2 must always be a point in time at which a pulse has just occurred. Therefore the integrals must always be equal, and the modulator does not compromise the optimality of the system. However, the assumption of havíng time instances being multiples of the minimum pulse only holds for very small minimum pulses, i.e., high thruster control frequencies. Hence the smaller the control frequency, the more the system deviates from optimality.

4.5 Frequencies

Generally speaking, increasing the frequency also improves the performance of estimators and controllers, and running at a lower frequency is usually only done to decrease the computational burden. Table 4.2 shows the frequencies used for this architecture throughout the rest of this work. One should note that the estimator and the torque control run at the highest possible frequency as the Motion-Capture (MoCap) system limits the architecture to 100 Hz. The force/thruster control, however, runs at a magnitude slower. Since the force control influences the opening and closing times of the thrusters a minimal interval must be respected to allow for the thrusters to fully open and close again. Therefore, while the integrator in the $\Sigma\Delta$-Modulator is accumulated at 100 Hz as all control values, the output is only sampled at 10 Hz.

KF	Torque - Control	Force - Control
100 Hz	100 Hz	10 Hz

Table 4.2: Estimation and Control frequencies of the architecture.

Chapter 5

Evaluation in Simulation

Before evaluating the control architecture on the physical system, this work tests the controller in a simulation. The simulation allows for faster prototyping and pre-validation before testing the controller on the physical system. Different simulators have different benefits and disadvantages; hence, choosing one always involves making some trade-offs. Usually, this involves precision, computational load, visualization, and ease of use, among other things.

5.1 Simulation

5.1.1 Choice of Simulator

The controller is implemented using C++ [73] and ROS2 [74]. Hence the simulator of choice should integrate well with the requirements of precision, computational load, visualization, and ease of use. Further, it is desirable that the simulator quickly runs on an average computer with reasonable speed and visualizes the process. Finally, the unevenness of the flat-floor should be integrable into the framework. Table 5.1 gives an (incomplete) overview of robotics simulators.

	Gazebo	Drake	CoppeliaSim	Webots
C++	Yes	Yes	Yes	Yes
ROS2	Yes	Yes[1]	Yes	Yes
Precision	+	++	+	+
Computational Load	-	+	0	0
Visualization	++	+	+	+
Uneven Floor	Yes	Yes[2]	Yes	Yes

Table 5.1: An (incomplete) overview of robotics simulators and their capabilities of interest for simulating the presented system [18, 19].

From this and the author's previous experience Gazebo [75] is chosen as the simulator. In particular, its simple implementation of a height-map in the form of a simple greyscale image file is an advantage. Gazebo uses the Open Dynamics Engine [76] by default as a physics engine that simulates the system behavior. The software structure allows adding custom plug-ins that subscribe to the ROS2 topics, forward the commands to the physics engine, and finally publish the results again on ROS2 topics. The model derived in chapter 3 is then implemented using a Simulation Description Format (SDF) file and put into a world that emulates the Orbital Robotics and GNC Lab (ORGL). Figure 5.1 shows a screenshot of the simulation. The ROS2 software structure consists of numerous nodes and its communication network of various topics, services, and actions representing the controller, simulation, and physical system as visualized in Figure 5.2.

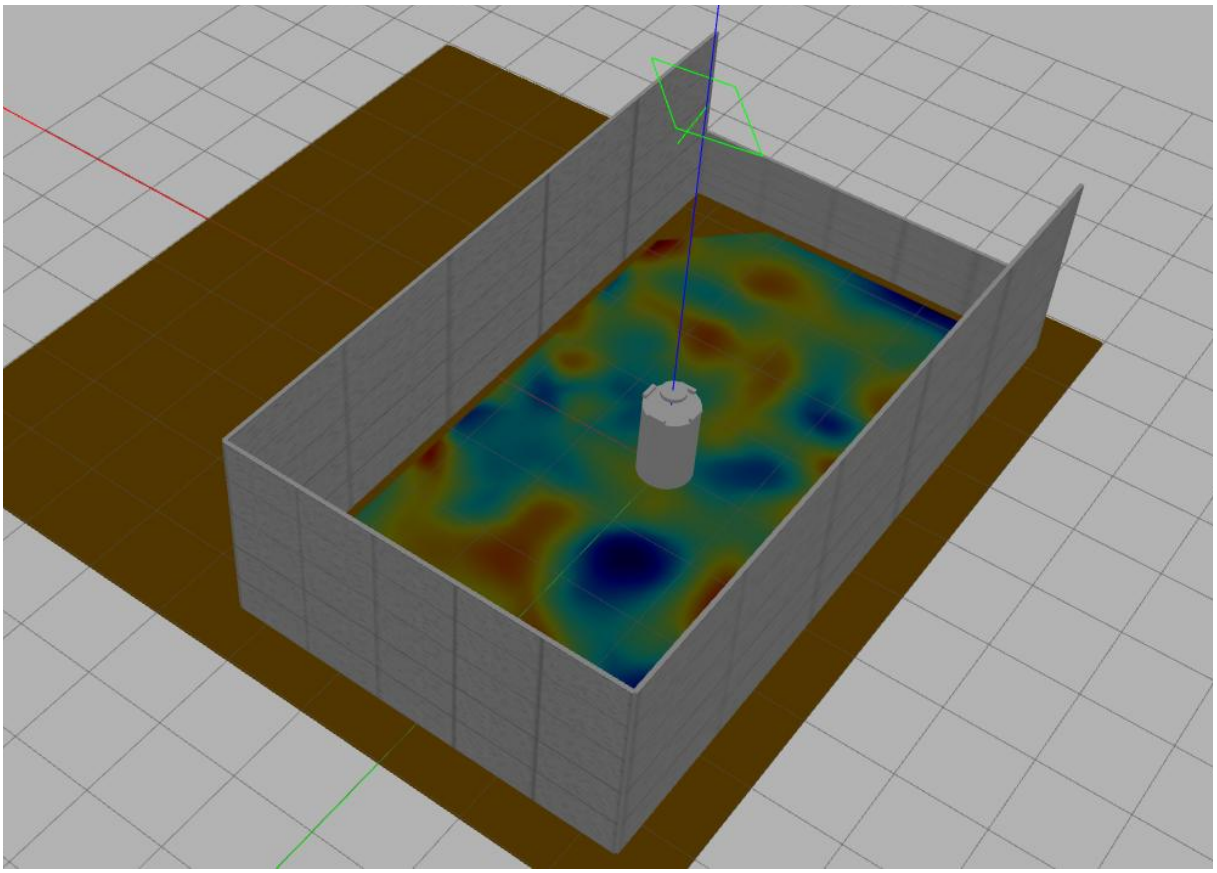


Figure 5.1: Screenshot of the simulated ORGL including the system. The simulated model of the floating platform is placed at the origin of the flat-floor. It consists of a cylindrical chassis, rectangular blocks representing the thrusters, and a disk representing the Reaction-Wheel (RW). The flat-floor is represented by its height-map where blue colors imply low values and red colors high values. Finally, the entire setup is surrounded by walls, and a light source (green square) is placed above the origin.

[1] however, less natively
[2] Beta

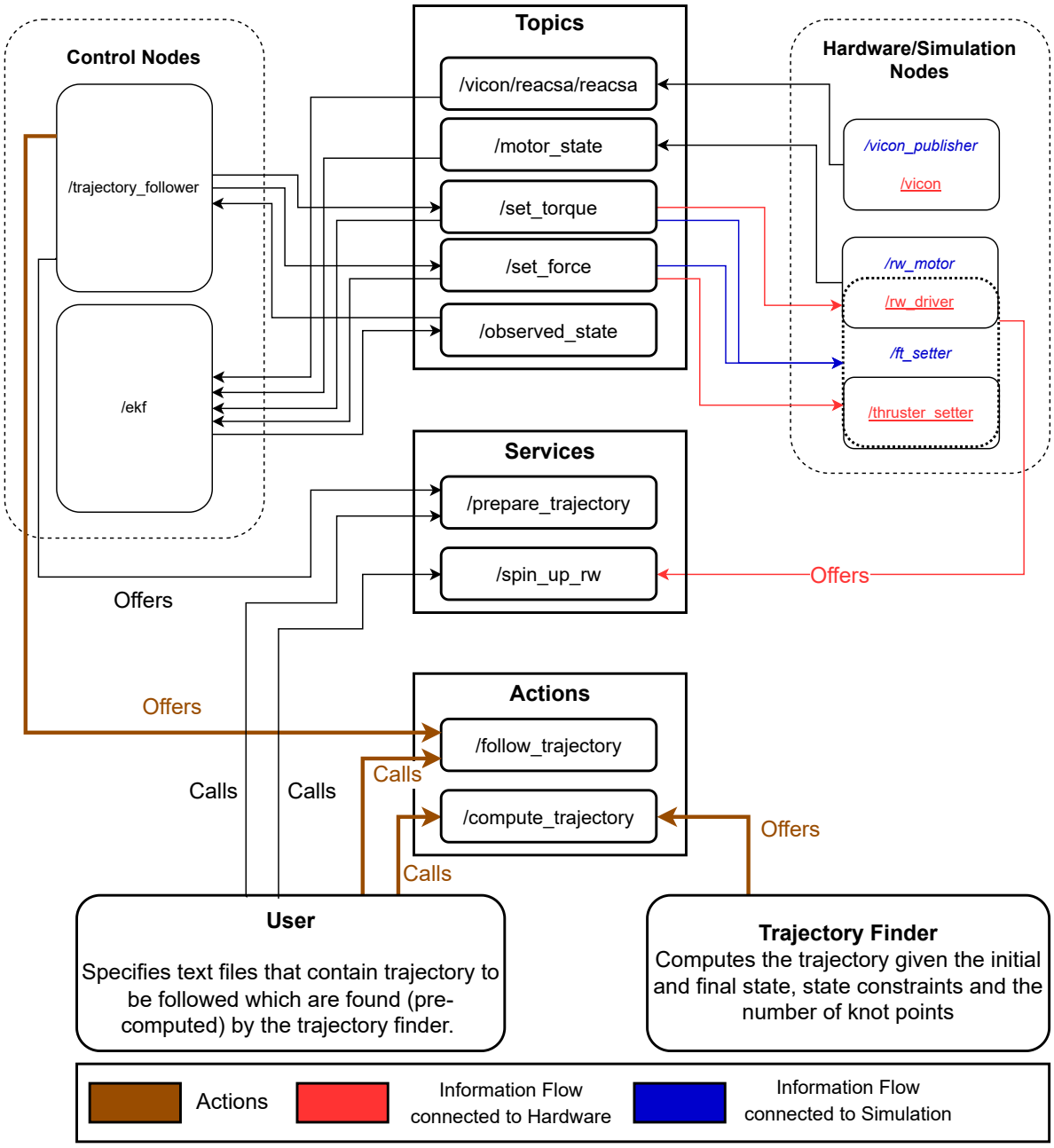


Figure 5.2: Nodes, topics, and actions within the developed ROS2 software stack. Arrows ending in a topic imply the respective node publishes to the topic. Arrows ending in a node imply the respective node subscribes to this topic. Arrows related to services and actions are labeled to indicate which node offers the service and which node calls it. The simulated nodes emulating the hardware are grouped with their respective real component/driver where blue node names imply simulated nodes and red node names imply hardware nodes/drivers.

5.1.2 Real World Approximations

Simulating the real system with more fidelity requires the unevenness of the flat-floor to be considered in the Gazebo simulation. From recent measurements of the ORGL [11] a heightmap, which is shown in Figure 5.3, is included.

Further, all sensors are subject to different levels of noise. In particular, since the pseudo measurement of the velocities is the numerical differentiation of the Motion-Capture (MoCap) pose measurements, it is essential to simulate any imprecision that is also present in the true MoCap system. For this, the simulated pose is subjected to Additive White Gaussian Noise (AWGN). In the case of the position, this adds noise sampled from a dual-variate Gaussian distribution to the measurements. The orientation is multiplied by a quaternion equivalent to a rotation around the z-axis. The rotation angle results from drawing from a Gaussian distribution. Also, this adds a randomly sampled value to the measurement of the current RW velocity.

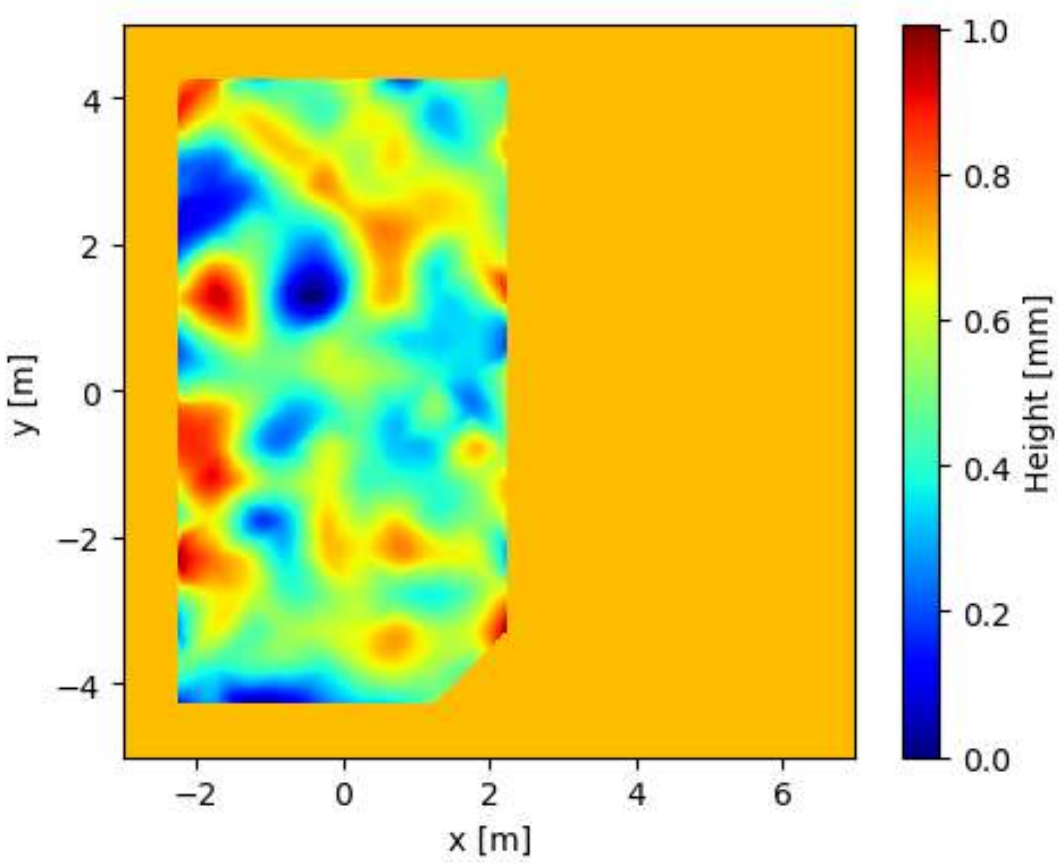


Figure 5.3: Height-map of the flat-floor. Gazebo requires a square height map, whereby the pixel side lengths must be a binary exponential plus one. Therefore, the height-map is chosen for enough resolution without deprecating the simulation speed, given that the simulation time increases exponentially with resolution. The overall size is 513 px × 513 px and height-values are discretized by one byte, i.e. 256 values. All values outside the walls of the ORGL are set to some arbitrary constant value.

5.2 Kalman Filter Evaluation

The two Kalman Filter (KF) running in parallel estimate the pose, twist, and current RW velocity. Figure 5.4 shows an example of a strongly exaggerated noise level. The KF (orange) follows the ground truth (black) very well in all states despite receiving only very noisy data (blue). The filter also converges to the ground truth, even when having a large initial offset as seen in the RW speed. Further, the filter running on $SO(2) \times \mathbb{R}$ follows the angle wraparound, justifying its usage.

The following compares the average error in position and orientation of the observer and the raw data to quantify the KFs performance. The average errors are computed as standard deviations via:

$$e_x = \sqrt{\frac{1}{N}\sum_{k=0}^{N}(x_k - \hat{x}_k)^2} \tag{5.1}$$

$$e_y = \sqrt{\frac{1}{N}\sum_{k=0}^{N}(y_k - \hat{y}_k)^2} \tag{5.2}$$

$$e_{|x,y|} = \sqrt{e_x^2 + e_y^2} = \sqrt{\frac{1}{N}\sum_{k=0}^{N}\left[(x_k - \hat{x}_k)^2 + (y_k - \hat{y}_k)^2\right]} \tag{5.3}$$

$$e_\theta = \sqrt{\frac{1}{N}\sum_{k=0}^{N}\left(\theta_k - \hat{\theta}_k\right)^2} \tag{5.4}$$

where all variables denoted with a hat are the estimates, while the ground truth is without. Table 5.2 summarizes the results for raw data and the observer. As expected, the average errors for the individual coordinates of the raw data are precisely the square root of the defined variance of the AWGN. When using the observer, the average errors in the individual coordinates are each an order of magnitude less than the raw data. In particular, the average euclidean drops from centimeter to millimeter precision, making positional estimation errors negligible.

	e_x	e_y	$e_{\|x,y\|}$	e_θ
Raw Data	0.0315 m	0.0318 m	0.0447 m	1.80°
KF	0.002 52 m	0.002 53 m	0.003 57 m	0.199°

Table 5.2: Average error of the raw data and the KF in simulation while being subjected to exaggerated AWGN ($\sigma_x^2 = \sigma_y^2 = 0.001\,\mathrm{m}^2$ and $\sigma_\theta^2 = 0.001\,\mathrm{rad}^2$)

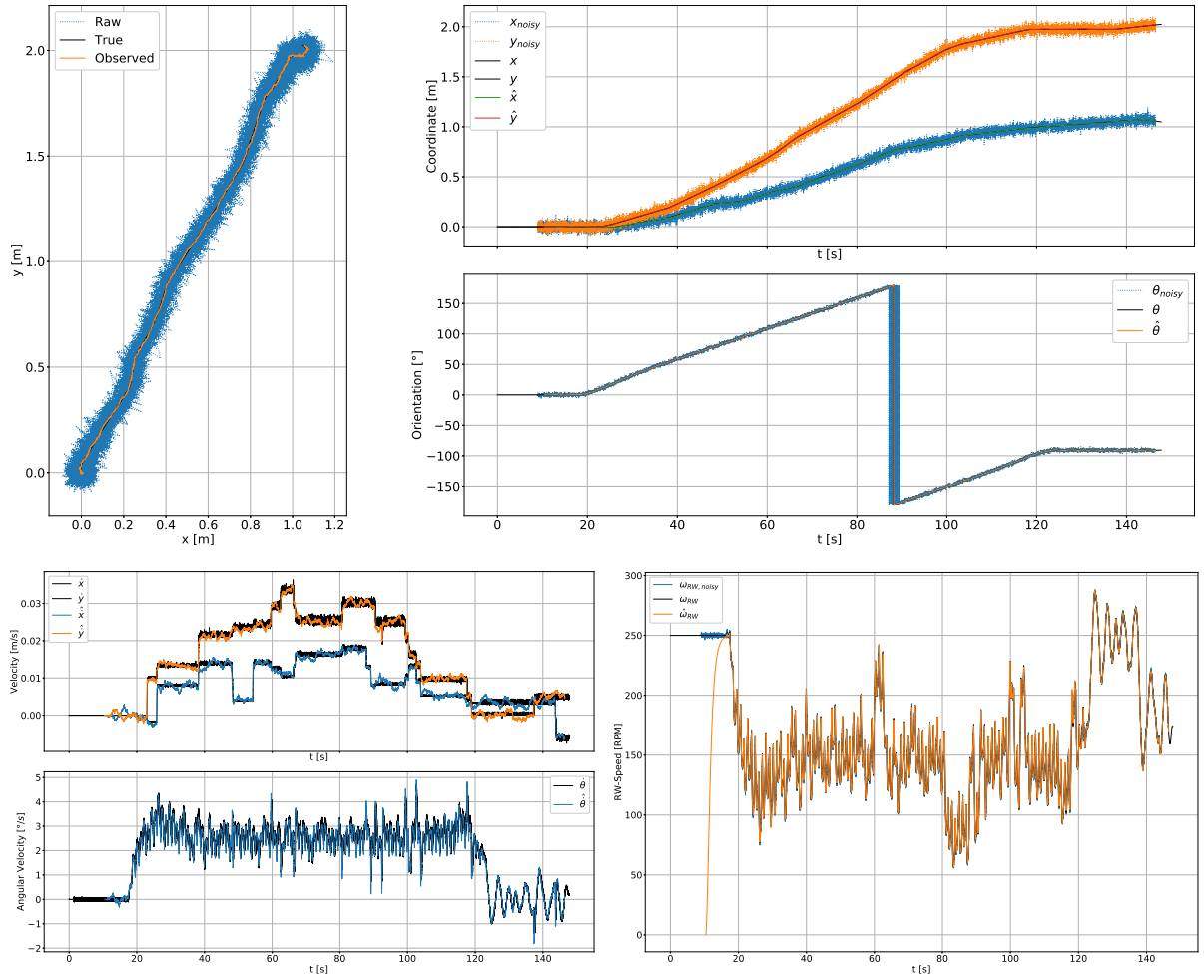


Figure 5.4: Example of the KF following the true state, given noisy measurements (cf. Table 5.2). ground truth is obtained from simulation; in particular, the velocity ground truth is given by a sliding window numerical differentiator from the ground truth position.

5.3 Controller Evaluation

For the observer and for the controller the weight matrices are tuned to yield the best performance in simulation. The matrices can be found in appendix A. The sensor properties are modeled to match the ones of the real system. Their variances are summarized in Table 5.3.

5.3.1 Tested Trajectories

For the controller, the trajectories can differ in their complexity and difficulty levels. This subsection introduces different trajectories used to evaluate the controller. In particular, it summarizes the tests performed, while the subsequent subsection provides a more detailed analysis of the results.

σ_x^2	σ_y^2	σ_θ^2	$\sigma_{\omega_{RW}}^2$
$0.000\,01\,\mathrm{m}^2$	$0.000\,01\,\mathrm{m}^2$	$0.000\,01\,\mathrm{rad}^2$	$0.0001\,(\mathrm{rad/s})^2$

Table 5.3: Variances of AWGN for each sensor in the simulation used for evaluation of the controller. The variances are chosen to match the measured variances of the real system.

Stabilization

The minimal requirement for the controller is that it stabilizes the system at some equilibrium. Without loss of generality, the stabilization is tested at the origin. The system is simulated while being subjected to an instantaneous disturbance wrench $\mathbf{d} = \begin{bmatrix}\mathbf{f}_d & \tau_d\end{bmatrix}$ consisting of a force and a torque and lasting for Δt, where:

$$\mathbf{f}_d = \begin{bmatrix}5000\,\mathrm{N}\\ 5000\,\mathrm{N}\end{bmatrix} \quad \tau_d = 1000\,\mathrm{N\,m} \quad \Delta t = 0.001\,\mathrm{s} \tag{5.5}$$

The response is shown in Figure 5.5 and the actuation commanded by the controller in Figure 5.6. Accompanying each plot of the controlled system's response is a plot of the system's response to the disturbance without being controlled to highlight the effect of the controller.

Straight-Line Trajectory

The straight-line trajectory is the same as introduced in subsection 4.2.3. Figures 5.7 and 5.8 show the results of the controller following the straight line on a perfectly flat floor and on an uneven floor.

Circular Trajectory

Another trajectory used to evaluate the control system is a complete circle around the point $\mathbf{p} = \begin{bmatrix}0 & 1\end{bmatrix}^T$, whereby the system always points to the center of the circle. This trajectory consists of 40 points evenly distributed on the circle, each connected by an optimal trajectory. The trajectory planner enforces that the system has the same tangential velocity at all points except for the initial and final state, where all velocities are zero. Further, it enforces that the RW velocity at each of the 40 points is zero. Figure 5.9 shows the resulting optimal trajectory. Figures 5.10 and 5.11 show the results of executing the trajectory in simulation on an even and uneven ground.

S-Shape Trajectory

Similar to the circular trajectory, the controller is also evaluated on an "s"-shaped trajectory. For this, the trajectory planner finds optimal trajectories between the states on the "s"-shape. It enforces that the velocity in y-direction remains constant while the velocity in x varies sinusoidally. Additionally, the orientation is held constant at 0°. Figure 5.12 shows the resulting optimal trajectory. Figures 5.13 and 5.14 show the results of executing the trajectory in simulation on an even and uneven ground.

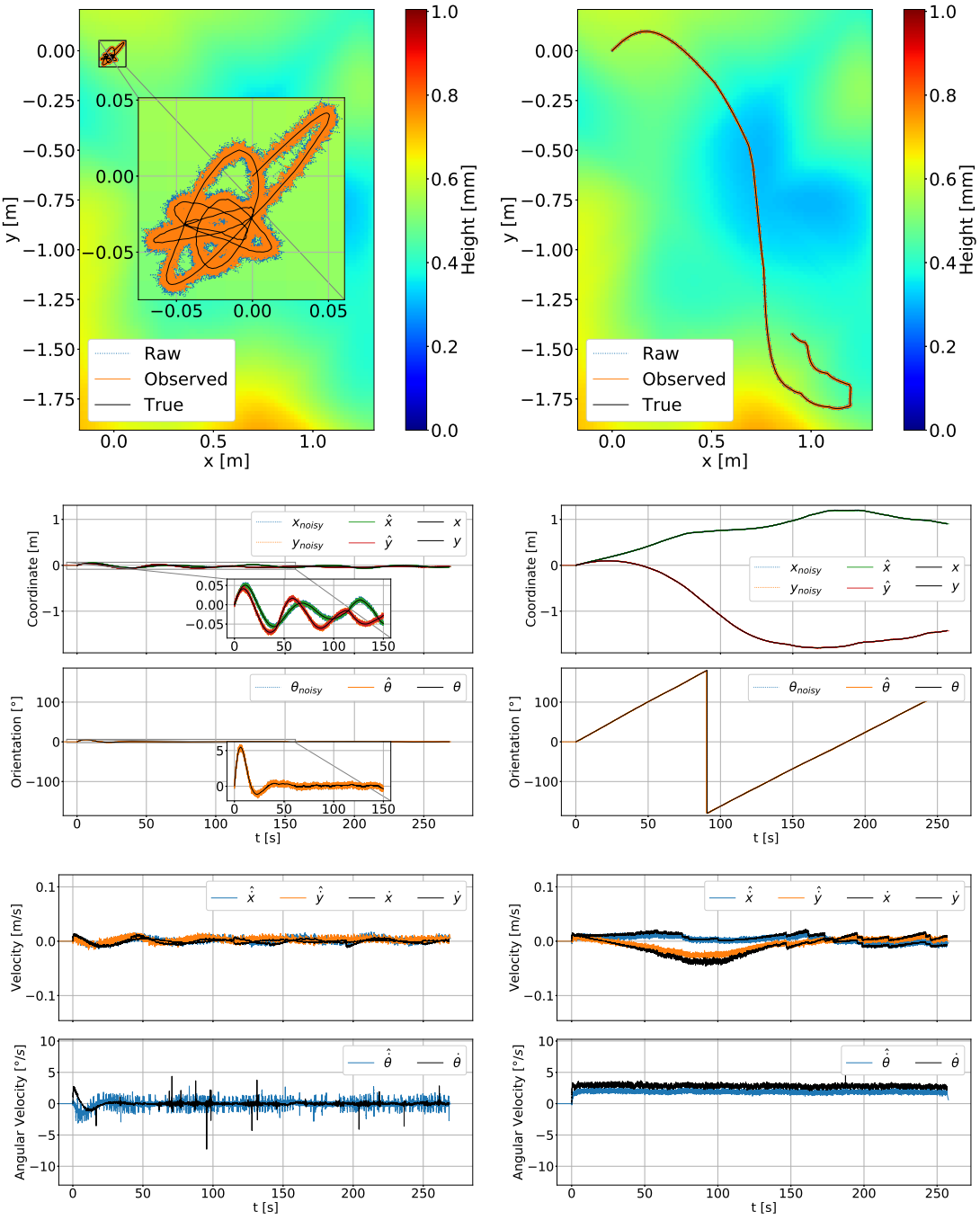


Figure 5.5: Ground-track, individual coordinates and velocities of the system responding to a disturbance (immediately at $T = 0\,\mathrm{s}$) on an uneven floor. Left: Controller stabilizing the system at the origin. Right: No controller running. The system keeps floating along the trajectory put on by the disturbance slowly converging to a local height minimum on the floor. An animation of the stabilization process is given at https://youtu.be/1A5xJVEAU9w?t=2.

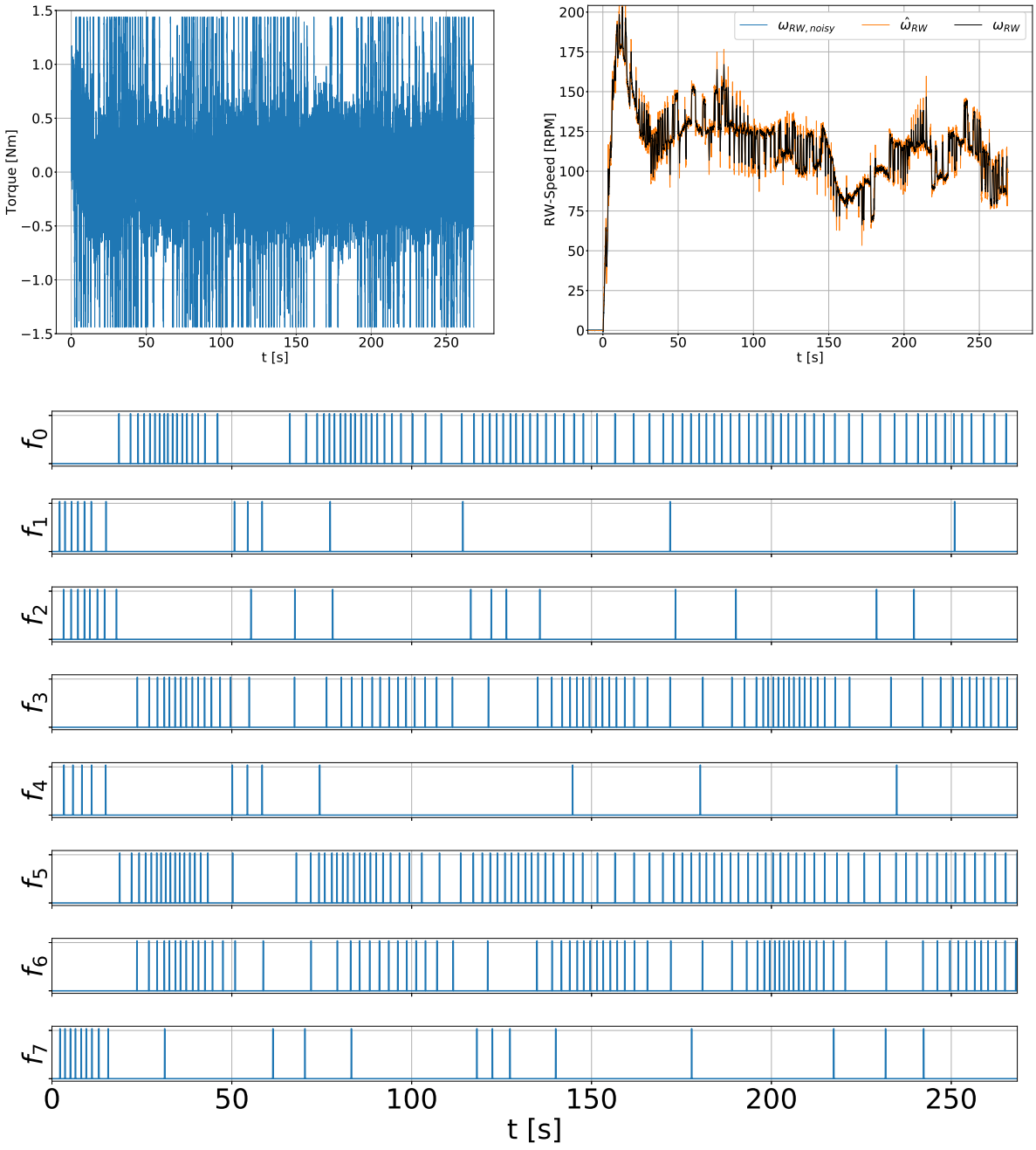


Figure 5.6: Actuation required to stabilize the system at the origin on an uneven floor while being subjected to a disturbance (cf. figure 5.5). The controller resorts to "Bang-bang" control in the torque commands to quickly compensate for the added angular momentum. Since the RW has much less inertia than the overall system its velocity changes very fast.

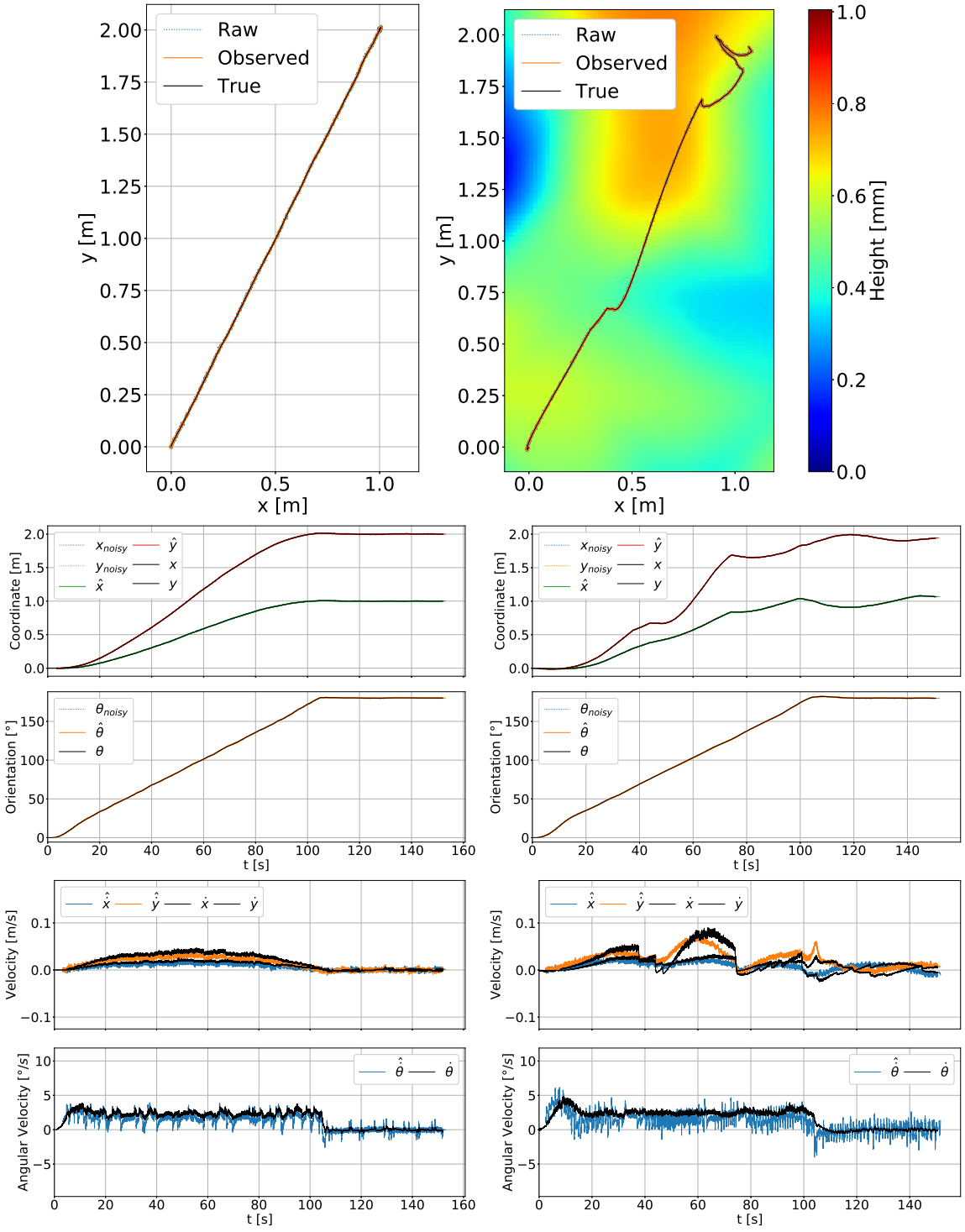


Figure 5.7: Ground-track, individual coordinates, and velocities of the system following a straight-line trajectory. Left: ideally flat floor. Right: the floor is slightly uneven and induces disturbances. An animation of the trajectory is given at https://youtu.be/1A5xJVEAU9w?t=27.

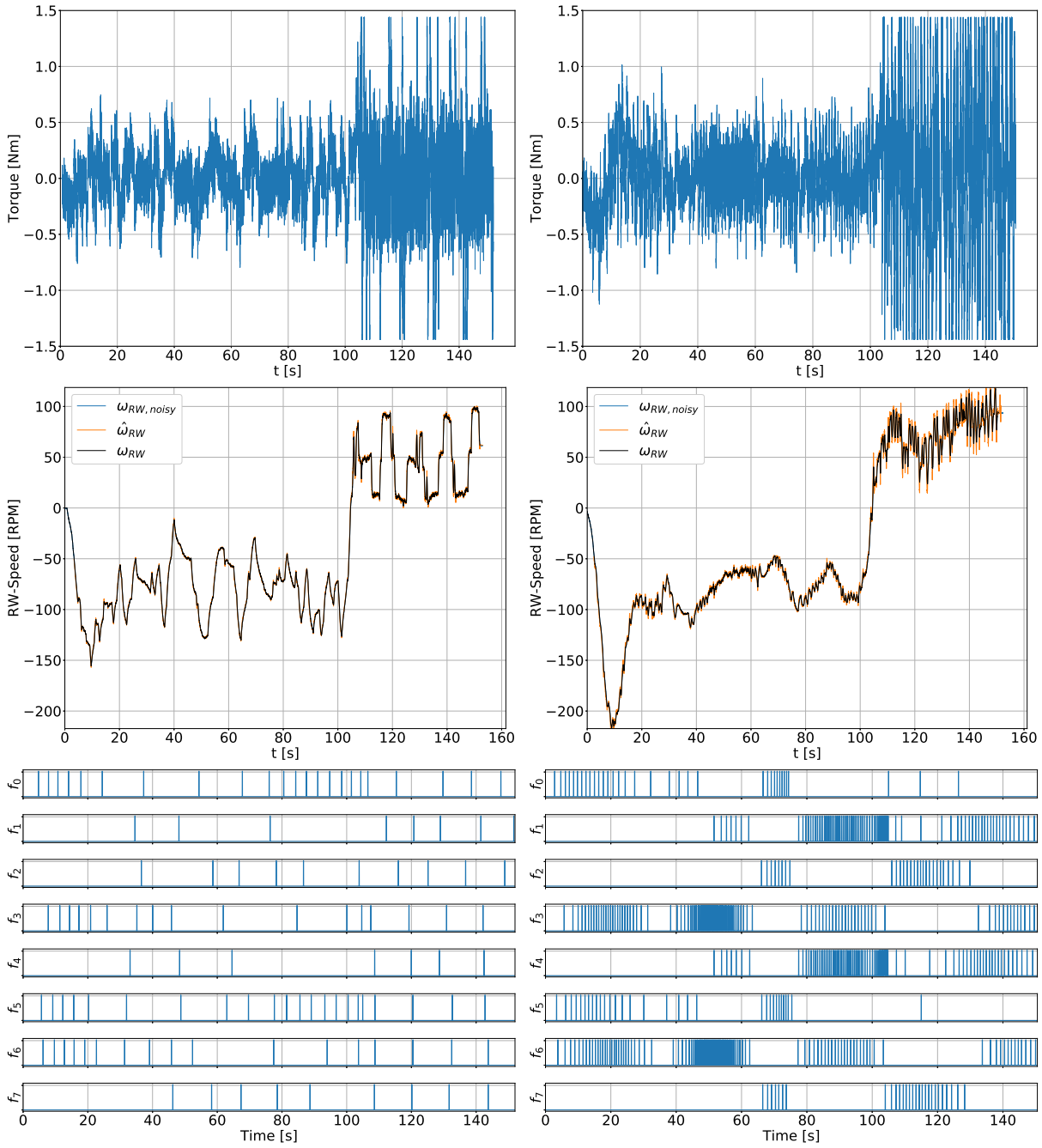


Figure 5.8: Actuation needed to follow a straight-line trajectory (cf. figure 4.3), while being subjected to noise in the state measurements. From top to bottom: commanded torque, RW velocity, and thruster firings. Left: The trajectory is followed on a perfectly flat floor. Right: the floor is slightly uneven and induces disturbances.

5.3.2 Monte Carlo

A Monte Carlo test simulation is executed to further convince the reader of the generalizability of the controller to different initial locations on the flat-floor. For a large ($n = 100$) number of episodes, the robot spawns at a random initial pose whereby x, y, and θ result from drawing from uniform random distributions that span the range $[-2, 2]$, $[-4, 4]$ and $[-\pi, \pi]$ respectively. The controller computes the optimal trajectory to the origin from the randomly drawn initial state and starts following it. An episode is considered a success when the euclidean distance to the origin, the euclidean velocity as well as the angular error and velocity are smaller than the threshold ϵ:

$$\epsilon = \begin{bmatrix} \epsilon_{linear} \\ \epsilon_{angular} \\ \epsilon_{linear,vel} \\ \epsilon_{angular,vel} \end{bmatrix} = \begin{bmatrix} 0.05\,\mathrm{m} \\ 0.05\,\mathrm{m\,s^{-1}} \\ 0.05\,\mathrm{rad} \\ 0.05\,\mathrm{rad\,s^{-1}} \end{bmatrix} = \begin{bmatrix} 0.05\,\mathrm{m} \\ 0.05\,\mathrm{m\,s^{-1}} \\ 2.9^{\circ} \\ 2.9\,^{\circ}\,\mathrm{s^{-1}} \end{bmatrix} \tag{5.6}$$

Figure 5.15 shows the results of the Monte Carlo test.

5.4 Results and Discussion

While having limited transferability to the real system due to the ever prevalent Sim-to-Real Gap [77] it provides a good approximation of the upper limit of the performance of an ideal scenario. Further, it allows comparing the controller in different scenarios that might not be physically feasible, such as the perfectly flat floor.

Stabilization The controller fulfills the minimum requirement of stabilizing the system at equilibrium. As Figure 5.5 shows, the controller manages to bring the system back to the origin after the initial disturbance. It holds the system's position within less than 10 cm and 6° of the origin. The unactuated system, on the other hand, drifts over 1.5 m and does more than a full rotation during the same time. The controlled system's offset is attributed to two factors: the thrusters' binary nature and the floor's unevenness. The error in position needs to accumulate until a firing is triggered, which in turn will propel the system through the origin, and the process repeats in the opposite direction. At the same time, the controller needs to compensate for the disturbances induced by the flat-floor, further adding to the oscillations about the origin. In Figure 5.6 the torque clearly shows "Bang-Bang" behavior. Since the RWs inertia is two orders of magnitude smaller than that of the system, the controller demands high torques such that it quickly jumps between its maximal and minimal torque. The RW velocity, however, remains within its allowable bounds. These fast changes in commanded torque are difficult for the physical system to follow. In the same Figure 5.6, the thrusters initially show alternating behavior, which corresponds to the small oscillations about the origin. Later in the simulation, the thrusters regularly fire to move the system towards positive y-direction (Thruster 0 and 5) and slightly less frequently towards positive x-direction (Thrusters 3 and 6), which counteracts the slope of the floor at the origin.

	Straight Line		Circular		"S"-Shaped	
Uneven Ground	Without	With	Without	With	Without	With
e_x	0.0216 m	0.0531 m	0.0238 m	0.0566 m	0.0163 m	0.0469 m
e_y	0.0430 m	0.147 m	0.0224 m	0.0506 m	0.0113 m	0.0573 m
$e_{\|x,y\|}$	0.0481 m	0.156 m	0.0327 m	0.0759 m	0.0198 m	0.0740 m
e_θ	3.42°	3.48°	4.52°	4.36°	0.222°	0.136°

Table 5.4: The average error of the system trying to follow a trajectory for the individual coordinates and the euclidean distance. Both cases, with and without an uneven floor, are shown. The error is computed from the desired trajectory and the ground-truth pose obtained from the simulation.

Trajectories For the trajectories the following uses the same performance metrics as for the KF, the average error as defined in equations (5.1) to (5.4). Table 5.4 summarizes the average error in the individual coordinates and the euclidean distance for the different trajectories in the different scenarios. It shows that on an ideally even floor, the controller follows all trajectories with centimeter accuracy and a pointing error of less than 5°. The average euclidean error approximately doubles with the uneven floor, whereas the angular error remains in the same order of magnitude. The system model has a single-cylinder as the main chassis, which experiences disturbance forces due to the uneven floor. However, it experiences no disturbance torques since the cylinder has a single contact to the uneven floor, thus not influencing the average pointing accuracy. The most significant disturbance occurs for the straight-line trajectory. The robot slides down a local height maximum and has to correct aggressively, causing the increased average euclidean error.

The error measurements are consistent with Figures 5.7, 5.10, and 5.13. They show that the trajectories are followed similarly on the perfectly even and uneven floor. The deviations on the uneven floor occur when surpassing steeper slopes of the flat-floor. In all trajectories, the controller commands a fast switching torque (cf. Figures 5.8, 5.11, 5.14). Similar to the stabilization case, this is hard to follow on the physical system as the underlying motor controller cannot provide the demand value instantaneously. During all trajectories on the uneven floor, extended periods of nearly continuous firing occur when overcoming slopes. As is shown in chapter 3 the force exerted by a thruster decreases after some time. Thus, making these periods difficult on the physical system.

Since the integral of the continuous optimal force computed by the Time-Varying Linear Quadratic Regulator (TVLQR) is equivalent to the optimal thruster actuation after modulation, the equivalent on-time for the optimal trajectory calculates as the area under the continuous force curve divided by the nominal force. Table 5.5 shows these optimal thruster on-times and the thruster on-times for the simulated trajectories on the perfectly flat floor and the uneven floor. All trajectories exhibit a higher total thruster on-time in simulation than the optimal value for the perfectly flat floor and the uneven floor. This increase is approximately one order of magnitude for the perfectly flat floor, whereby the longer trajectories (Circular and "S"-Shaped) have a more significant increase than the straight line. Two main factors contribute to this deviation from the optimal values: the low control frequency of the thrusters and the high

Thruster	Straight Line			Circular			"S"-Shaped		
No.	Optimal	Flat	Uneven	Optimal	Flat	Uneven	Optimal	Flat	Uneven
0	0.499 s	1.80 s	3.31 s	0.674 s	5.92 s	18.0 s	0.0788 s	4.22 s	27.5 s
1	0.004 s	0.306 s	6.51 s	0.0 s	3.80 s	13.8 s	0.0895 s	4.00 s	1.4 s
2	0.0614 s	0.601 s	1.21 s	0.189 s	4.72 s	12.3 s	0.392 s	4.30 s	6.41 s
3	0.296 s	1.31 s	10.1 s	0.157 s	5.19 s	20.6 s	0.370 s	4.71 s	19.9 s
4	0.004 s	0.297 s	6.29 s	0.0 s	3.70 s	13.7 s	0.0895 s	3.91 s	1.41 s
5	0.499 s	1.78 s	3.09 s	0.674 s	5.90 s	17.8 s	0.0788 s	4.10 s	27.6 s
6	0.296 s	1.30 s	10.2 s	0.157 s	5.09 s	20.7 s	0.370 s	4.70 s	20.0 s
7	0.0614 s	0.499 s	1.20 s	0.189 s	4.70 s	12.3 s	0.392 s	4.30 s	6.41 s
Σ	1.72 s	7.90 s	41.9 s	2.04 s	39.0 s	129 s	1.86 s	34.24 s	110 s

Table 5.5: Optimal and simulated thruster on-times for different trajectories on a perfectly flat floor and uneven floor. For the simulated trajectories, the thruster on-time considers all thruster firings until reaching the final state.

penalization of errors in the pose. As shown in section 4.4, the deviation from the optimal value increases, the slower the control frequency is. Since the physical limitations of the thrusters (valve opening and closing times) do not allow for higher control frequencies, this imposes a hard limit on optimality. Further, the weights used in the TVLQR formulation to compute the optimal feedback gain matrix are penalizing errors in position with a weight in the order of 1e4. While improving performance in terms of reduced average error, every small error triggers a pulse, significantly increasing the thruster usage. The thruster usage increases dramatically, especially on the uneven floor, where minor errors occur more regularly. More specifically, performance and thruster actuation are inversely proportional to each other, and thus the user must make a trade-off by tuning the weight matrices.

Monte Carlo During the Monte Carlo test simulation, the controller successfully controls the system to the origin in all 100 tested initial conditions. As seen in Figure 5.15 the trajectories are mostly straight lines with some minor deviations caused by a combination of uneven ground, the measurement noise, and the delay introduced by the low thruster control frequency. One particular trajectory (purple, bottom right of the ground-track) overshoots the desired state along the trajectory at approximately 38 s. Since the controller only attempts to control the system to the desired state at some time, it returns to a previous location resulting in a loop. This loop also corresponds to the spikes in velocities (purple). All trajectories reach the desired region at the origin and slow down to the desired maximal velocity in less than 140 s. This supports the hypothesis that the controller can control the system at any region of the uneven flat-floor.

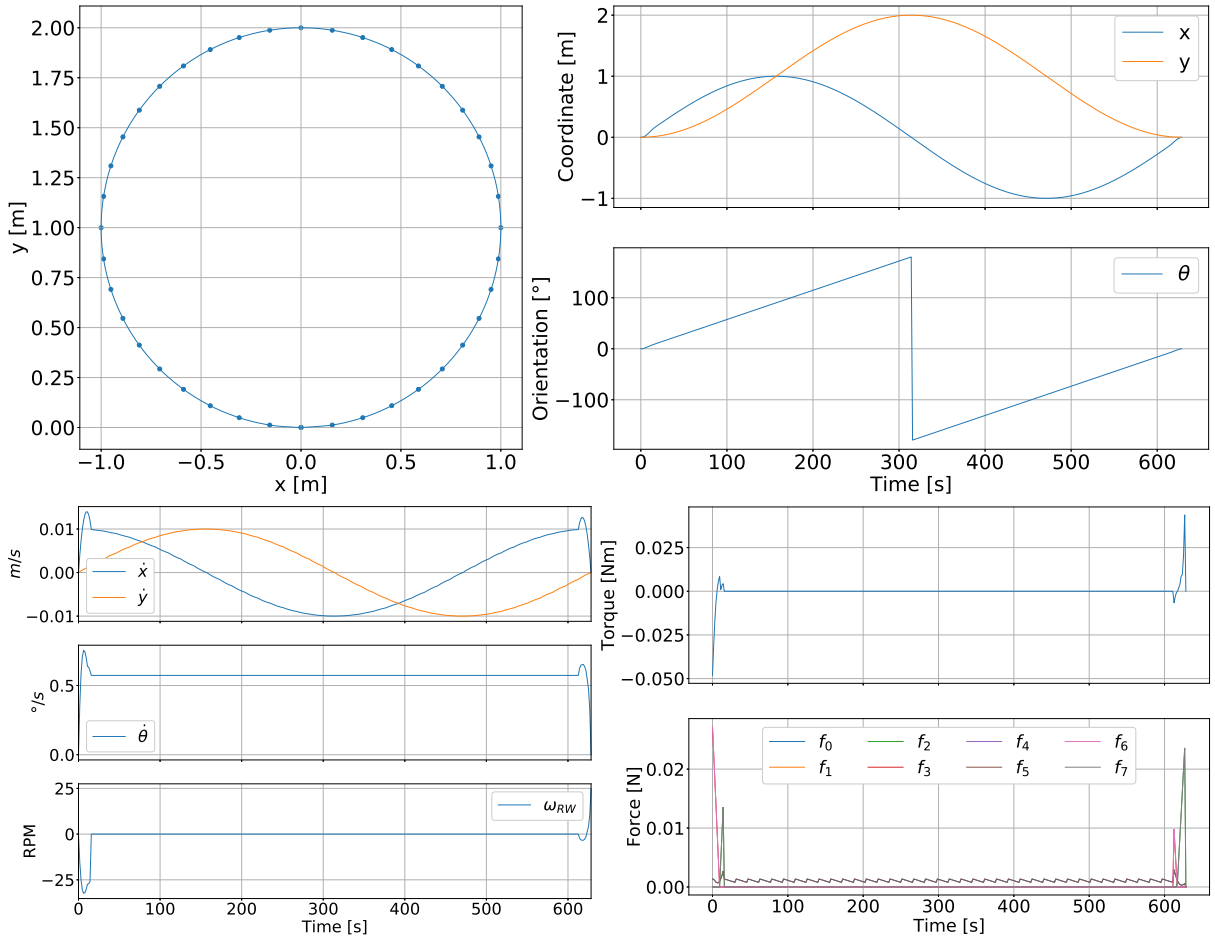


Figure 5.9: The optimal, pre-computed circular trajectory. Each dot in the ground-track (top left) represents one state in between which the trajectory planner computes an optimal trajectory.

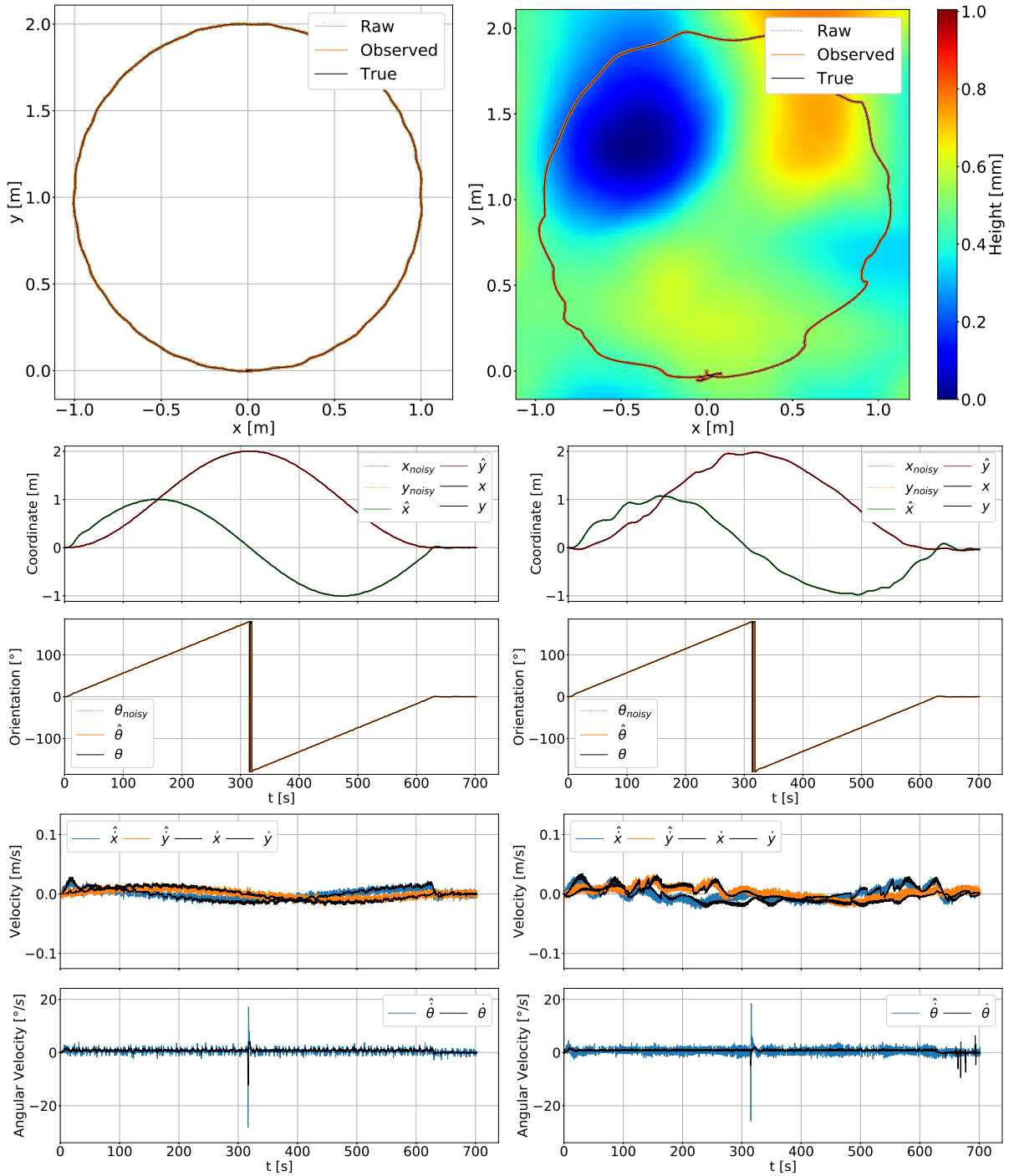


Figure 5.10: Ground-track, individual coordinates, and velocity of the system following a circular trajectory in simulation. Left: ideally even floor. Right: the floor is slightly uneven and induces disturbances. An animation of the trajectory is given at https://youtu.be/1A5xJVEAU9w?t=42.

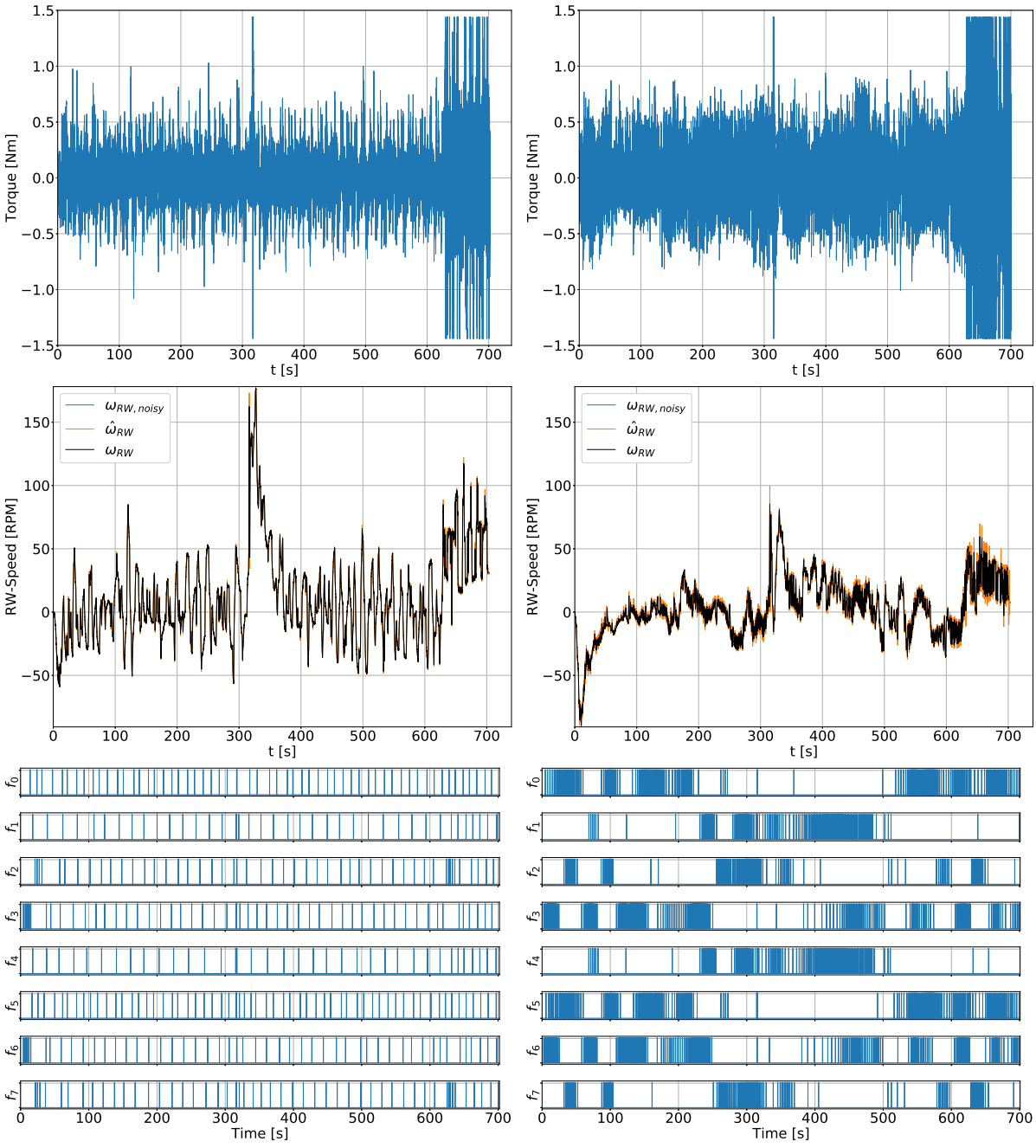


Figure 5.11: Actuation needed to follow a circular trajectory (cf. figure 5.9), while being subjected to noise in the state measurements. From top to bottom: commanded torque, RW velocity, and thruster firings. Left: The trajectory is followed on a perfectly flat floor. Right: the floor is slightly uneven and induces disturbances.

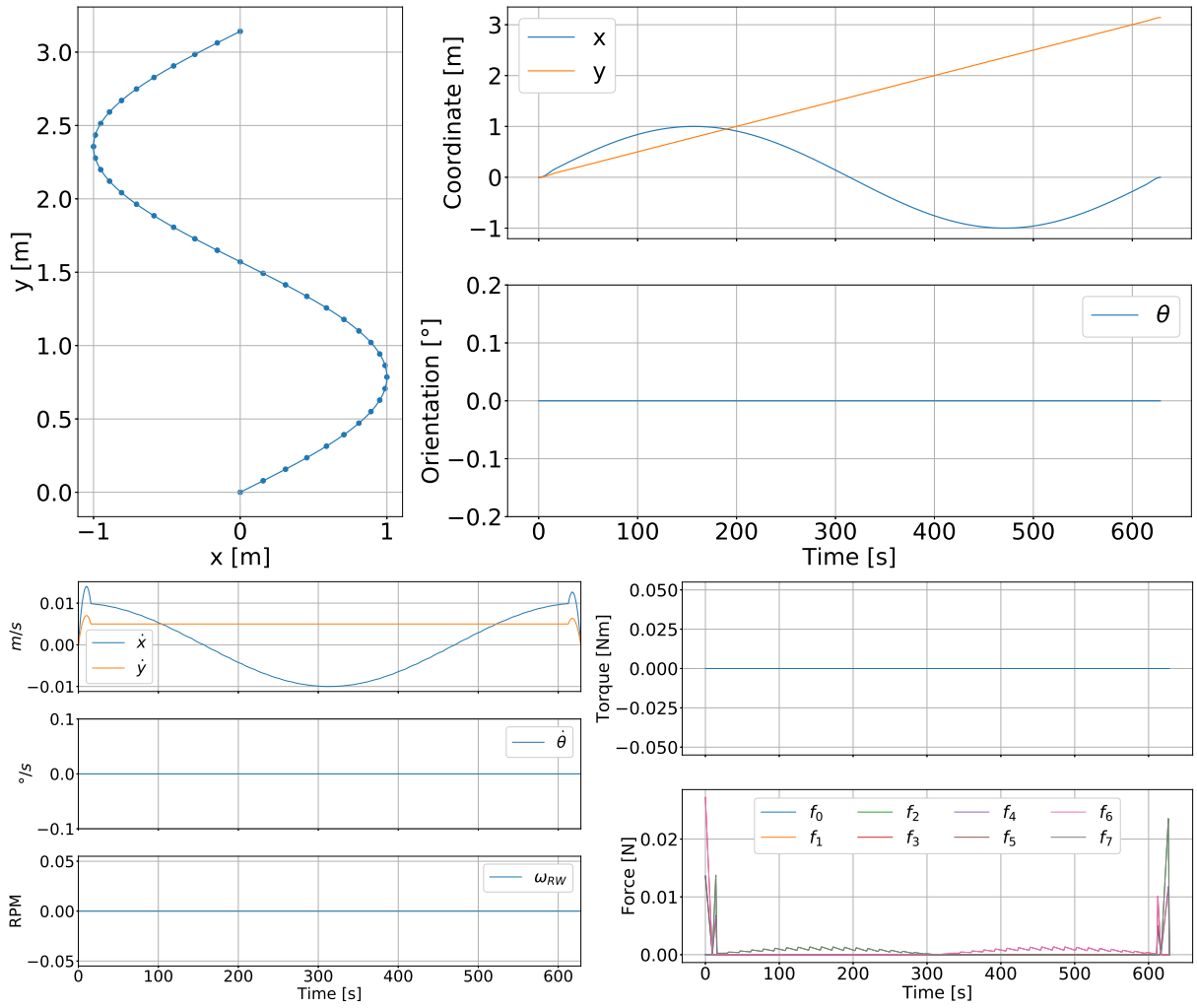


Figure 5.12: The optimal, pre-computed "s"-shaped trajectory. Each dot in the ground-track (top left) represents one state in between which the trajectory planner computes an optimal trajectory.

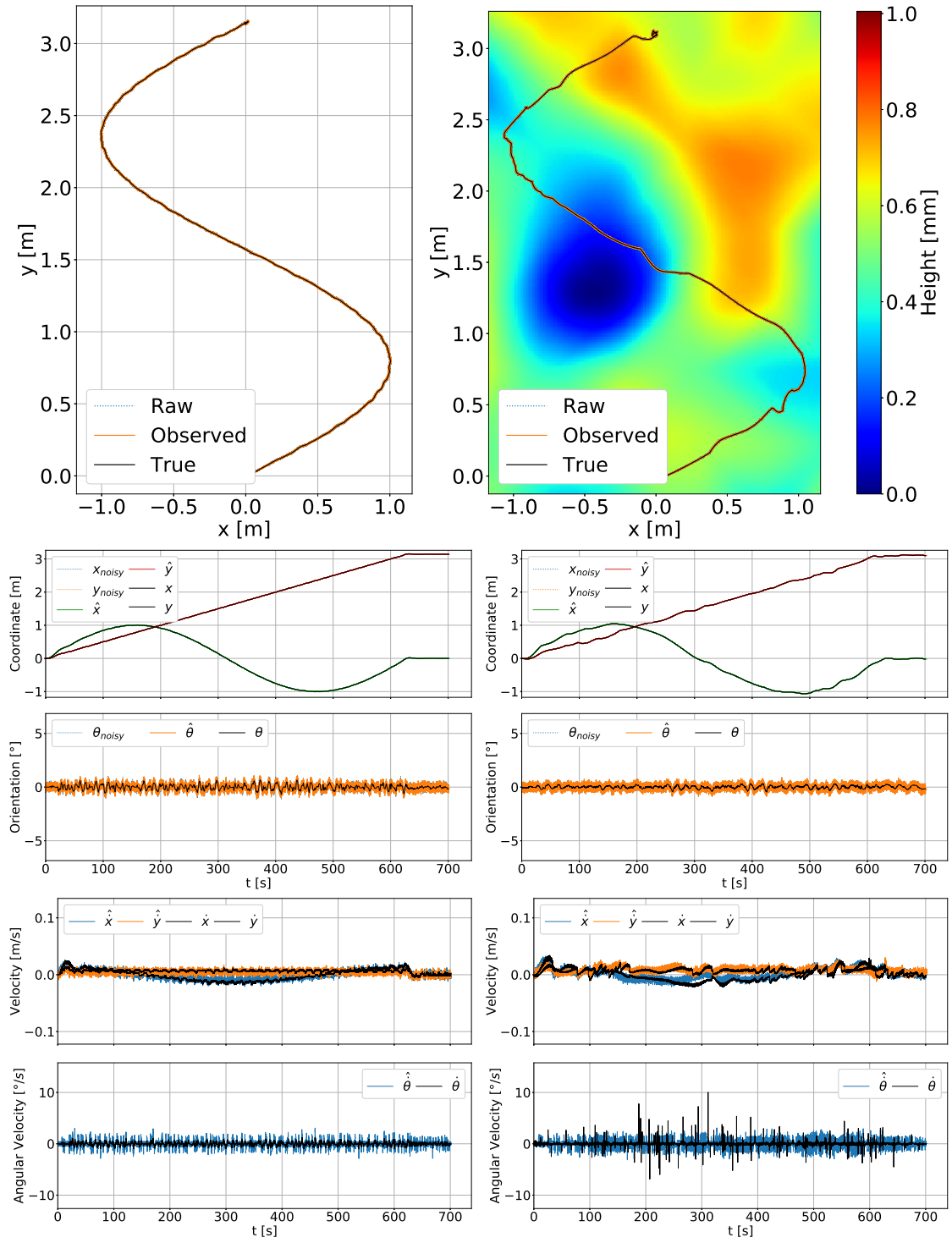


Figure 5.13: Ground-track, individual coordinates, and velocities of the system following an "s"-shape trajectory. Left: ideally even floor. Right: the floor is slightly uneven and induces disturbances. An animation of the trajectory is given at https://youtu.be/1A5xJVEAU9w?t=104.

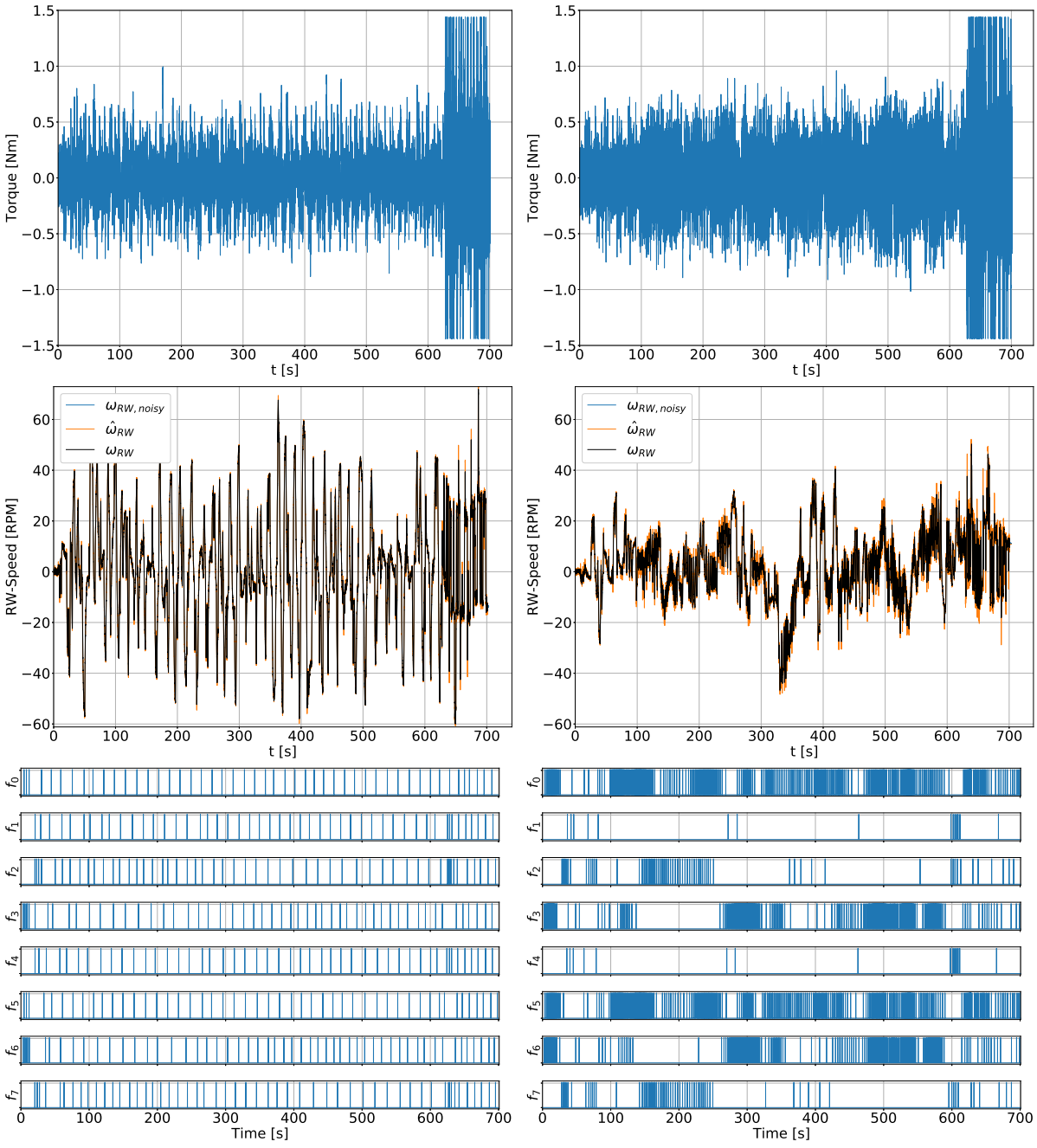


Figure 5.14: Actuation needed to follow an "s"-shaped trajectory (cf. figure 5.12), while being subjected to noise in the state measurements. From top to bottom: commanded torque, RW velocity, and thruster firings. Left: The trajectory is followed on a perfectly flat floor. Right: the floor is slightly uneven and induces disturbances.

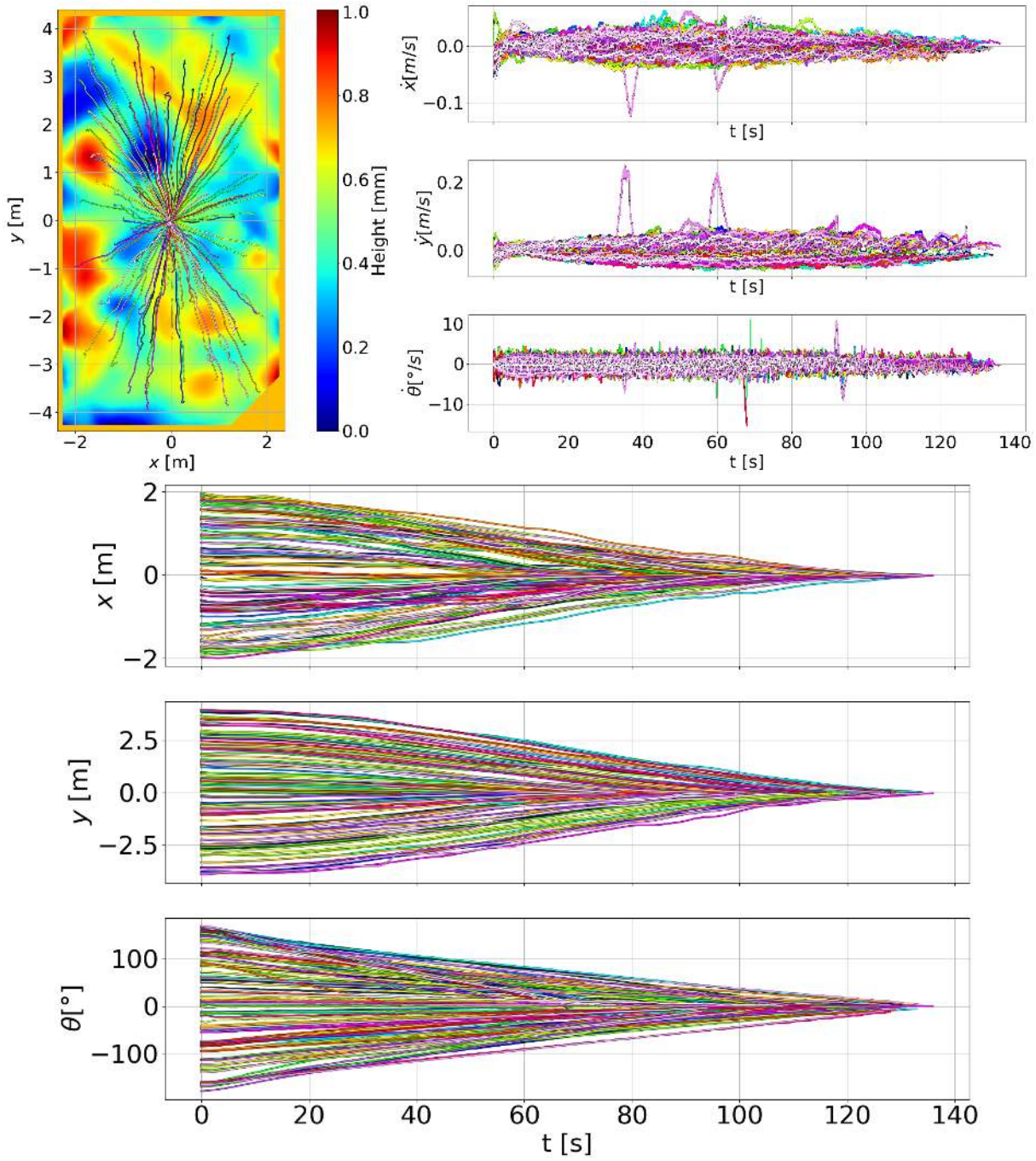


Figure 5.15: Monte Carlo Simulation of finding and following a trajectory to the origin. For $n = 100$ episodes the system is spawned at a pose (x, y, θ), drawn from uniformly distributed random distributions in the ranges $[-2, 2]$, $[-4, 4]$, and $[-\pi, \pi]$ respectively. The controller is then tasked to find a trajectory to the origin and command the system to follow it.

Chapter 6

Evaluation on the Real System

6.1 Experimental Setup

The tests on the physical system are performed at the Orbital Robotics and GNC Lab (ORGL) at ESTEC, ESA. Figure 1.1 shows the author at the test facility with the entire system. Before each testing day, dust and dirt are removed from the flat-floor to avoid disturbances and even damage to the floor, by particles getting caught between the air-bearings and the ground. Further to avoid issues with static friction, which the Reaction-Wheel (RW) experiences at low RPM, before each trajectory it is spun up to half its rated velocity such that it operates around some idling velocity. The world origin is defined somewhere on the flat-floor using an active marker for the Motion-Capture (MoCap) system as in [78]. From this state, the system is manually placed approximately at the origin of the local coordinate system and the trajectory following controller is started. The weight matrices for the observer and the controller are re-tuned for the physical system and can be found in appendix A.

6.2 Results

While the simulation of the ORGL provides a helpful development environment and allows testing in otherwise physically infeasible environments (such as the perfectly flat floor), its transferability to the physical world is limited. The simulation can provide upper limits on the performance, but unmodeled disturbances always lead to worse results in the physical world. This section evaluates this lesser performance on the real system, using the same metrics and trajectories as for the results in simulation.

6.2.1 Stabilization

First, the minimum requirement of stabilization is tested also on the physical system. The disturbance is added by manually pushing the system. The results are shown in Figures 6.1 and 6.2.

In all experiments, a general tendency to drift in the negative x-and y-direction is observed. From this, it is concluded that there is a slope pointing in this direction, which is further sup-

	e_x	e_y	$e_{\lvert x,y \rvert}$	e_θ
Straight Line	0.256 m	0.200 m	0.325 m	23.8°
Semi Circle	0.317 m	0.311 m	0.444 m	51.2°

Table 6.1: The average error of the physical system trying to follow a trajectory to that desired trajectory for the individual coordinates and the euclidean distance. The error is computed from the desired trajectory and the observed pose obtained from the Kalman Filter (KF).

ported by the heightmap (cf. Figure 5.3) which has a slightly elevated region (green/yellow) close to the origin with a lower region (blue) in negative x-and y-direction. During the stabilization process, the system initially stabilizes within approximately 15 cm and 15° of the origin. The subsequent disturbance moves the system about 35 cm in positive x and y direction, and within 30 s the system stabilizes in the initial region again. Figure 6.2 shows that very little thrusting is required to achieve this result, firing no thruster more than once every two seconds during the entire process. The RW, on the other hand, saturates rather quickly after the disturbance. As it attempts to absorb the entire angular momentum put into the system by the disturbance, it quickly reaches its maximum rotational velocity. Afterward, the thrusters are no longer supported by the RW in orientation control. Thus, the pointing accuracy degrades, as larger errors are required to trigger a pulse since thruster usage is penalized more than RW acceleration.

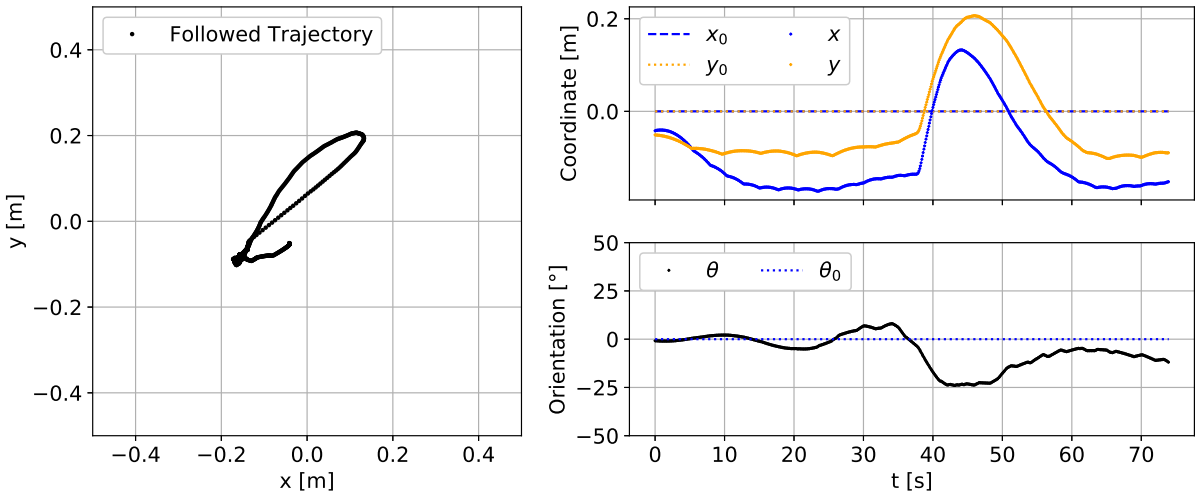


Figure 6.1: Ground-track and individual coordinates of the controller stabilizing at the origin. The controller attempts to bring the system to the origin. In the coordinate plots (right) the desired value is indicated as a dashed line. A video of the stabilization process is given at https://youtu.be/1A5xJVEAU9w?t=169.

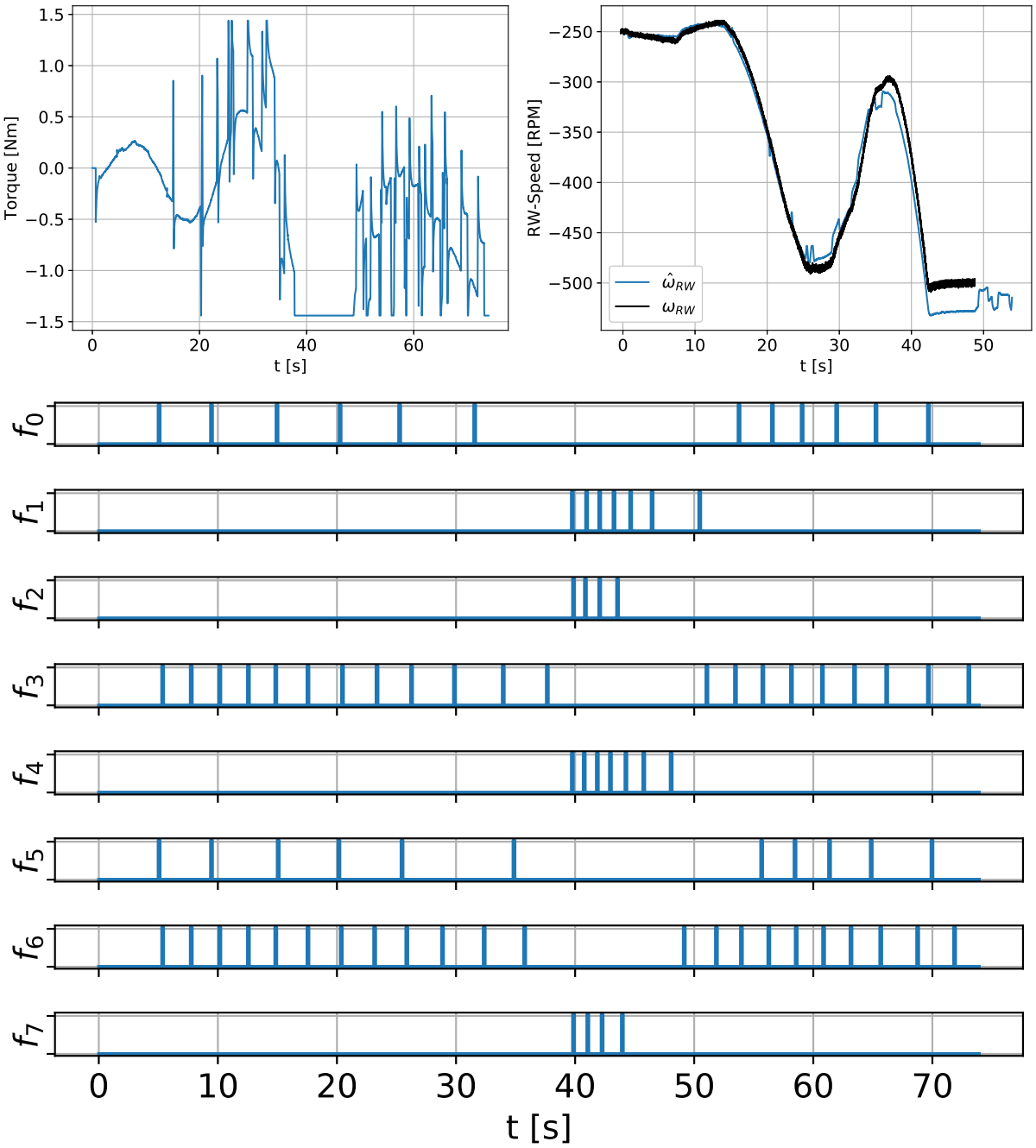


Figure 6.2: Actuation of stabilizing at the origin (cf. Figure 6.1). Top Left: the torque exerted by the motor onto the RW. Top Right: the resulting RW velocity (raw and observed). Bottom: the thruster activity for all eight thrusters.

6.2.2 Straight-Line Trajectory

Further, the Straight-Line Trajectory is tested on the physical system. The results are shown in Figures 6.3 and 6.4. As specified in Table 6.1, on the physical system, the controller achieves an average euclidean error that approximately doubles and an angular error that is one order of magnitude larger compared to the simulated trajectory. In particular, the error in the x-axis is significant. This error is due to the slope in that region of the flat-floor, with a gradient in the negative x-direction.

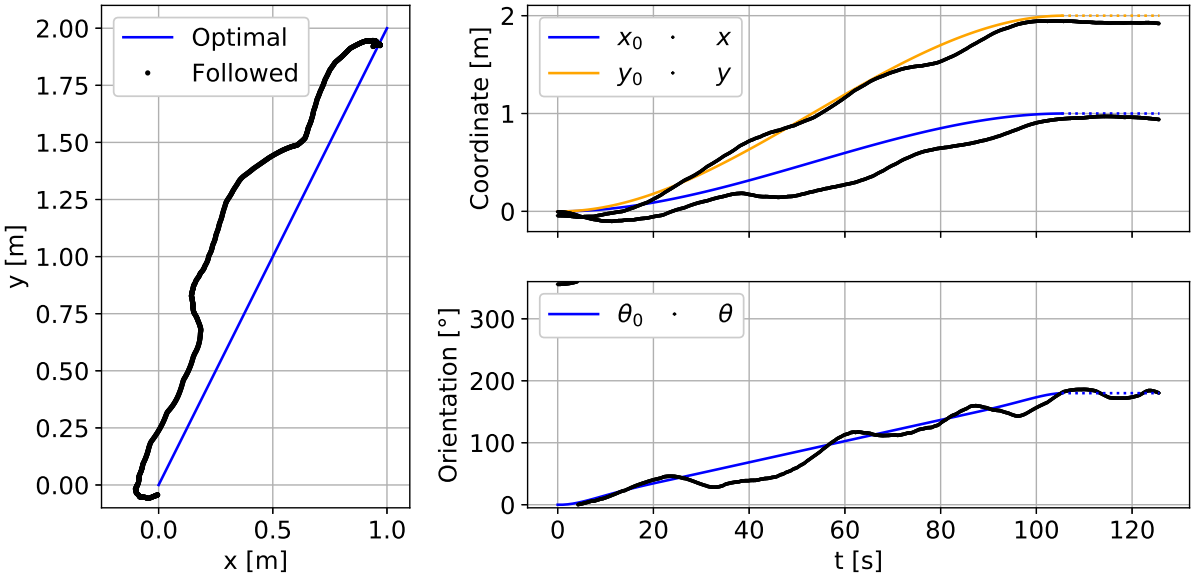


Figure 6.3: Ground-track and individual coordinates of the controller following a straight-line trajectory on the real system. After reaching the final pose of the trajectory plots (right) in the coordinate plots, the desired value is indicated as a dashed line. A video of the trajectory is given at https://youtu.be/1A5xJVEAU9w?t=239.

The heightmap (cf. Figure 5.3) shows one of the lowest local minima on the entire floor adjacent to the desired trajectory (deep blue circle at approximately $[0, 2]$) next to a higher region (orange at approximately $[1, 2]$), subjecting the system to steeper slopes and thus supporting this hypothesis. Further, the RW saturates at both ends of the allowable range throughout the entire trajectory. At each instance (around 40 s, 70 s, and 100 s) a larger deviation from the desired orientation is observed which implies that large desired changes in angular momentum of the entire system quickly leads to RW saturation. The controller then performs orientation control using only thrusters; thus, losing precision. However, at the end of the trajectory, the RW is not saturated, and therefore, the pointing accuracy at the final state is within 10°. Finally, the thrusters were firing appropriately, never exceeding the one fire per second threshold for an individual thruster. Moreover, when considering the thruster on-times, as depicted in Table 6.2, it shows an increase of less than one order of magnitude for the overall thruster on-time comparing to the simulated trajectory on the perfectly even floor. Some individual thrusters (0, 5) decrease on-time, whereas others increase by one or multiple orders of magnitude.

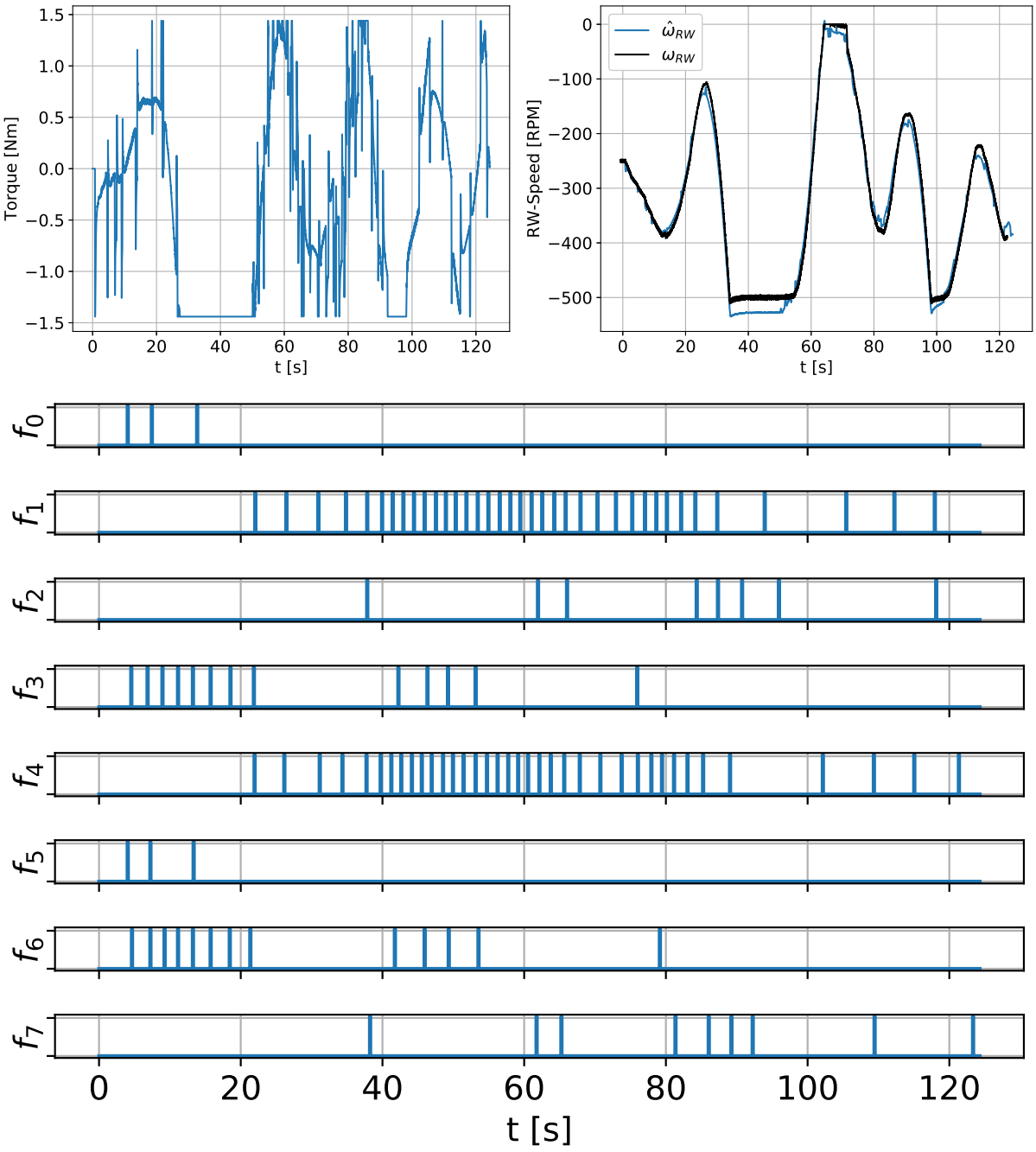


Figure 6.4: Actuation of following a straight-line trajectory (cf. Figure 6.3). Top Left: the torque exerted by the motor onto the RW. Top Right: the resulting RW velocity (raw and observed). Bottom: the thruster activity for all eight thrusters.

Thruster	Straight Line		Semi-Circle	
No.	Optimal	Real	Optimal	Real
0	0.499 s	0.300 s	0.338 s	6.00 s
1	0.004 s	3.40 s	0.0 s	1.80 s
2	0.0614 s	0.700 s	0.181 s	2.50 s
3	0.296 s	1.30 s	0.158 s	5.00 s
4	0.004 s	3.40 s	0.0 s	1.90 s
5	0.499 s	0.300 s	0.338 s	6.30 s
6	0.296 s	1.300 s	0.158 s	4.90 s
7	0.0614 s	0.700 s	0.181 s	2.50 s
Σ	1.72 s	11.4 s	1.35 s	30.90 s

Table 6.2: Optimal and real thruster on times for different trajectories. For the real trajectories, all thruster firings until the final state is reached are considered.

6.2.3 Semi-Circle

Due to propellant constraints, the system cannot execute a full-circle trajectory, as in the simulation, without risking the air-bearing touching and scratching the floor. For this reason, a Semi-Circle trajectory with all other properties remaining identical substitutes the full-circle on the physical system. The results are shown in Figures 6.5 and 6.6. While being less successful than the straight-line trajectory, the controller still maintains the general shape of the desired trajectory. The average euclidean distance is 0.444 m and the angular error is 51.2°. However, the performance throughout the first half of the semi-circle (until approximately 140 s) is better than the second half. The RWs saturation is the main culprit for this drop in performance, from which it does not recover for the rest of the trajectory. Afterward, the thrusters will only correct the orientation error once it reaches a certain threshold that triggers a thruster pulse, leading to large oscillations about the desired orientation. Further, these large orientation correction manoeuvrers disturb the position, which the thrusters must correct. In Figure 6.6 this is shown in the thruster actuation starting at approximately 250 s where the thruster is firing extensively. Lastly, the trajectory is also affected by the floor unevenness. Similar to one trajectory in the Monte-Carlo simulation in section 5.4, the system slides down a steep slope (approximately at the local blue minima at $[0, 2]$) and gets ahead of the trajectory. Since the trajectory follower does not recompute the planned trajectory and only attempts to move the system to a currently desired state, it returns to a previous location. It then follows the trajectory, resulting in a loop in the ground-track.

6.3 Discussion

The most significant factors that influence the overall system's performance are lack of precision in system identification and limited control authority. The former is outside the scope of this

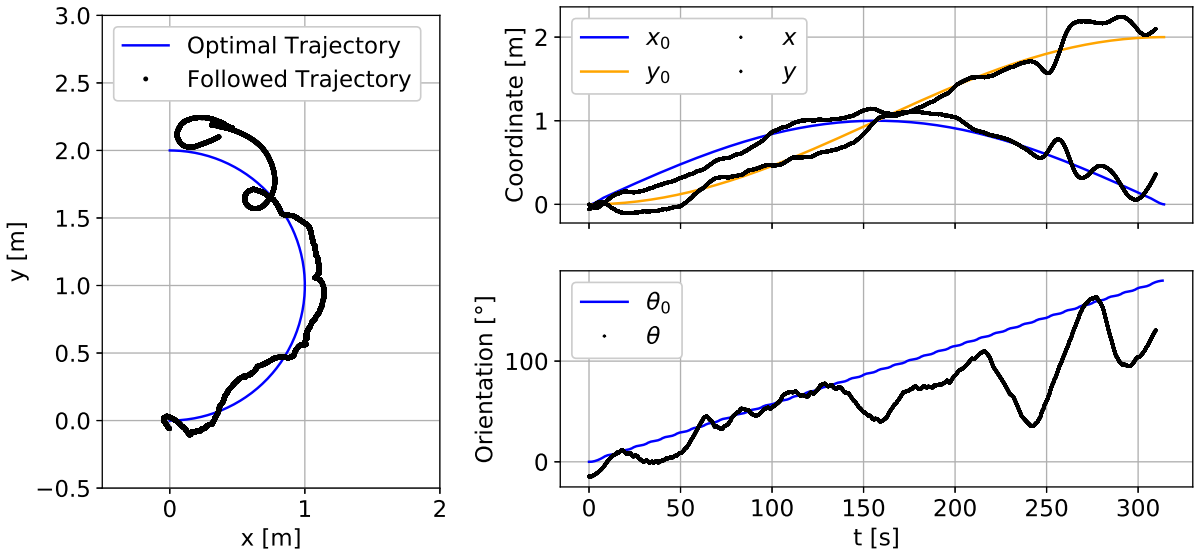


Figure 6.5: Ground-track and individual coordinates of the controller following a semi-circle trajectory on the real system. A video of the trajectory is given at https://youtu.be/1A5xJVEAU9w?t=311 .

work; thus, it relies on previous inertial measurements of the system. An offset here contributes to the oscillations around the desired orientation (cf. semicircle in Figure 6.5). This is especially the case, as the control architecture combines the controller with a KF, which famously does not provide any stability margin guarantees [79] for misidentification. Improving this model in terms of inertial parameters, thrust, and thrust vector for individual thrusters will significantly enhance the system's performance.

The latter is a consequence of the initial design of the floating platform. The system's large mass implies that the unevenness of the floor significantly influences the system's state, in contrast to its control authority. For example on a slope of 1 $\frac{\text{mm}}{\text{m}}$ the system experiences a constant disturbance force of approximately 2.2 N. In an ideal scenario with two thrusters precisely aligned with this disturbance force, they can provide approximately 21 N in the opposing direction. Thus to compensate for the disturbance force, the thrusters would have to have an on-time of at least 10.5%. One possibility is a firing of 100 ms each second only to compensate for the disturbance, which does not include any control action to follow some trajectory yet. This firing rate is larger than the firing rate observed for the straight-line trajectory. While physically feasible, this imposes a challenge on the controller and requires a large amount of propellant to execute. As already observed in the simulation, a tradeoff between performance and thruster usage, i.e., propellant consumption, is made by tuning the weight matrices used for computing the optimal feedback gain matrices. For the physical system, this tradeoff is made to favor thruster usage since the tank capacities are minimal, thus sacrificing some performance.

Additionally, the RW saturates very fast. Given the inertias of the entire system and the RW, it can compensate only for a slight change in angular velocity. The maximal change before

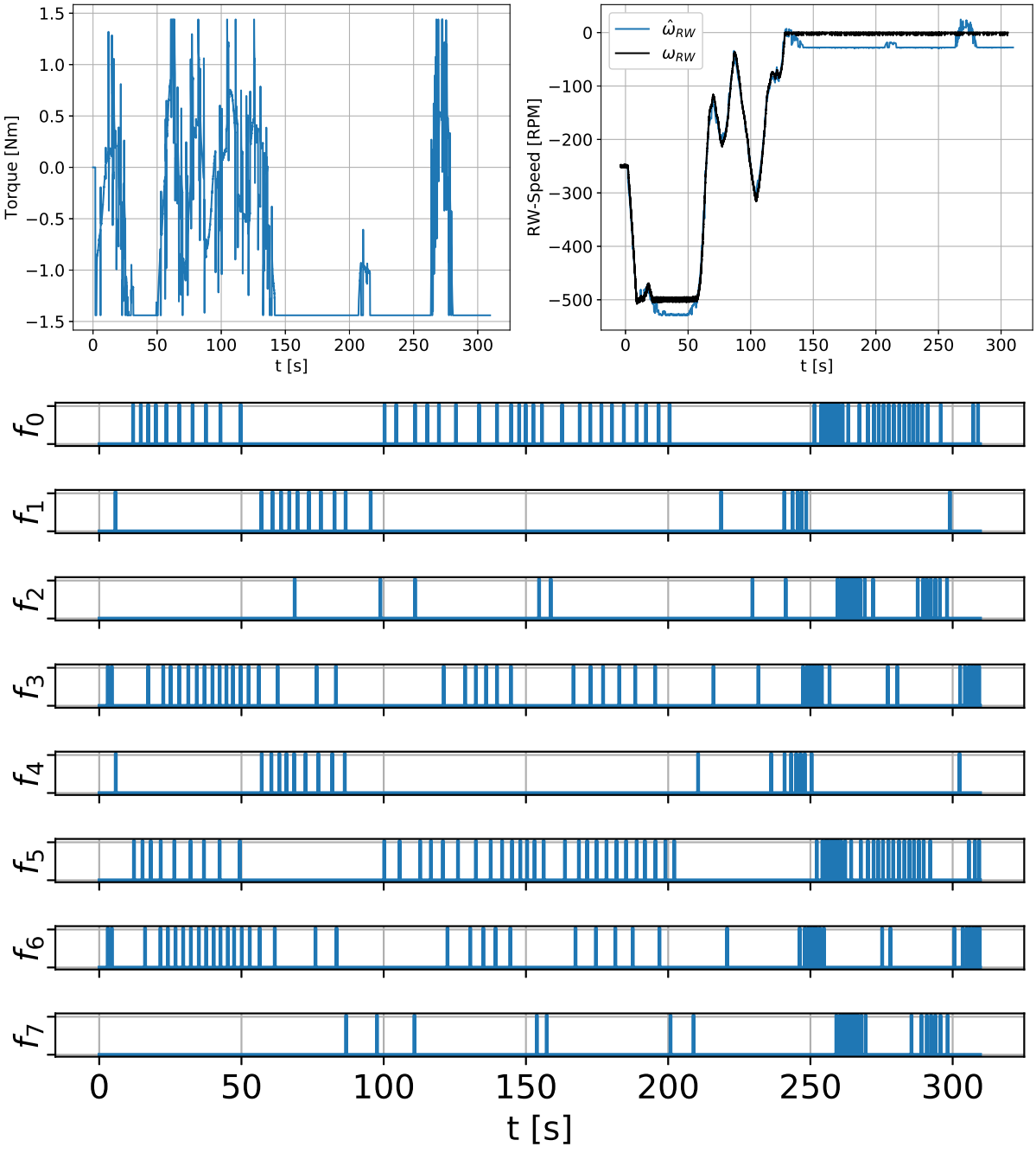


Figure 6.6: Actuation of following a semi-circle trajectory (cf. Figure 6.5). Top Left: the torque exerted by the motor onto the RW. Top Right: the resulting RW velocity (raw and observed). Bottom: the thruster activity for all eight thrusters.

saturation is:

$$\omega_{S,max} = \omega_{RW,max}\frac{I_{RW}}{I_S} = 5.81\,^\circ\,\mathrm{s}^{-1} \tag{6.1}$$

Therefore any trajectory that requires this angular velocity or any disturbance that imposes an equivalent torque on the system quickly saturates the RW and thus degrades the control authority of the system. One source of such disturbance is the uneven floor. Another is an unequal nominal force of individual thrusters. When firing two thrusters that point the same way (e.g., thruster 0 and 5), the exerted wrench should only contain a force pointing forward. However, if one assumes a 10% difference between the two thrusters a disturbance torque τ_d occurs:

$$\tau_d = (10\% \cdot f_{nom})r = 0.36\,\mathrm{N\,m} \tag{6.2}$$

From this we can find the firing time of the thrusters that is needed to impose enough change in angular velocity to saturate the RW:

$$\tau_d T = I_s \Delta\omega \Rightarrow T = \frac{I_s \omega_{S,max}}{\tau_d} = 3.4\,\mathrm{s}\,, \tag{6.3}$$

which means that after 3.4 s of thruster firing the RW will saturate. At a minimal thruster on time of 10% to compensate for the floor unevenness, this implies a test duration of 34 s. Therefore any trajectory, even just stabilizing the system at the origin, is likely to saturate the RW within less than 40 s only from disturbance torques imposed by the thrusters. The executed trajectories on the physical system show saturation of the RW within time limits in a similar range (42 s, 38 s, and 10 s). With the compensation of the thrusters, the controller can recover the reaction wheel to the non-saturated state, but its control authority remains very limited.

Lastly, in this control architecture, the trajectories are only pre-computed once, leading to undesirable behavior once the system strives too far from the desired trajectory. For example, since the trajectory follower only considers the state the system is currently desired to be at based on time, it will return to previous locations in case it moved too far ahead before. This behavior results in loops in the ground-track as observed in the semicircle in Figure 6.5. Using the closest point on the trajectory in terms of physical distance as the desired state instead of based on time avoids this behavior. However, this approach forfeits the possibility to control the system's state at a given point in time and thus also the overall trajectory execution time.

Alternatively, the trajectory follower could invoke the trajectory planner regularly to re-plan the trajectories to the desired destination under given constraints. This control approach, also known as Model Predictive Control (MPC), allows the system to better respond to previously unmodeled disturbances that lead to a system state further away from the initial trajectory than the Time-Varying Linear Quadratic Regulator (TVLQR) is capable of handling gracefully. However, solving the optimization problem online also poses additional challenges. The controller must guarantee that the solver converges to a solution within some set maximal time, which should be fast enough to respond to the system behavior. It needs to do that for trajectories of non-zero initial velocities, which generally require more computational time. Lastly, it needs to handle the cases in which the solver does not converge without becoming unstable.

Within this MPC framework also, alternative optimization schemes are imaginable. Given MPCs discrete nature, an optimization directly over thruster on and off times is feasible (if only for a short control horizon). Such an optimization scheme would make the modulator obsolete, thus discarding one more abstraction layer in the control architecture.

Chapter 7

Summary and Outlook

Three Degrees of Freedom (DoF) floating platforms are a good way of partially emulating microgravity environments as they are present in space applications. This work introduced a controller for one of the heaviest floating platforms in Europe located within the Orbital Robotics and GNC Lab (ORGL) at ESTEC, ESA.

First, this work develops a dynamic model of the overall system, actuated by eight solenoid-valve thrusters and one Reaction-Wheel (RW). The developed controller consists of two main components: the trajectory planner and the trajectory follower. The former pre-computes optimal trajectories that connect two arbitrary states while minimizing the force exerted by the thrusters. The trajectory follower computes the control actions to follow these and any other physically feasible trajectory using a Time-Varying Linear Quadratic Regulator (TVLQR) formulation. To abide by the binary nature of the thrusters, it uses a $\Sigma\Delta$- Modulator to modulate the continuous force command computed by the TVLQR formulation onto the thrusters. Finally, a combination of a classical Kalman Filter (KF) and a KF over the Lie-algebra $SO(2) \times \mathbb{R}$ estimates the system state; thus providing state feedback for the control architecture.

This work further develops a simulation of the overall system that takes measurement noise and the unevenness of the floor into account. When testing the control architecture in this simulation, the controller achieves at least 16 cm average euclidean and 5° angular error and in the best case up to 8 cm and 0.2° over the testes trajectories. The controller is also robust to arbitrary initial poses on the flat-floor. In a Monte-Carlo simulation in which the robot is spawned at arbitrary initial poses and tasked with finding and following optimal trajectories to the origin, the controller achieves a 100% success rate.

The controller is further evaluated on the physical system. There the trajectories are also followed successfully but experience a drop in performance. The average euclidean error approximately doubles, and the angular error increases by one order of magnitude. The increase in error is mostly attributed to a lack of precise system identification and control authority.

A thorough system identification will improve the system's performance in the future. A more precise system model with reduced error on the inertial parameters will significantly decrease the average errors as the state estimation and control match the physical system better. The thorough identification includes a more precise characterization of individual thrusters and their thrust vectors. This improved model is helpful in two domains: the trajectory planner would be

able to plan trajectories that are more representative of the physical system, and the controller would follow those better.

Redesigning some system components to increase control authority also helps to improve accuracy. The two most promising options are decreasing the overall weight and inertia of the system and increasing the reaction wheel inertia. The first one gives the current actuator more control authority and requires a redesign of the Air Cushion Robotic Platform (ACROBAT) – the base platform – which has the most significant contribution to weight and inertia while being the only passive platform. One way to satisfy the second option is to incorporate a different spinning mass in Reaction Control Autonomy Platform (RECAP) with higher inertia providing more margin before RW saturation.

In the future, it will be exciting to implement, test, and compare multiple controllers based on the software framework developed in this work. One class of controllers to compare incorporates knowledge about the unevenness of the ground into the system model. This prior knowledge allows the trajectory optimization to find paths of least resistance from one state to another, followed by a controller that enacts more force if it needs to overcome a significant slope. An alternate way of handling the unevenness of the ground would be robust control approaches that explicitly take model uncertainties into account. While generally having higher actuation costs, these approaches perform better if the exact dynamic model is unknown.

One promising approach is to solve the trajectory optimization problem online and control the system in a Model Predictive Control (MPC) fashion. That way, the controller would recompute optimal trajectories to the desired state after a disturbance occurred and thus would be more capable of rejecting those disturbances. An MPC approach in its discrete nature is also able to optimize over thruster on and off times directly (albeit only for smaller control horizons), making a modulator that translates the desired force obsolete. Further, this gives the chance to minimize the thruster on times providing a truly thrust-optimal solution to the trajectory tracking problem.

Appendix A

Weight Matrices

	Simulation	Real System
Q_{kf}	$\mathrm{Diag}\left(\begin{bmatrix} 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 1 \end{bmatrix}\right)$	$\mathrm{Diag}\left(\begin{bmatrix} 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 1 \end{bmatrix}\right)$
R_{kf}	$\mathrm{Diag}\left(\begin{bmatrix} 1e{-}3 & 1e{-}3 & 1e{-}3 & 5e3 & 5e3 & 5e3 & 1 \end{bmatrix}\right)$	$\mathrm{Diag}\left(\begin{bmatrix} 1e{-}3 & 1e{-}3 & 1e{-}3 & 5e3 & 5e3 & 5e3 & 1 \end{bmatrix}\right)$
Q_{TVLQR}	$\mathrm{Diag}\left(\begin{bmatrix} 1e4 & 1e4 & 1e4 & 100 & 100 & 100 & 1e{-}3 \end{bmatrix}\right)$	$\mathrm{Diag}\left(\begin{bmatrix} 5e3 & 5e3 & 5e2 & 5e2 & 5e2 & 2e3 & 1e{-}3 \end{bmatrix}\right)$
Q_{final}	$\mathrm{Diag}\left(\begin{bmatrix} 1e5 & 1e5 & 1e5 & 1e6 & 1e6 & 1e6 & 1e{-}7 \end{bmatrix}\right)$	$\mathrm{Diag}\left(\begin{bmatrix} 5e3 & 5e3 & 5e2 & 5e2 & 5e2 & 2e3 & 1e{-}3 \end{bmatrix}\right)$
R_{TVLQR}	$\mathrm{Diag}\left(\begin{bmatrix} 1e1 & 1e1 & 1e1 & 1e1 & 1e1 & 1e1 & 1e1 & 1e1 & 1e1 \end{bmatrix}\right)$	$\mathrm{Diag}\left(\begin{bmatrix} 1 & 200 & 200 & 200 & 200 & 200 & 200 & 200 & 200 \end{bmatrix}\right)$
R_{final}	$\mathrm{Diag}\left(\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}\right)$	$\mathrm{Diag}\left(\begin{bmatrix} 1 & 200 & 200 & 200 & 200 & 200 & 200 & 200 & 200 \end{bmatrix}\right)$

Table A.1: Weight Matrices used for the observer and controller for the simulation and the real-system

Appendix B

Pneumatic Plan

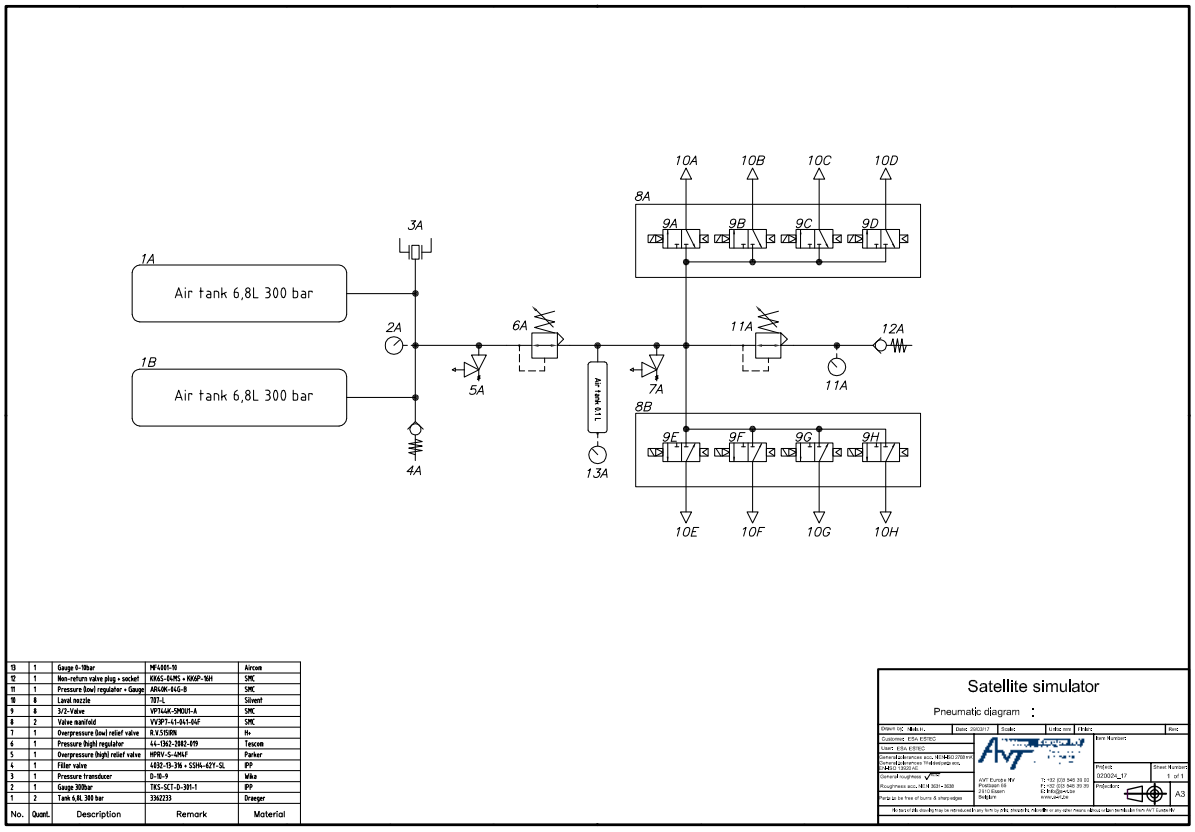


Figure B.1: The pneumatic plan of the thruster system. A high-pressure system consisting mostly of two air tanks that store up to 300 bar connected via a pressure regulator to the low-pressure system. The low-pressure system operates at pressures less than 8 bar.

Bibliography

[1] "Air-bearing floor image." https://er.jsc.nasa.gov/er5/. Accessed: 2022-02-12.

[2] "Formation control testbed." https://scienceandtechnology.jpl.nasa.gov/formation-control-testbed-fct. Accessed: 2022-01-08.

[3] A. Redah, T. Mikschl, and S. Montenegro, "Physically distributed control and swarm intelligence for space applications," 2018.

[4] A. Redah, H. R. Ramavaram, and S. Montenegro, "Robotic testing platform for autonomous rendezvous and docking of floating vehicles," 2019.

[5] "Zero-g lab uni luxembourg." https://ism.uni.lu/facility/zero-gravity-lab/. Accessed: 2022-01-31.

[6] E. Papadopoulos, I. Paraskevas, T. Flessa, K. Nanos, Y. Rekleitis, and I. Kontolatis, "The ntua space robot simulator: Design & results," 11 2008.

[7] M. Wilde, B. Kaplinger, T. Go, H. Gutierrez, and D. Kirk, "Orion: A simulation environment for spacecraft formation flight, capture, and orbital robotics," in *2016 IEEE Aerospace Conference*, pp. 1–14, 2016.

[8] T. Rybus and K. Seweryn, "Planar air-bearing microgravity simulators: Review of applications, existing solutions and design parameters," *Acta Astronautica*, vol. 120, pp. 239–259, 2016.

[9] S. Wehrmann and M. Schlotterer, "Coordinated orbit and attitude control of a satellite formation in a satellite simulator testbed," 05 2017.

[10] M. Zwick, I. Huertas, L. Gerdes, and G. Ortega, "Orgl – esa's test facility for approach and contact operations in orbital and planetary environments," 2018.

[11] H. Kolvenbach and K. Wormnes, "Recent developments on orbit, a 3-dof free floating contact dynamics testbed," 01 2016.

[12] *Optimal tuning of PWPF modulator for attitude control*. PhD thesis, 2005.

[13] A. I. Automation, "Ati gamma f/t-sensor," 2022.

[14] "Brushless dc motors." https://www.renesas.com/us/en/support/engineer-school/brushless-dc-motor-01-overview. Accessed: 2022-01-15.

[15] M. Kelly, "An introduction to trajectory optimization: How to do your own direct collocation," *SIAM Review*, vol. 59, pp. 849–904, 01 2017.

[16] R. Zappulla, "Experimental evaluation methodology for spacecraft proximity maneuvers in a dynamic environment," 2017.

[17] W. Commons, "Basic concept of kalman filteringf," 2014. File: Basic_concept_of_Kalman_filtering.svg.

[18] A. Ayala, F. Cruz, D. Campos, R. Rubio, B. Fernandes, and R. Dazeley, "A comparison of humanoid robot simulators: A quantitative approach," in *2020 Joint IEEE 10th International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob)*, pp. 1–6, 2020.

[19] R. Tedrake and the Drake Development Team, "Drake: Model-based design and verification for robotics," 2019.

[20] R. Crowther, "Space junk–protecting space for future generations," *Science*, vol. 296, no. 5571, pp. 1241–1242, 2002.

[21] C. J. Newman and M. Williamson, "Space sustainability: Reframing the debate," *Space Policy*, vol. 46, pp. 30–37, 2018.

[22] T. Schildknecht, "Optical surveys for space debris," *The Astronomy and Astrophysics Review*, vol. 14, no. 1, pp. 41–111, 2007.

[23] H. Klinkrad, *Space debris: models and risk analysis*. Springer Science & Business Media, 2006.

[24] S.-I. Nishida, S. Kawamoto, Y. Okawa, F. Terui, and S. Kitamura, "Space debris removal system using a small satellite," *Acta Astronautica*, vol. 65, no. 1-2, pp. 95–102, 2009.

[25] D. Mehrholz, L. Leushacke, W. Flury, R. Jehn, H. Klinkrad, and M. Landgraf, "Detecting, tracking and imaging space debris," *ESA Bulletin(0376-4265)*, no. 109, pp. 128–134, 2002.

[26] D. J. Kessler and B. G. Cour-Palais, "Collision frequency of artificial satellites: The creation of a debris belt," *Journal of Geophysical Research: Space Physics*, vol. 83, no. A6, pp. 2637–2646, 1978.

[27] G. Neuneck, *China's ASAT test — A warning shot or the beginning of an arms race in space?*, pp. 211–224. Vienna: Springer Vienna, 2008.

[28] T. Kelso, "Analysis of the iridium 33 cosmos 2251 collision," 2009.

[29] L. Crane, "Anti-satellite weapons," *New Scientist*, vol. 252, no. 3362, p. 15, 2021.

[30] B. Bastida Virgili, J. Dolado, H. Lewis, J. Radtke, H. Krag, B. Revelin, C. Cazaux, C. Colombo, R. Crowther, and M. Metz, "Risk to space sustainability from large constellations of satellites," *Acta Astronautica*, vol. 126, pp. 154–162, 2016. Space Flight Safety.

[31] J. Chatterjee, J. N. Pelton, and F. Allahdadi, *Active Orbital Debris RemovalActive orbital debris removaland the Sustainability of Space*, pp. 921–940. Cham: Springer International Publishing, 2015.

[32] S. Peters, H. Fiedler, W. Mai, and R. Förstner, "Research issues and challenges in autonomous active space debris removal," p. 9, 09 2013.

[33] C. P. Mark and S. Kamath, "Review of active space debris removal methods," *Space Policy*, vol. 47, pp. 194–206, 2019.

[34] "Clearspace-1 executive summary report." https://nebula.esa.int/sites/default/files/neb_study/2508/C4000128786ExS.pdf. Accessed: 2021-08-05.

[35] J. Schwartz, M. Peck, and C. Hall, "Historical review of air-bearing spacecraft simulators," *Journal of Guidance, Control, and Dynamics*, vol. 26, 05 2003.

[36] C. Menon, A. Aboudan, S. Cocuzza, A. Bulgarelli, and F. Angrilli, "Free-flying robot tested on parabolic flights: Kinematic control," *Journal of Guidance Control and Dynamics - J GUID CONTROL DYNAM*, vol. 28, pp. 623–630, 07 2005.

[37] E. Papadopoulos, F. Aghili, O. Ma, and R. Lampariello, "Robotic manipulation and capture in space: A survey," *Frontiers in Robotics and AI*, vol. 8, p. 228, 2021.

[38] K. Yoshida, "Ets-vii flight experiments for space robot dynamics and control," in *Experimental Robotics VII* (D. Rus and S. Singh, eds.), (Berlin, Heidelberg), pp. 209–218, Springer Berlin Heidelberg, 2001.

[39] "Air bearing floor." https://www.nasa.gov/centers/johnson/engineering/integrated_environments/air_bearing_floor/index.html. Accessed: 2021-08-18.

[40] J. D. Mitchell, S. P. Cryan, D. Strack, L. L. Brewster, M. J. Williamson, R. T. Howard, and A. S. Johnston, "Automated rendezvous and docking sensor testing at the flight robotics laboratory," in *2007 IEEE Aerospace Conference*, pp. 1–16, 2007.

[41] M. Regehr, A. Acikmese, A. Ahmed, M. Aung, K. Clark, P. MacNeal, J. Shields, G. Singh, R. Bailey, C. Bushnell, A. Hicke, B. Lytle, and R. Rasmussen, "The formation control testbed," in *2004 IEEE Aerospace Conference Proceedings (IEEE Cat. No.04TH8720)*, vol. 1, pp. 557–564 Vol.1, 2004.

[42] "Zero-g lab uni luxembourg." https://www.uni.lu/snt/research/spacer/infrastructures. Accessed: 2022-01-31.

[43] M. Schlotterer and S. Theil, "Testbed for on-orbit servicing and formation flying dynamics emulation," 08 2010.

[44] B. Wie and P. Barba, "Quaternion feedback for spacecraft large angle maneuvers. aiaa j. guid. control dyn. 8, 360-365," *Journal of Guidance Control and Dynamics - J GUID CONTROL DYNAM*, vol. 8, pp. 360–365, 05 1985.

[45] T. Lovelly and A. George, "Comparative analysis of present and future space-grade processors with device metrics," *Journal of Aerospace Information Systems*, vol. 14, pp. 1–14, 03 2017.

[46] E. Canuto, C. Novara, L. Massotti, D. Carlucci, and C. P. Montenegro, "Chapter 9 - orbit and attitude actuators," in *Spacecraft Dynamics and Control* (E. Canuto, C. Novara, L. Massotti, D. Carlucci, and C. P. Montenegro, eds.), Aerospace Engineering, pp. 463–520, Butterworth-Heinemann, 2018.

[47] E. Canuto, C. Novara, L. Massotti, D. Carlucci, and C. P. Montenegro, "Chapter 12 - attitude control: A case study," in *Spacecraft Dynamics and Control* (E. Canuto, C. Novara, L. Massotti, D. Carlucci, and C. P. Montenegro, eds.), Aerospace Engineering, pp. 607–658, Butterworth-Heinemann, 2018.

[48] M. Preisinger, "Advancing the attitude determination and control system for the cubesat move-ii," masterarbeit, Technische Universität München, 2019. RT-MA-2018/24.

[49] "Drager." https://www.draeger.com/nl_nl/Products/Compressed-Air-Breathing-Cylinders. Accessed: 2022-01-04.

[50] "Tescom 44-1300 series pressure regulator valve." https://www.emerson.com/en-us/catalog/tescom-44-1300. Accessed: 2022-01-26.

[51] "Smc." https://www.smcpneumatics.com/VP744K-5YOD1-04FA.html. Accessed: 2022-01-04.

[52] "Silvent." https://www.silvent.com/de/produkte/druckluftdusen-de/707-l-2/. Accessed: 2022-01-04.

[53] Y. Silik and U. Yaman, "Single axis attitude controller design using pulse width modulated thruster," pp. 1–6, 05 2019.

[54] E. Messerschmid and S. Fasoulas, *Raumfahrtsysteme: Eine Einführung mit Übungen und Lösungen*. Springer Berlin Heidelberg, 2017.

[55] G. Sutton, *Rocket Propulsion Elements: An Introduction to the Engineering of Rockets*. Wiley, 1992.

[56] "Nanotec brushless dc motors." https://de.nanotec.com/produkte/2245-db80m048030-enm05j. Accessed: 2022-01-15.

[57] P. Fortescue, J. Stark, and G. Swinerd, *Spacecraft Systems Engineering*. Wiley, 2003.

[58] "Newway." https://www.newwayairbearings.com/catalog/product/200mm-flat-round-air-bearings/. Accessed: 2022-01-04.

[59] J. Lunze, *Regelungstechnik 2: Mehrgrößensysteme, Digitale Regelung*. Springer-Lehrbuch, Springer Berlin Heidelberg, 2010.

[60] C. HARGRAVES and S. Paris, "Direct trajectory optimization using nonlinear programming and collocation," *AIAA J. Guidance*, vol. 10, pp. 338–342, 07 1987.

[61] P. Belotti, C. Kirches, S. Leyffer, J. Linderoth, J. Luedtke, and A. Mahajan, "Mixed-integer nonlinear optimization," *Acta Numerica*, vol. 22, p. 1–131, 2013.

[62] F. Borrelli, M. Baotić, A. Bemporad, and M. Morari, "Dynamic programming for constrained optimal control of discrete-time linear hybrid systems," *Automatica*, vol. 41, no. 10, pp. 1709–1721, 2005.

[63] M. Leomanni, A. Garulli, A. Giannitrapani, and F. Scortecci, "An mpc-based attitude control system for all-electric spacecraft with on/off actuators," p. 4853–4858, 12 2013.

[64] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, pp. 25–57, 2006.

[65] R. Tedrake, "Underactuated robotics: Algorithms for walking, running, swimming, flying, and manipulation (course notes for mit 6.832)." Downloaded on 21.12.2021 from http://underactuated.mit.edu/.

[66] R. Chartrand, "Numerical differentiation of noisy, nonsmooth data," *ISRN Appl. Math.*, vol. 2011, 05 2011.

[67] I. Knowles and R. J. Renka, "Methods for numerical differentiation of noisy data," 2014.

[68] F. van Breugel, B. Brunton, and J. Kutz, "Numerical differentiation of noisy data: A unifying multi-objective optimization framework," 09 2020.

[69] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME–Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.

[70] J. Hamilton, *Time Series Analysis*. Princeton University Press, 2020.

[71] I. Markovic, J. Cesic, and I. Petrovic, "On wrapping the kalman filter and estimating with the SO(2) group," *CoRR*, vol. abs/1708.05551, 2017.

[72] R. Kalman, "Contribution to the theory of optimal control," *Bol. Soc. Mat. Mexicana*, vol. 5, 02 2001.

[73] ISO, *ISO/IEC 14882:1998: Programming languages — C++*. Sept. 1998. Available in electronic form for online purchase at `http://webstore.ansi.org/` and `http://www.cssinfo.com/`.

[74] D. Thomas, W. Woodall, and E. Fernandez, "Next-generation ROS: Building on DDS," in *ROSCon Chicago 2014*, (Mountain View, CA), Open Robotics, sep 2014.

[75] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, vol. 3, pp. 2149–2154 vol.3, 2004.

[76] E. Drumwright, J. Hsu, N. Koenig, and D. Shell, "Extending open dynamics engine for robotics simulation," in *Simulation, Modeling, and Programming for Autonomous Robots* (N. Ando, S. Balakirsky, T. Hemker, M. Reggiani, and O. von Stryk, eds.), (Berlin, Heidelberg), pp. 38–50, Springer Berlin Heidelberg, 2010.

[77] M. Yan, Q. Sun, I. Frosio, S. Tyree, and J. Kautz, "How to close sim-real gap? transfer with segmentation!," *ArXiv*, vol. abs/2005.07695, 2020.

[78] "Vicon nexus product guide." https://documentation.vicon.com/nexus/v2.2/Nexus1_8Guide.pdf. Accessed: 2022-02-18.

[79] J. Doyle, "Guaranteed margins for lqg regulators," *IEEE Transactions on Automatic Control*, vol. 23, no. 4, pp. 756–757, 1978.

Proclamation

Hereby I confirm that I wrote this thesis independently and that I have not made use of any other resources or means than those indicated.

Würzburg, February 2022