Degree Project in Electrical Engineering, specializing in Systems, Control and Robotics

Second cycle, 30 credits

Failsafe Control for Space Robotic Systems

Model Predictive Control under Actuator Failures

RAPHAEL J. M. STÖCKNER

Failsafe Control for Space Robotic Systems

Model Predictive Control under Actuator Failures

RAPHAEL J. M. STÖCKNER

Date: October 21, 2024

Supervisors: Pedro Roque, Maria Charitidou

Examiner: Dimos Dimarogonas

School of Electrical Engineering and Computer Science

Swedish title: Fel-tolerant Reglering för Rymdrobotsystem

Swedish subtitle: Modellprediktiv Reglering vid Fel på Ställdon

Abstract

Future space missions face increasing safety challenges as Earth's orbit becomes fuller and fuller: With the growing number of satellites, the risk of collisions rises, potentially requiring emergency de-orbiting maneuvers when actuators fail in order to prevent crashes and further damage. Meanwhile, space missions are becoming safer and more cost-effective through the deployment of small autonomous space robots, such as NASA's Astrobee. These systems perform routine tasks inside space stations that would otherwise require human astronauts. In case of actuator failures, however, they must also maintain control to protect both equipment and the space station itself. This thesis thus explores control methods for thrust-propelled space robots under actuator failures, using the example of the KTH Freeflyers.

The analysis reveals that while single actuator failures can be addressed by adapting existing control schemes, multiple failures introduce nonholonomic constraints and prevent the simple compensation of an actuator. To address this, micro-orbiting is proposed as a strategy that allows to ensure stability for up to three simultaneous actuator failures. The idea is to bring the system on a small scale orbit in which the actuator failures and the orbit dynamics cancel out. A new model is derived, eliminating the nonholonomic constraints by controlling the orbit center.

Three controllers are designed: a feedback-linearization-based controller for setpoint-stabilization and two Model Predictive Control (MPC) based controllers for trajectory-tracking. One of the MPC controllers uses a novel approach of designing the terminal ingredients that is based on explicit MPC (eMPC) and shows to increase the terminal set drastically.

Simulation results indicate that this second MPC controller is preferred if hardware requirements are met, while the feedback-linearizing controller could serve as an alternative with low-computational needs if necessary.

Keywords

Model Predictive Control, Space robotics systems, Failsafe control, Actuator failure, Nonholonomic constraints, Astrobee

Sammanfattning

Framtida rymduppdrag står inför allt större säkerhetsutmaningar i takt med att jordens omloppsbana blir allt fullare: Med det växande antalet satelliter ökar risken för kollisioner, vilket kan kräva nödmanövrer för att ta ner satelliten ur omloppsbanan när ställdonen inte räcker till för att förhindra krascher och ytterligare skador. Samtidigt utvecklas rymduppdragen för att bli säkrare och mer kostnadseffektiva genom att små autonoma rymdrobotar, som NASA:s Astrobee, används. Dessa system utför rutinuppgifter inuti rymdstationer som annars skulle kräva mänskliga astronauter. I händelse av fel på ställdon måste de dock också behålla kontrollen för att skydda både utrustning och själva rymdstationen. I den här avhandlingen undersöks därför kontrollmetoder för tryckdrivna rymdrobotar vid fel på ställdonen, med KTH Freeflyers som exempel.

Analysen visar att medan fel på ett enstaka ställdon kan hanteras genom att anpassa befintliga styrsystem, medför multipla fel icke-holonomiska begränsningar och förhindrar enkel kompensation av ett ställdon. För att hantera detta föreslås micro-orbiting som en strategi som gör det möjligt att säkerställa stabilitet för upp till tre samtidiga ställdonsfel. Tanken är att föra systemet till en småskalig omloppsbana där fel på ställdonen och omloppsbanans dynamik neutraliseras. En ny modell härleds, som eliminerar de icke-holonomiska begränsningarna genom att styra omloppsbanans centrum.

Tre regulatorer designas: en feedback-lineariseringsbaserad regulator för setpoint-stabilisering och två MPC-baserade regulatorer (Model Predictive Control) för banföljning. En av MPC-styrenheterna använder ett nytt tillvägagångssätt för att utforma terminal ingredients som baseras på explicit MPC (eMPC) och visar sig förstora terminal sets drastiskt.

Simuleringsresultaten visar att denna andra MPC-styrenhet är att föredra om hårdvarukraven uppfylls, medan den feedback-lineariserande styrenheten kan fungera som ett alternativ med lägre beräkningskrav om det behövs.

Nyckelord

Modellprediktiv reglering, rymdrobotsystem, felsäker reglering, aktuatorfel, icke-holonomiska begränsningar, Astrobee

Acknowledgments

First of all, I would like to thank my professor Dimos Dimarogonas for overseeing this thesis and providing the opportunity to work on this project.

I am especially thankful to my supervisors Pedro Roque and Maria Charitidou. They gave me feedback and suggestions that helped me succeed with this project.

Finally, I want to thank my family and my friends for their support, understanding and encouragement during the last months. Especially I want to thank my girlfriend Johanna who proofread this thesis.

Parts of this work were stylistically enhanced, grammatically corrected or formulated using Claude 3.5 Sonnet. The content, proofs and scientific results all remain the work of the author and cited sources.

Stockholm, November 2024
Raphael J. M. Stöckner

Contents

List of Figures

List of Tables

Abbreviations and Symbols

Abbreviations

Abbreviation	Meaning
eMPC	explicit Model-predictive control
FTP	Failsafe trajectory planning
KKT	Karush-Kuhn-Tucker (conditions for optimality)
LE	Lyapunov equation
MPC	Model-predictive control
NASA	(American) National Aeronautics and Space Administration
NMPC	Nonlinear Model-predictive control
PWM	Pulse Width Modulation
SRL	Space Robotics Lab at KTH

Latin letters

Symbol	Meaning
c_i	Position (c_1, c_2) and velocity (c_3, c_4) of the micro-orbit center in x and y direction
$\boldsymbol{D}$	Map from the physical thrusters to the generalized input forces
d	Lever of each thruster w.r.t. the center of mass
e_i	Errors of the orbit center. $e_1 - e_4$: Equivalent to c_i, e_5: Error in ω.
$\boldsymbol{F}_{i,j}$	Vector of the physical thruster forces, $\boldsymbol{F}_{i,j} \in \mathbb{R}^8$.
$F_{i,j}$	Input forces of the thrusters. $i \in 1, \ldots, 4$ and $j \in 1, 2$
$\boldsymbol{F}$	Vector of resulting/generalized input forces $\boldsymbol{F} = \begin{pmatrix} F_x & F_y & T \end{pmatrix}^T \in \mathbb{R}^3$
$\boldsymbol{F}_{\text{comp}}$	Compensation force to set $\boldsymbol{F}_{\text{virt}}$
$\boldsymbol{F}_{\text{fault}}$	Absolute uncontrollable/faulty input force
$F_{\max}$	Maximal thrust force one thruster can emit
$\boldsymbol{F}_{\text{virt}}$	Virtual uncontrollable force
F_x, F_y	Resulting input forces in x and y direction
J	Inertia
l	MPC stage cost

l_N	MPC terminal cost
$\boldsymbol{M}$	Input transformation used in the model of the center dynamics
m	Mass
n_x, n_u	Number of system states and inputs
Q_e	MPC error/state cost
Q_u	MPC input cost
$\boldsymbol{R}$	2×2 rotation matrix
r	Radius of the orbit
t	Time
T	Resulting input torque
x_i, y_i	States in x/y-direction. $(\cdot)_1$: position, $(\cdot)_2$: velocity

Greek letters

Symbol	Meaning
α	Orientation of the freeflyer w.r.t. the x-axis
δ	Sampling time
Φ	Discretized system dynamics of the freeflyers
χ	Trajectory
ω	Angular velocity of the freeflyer
ω_0	(Constant) desired angular velocity for micro-orbiting

Calligraphic and other symbols

Symbol	Meaning
$\mathfrak{R}$	3×3 rotation matrix
$\mathcal{U}$	Set of inputs $u \in \mathbb{R}^3$ for a particular fault case with partial compensation/ virtual fault.
$\mathcal{U}_{i,j}$	Set of values one input $F_{i,j} \in \mathbb{R}$ can attain.
$\mathcal{U}_{\text{phys}}$	Set of thruster inputs $\boldsymbol{F}_{i,j} \in \mathbb{R}^8$.
$\mathcal{U}_{\text{rem}}$	Set of remaining inputs ($\in \mathbb{R}^3$) under actuator failurs without partial compensation/virtual fault.
$\mathcal{U}_{\text{res}}$	Set of resulting possible inputs ($\in \mathbb{R}^3$) if both failing and working thrusters are taken into account.
$\mathcal{X}$	Set of allowed/safe states x
$\mathcal{X}_{\text{term}}$	MPC terminal cost

Indices, exponents and operator names

Symbol	Meaning
$(\cdot)^T$	Transposed vector or matrix
$\hat{(\cdot)}$	Variable of explicit MPC

Chapter 1

Introduction

When designing a technical system, the failure-free state is usually considered first - after all, failures are not wanted and should ideally be prevented by robust design. However, full or partial system failures happen from time to time and is important to know how to deal with them. This matters particularly when the systems are deployed in environments where a fault can cause harm to humans. A lot of effort has therefore been put into the development of failsafe automotive systems or multicopters.

Despite these advancements in terrestrial applications, for space, there are surprisingly few results despite the space industry being more and more active in recent years. This gap is particularly unexpected given the inherent risks of space exploration.

This thesis contributes at closing this gap by proposing methods to control autonomous space robots even under actuator failures. It also proposes a novel way of calculating terminal sets based on explicit MPC. This method is in the present work mostly explained with the actual application in mind, while a full formal explanation of the method is still pending. As one of the main achievements of the thesis, it can nevertheless also be useful for other systems apart from space robots.

In this chapter, the background and the target system of this work are presented in Sections 1.1 and 1.2, followed by an overview of failsafe control in Sections 1.3 and 1.4. Afterwards, the contributions of the work are posed in Section 1.5 and finally the mathematical notation is stated in Section 1.6.

1.1 Motivation and Background

The travel to space has been a risky journey since the first humans were launched into the orbit, and it still is today. In the recent decade, a renewed interest into space has been found, leading to projects such as the Artemis program, lead by the American National Aeronautics and Space Administration (NASA)[1]. The ambition in this project is to build a stationary base on the Moon and a space station in the Moon's orbit, where the latter is planned to eventually serve as a gateway to Mars.

To support these ambitious goals while minimizing risks, autonomous robots have a huge potential for deployment. First of all, there is no human risk and additionally, they are even cheaper than sustaining a human at the station. A use case could, for example, be the maintenance of space stations.

Beyond lunar exploration, challenges also exist closer to home. In our own orbit around the Earth, we can see an ever-increasing number of satellites and space stations. The risk of collisions is hence rising with possibly disastrous consequences (ref. the Kessler syndrome, [2]). Besides of the development of techniques to clean space debris [3, 4], there is a need for precautionary methods to avoid crashes.

A study analyzed 129 spacecrafts that suffered from on-orbit failures between 1980 and 2005 [5]. It concluded that around 30% of the attitude and orbit control subsystems failures stem from actuator failures which makes it an important topic to be studied. Fewer redundancies in the new type of autonomous space robots due to space and cost constraints make the matter even more important.

Thus, to enhance safety in space for future missions, it is necessary to develop methods that enable the control of robots in zero- or microgravity conditions, even in the presence of actuator failures. This thesis focuses on advancing the theoretical foundations and practical applications of robust control strategies for space robotics, contributing to the ongoing efforts to make space operations safer and more reliable.

1.2 Target Platform

The arguments for the automation of space missions have of course also been thought of by the responsibles at the NASA. This has lead to the Astrobee program [6] whose goal is to automate routine tasks using small mobile robots inside of the International Space Station (ISS). These robots, called Astrobees like the program itself, have first been launched inside the ISS in 2019 and since then used by scientists and engineers to perform experiments in microgravity.

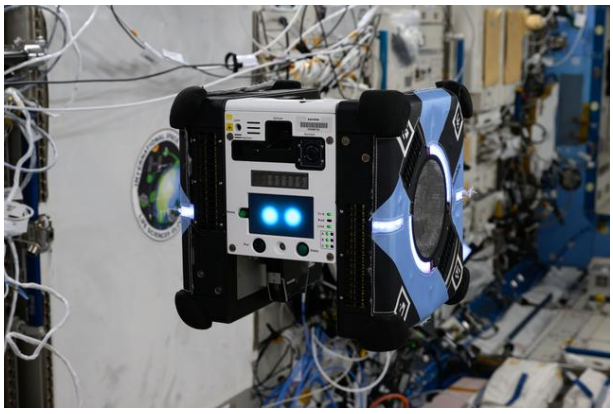


Figure 1.1: One of three NASA Astrobees at the ISS. Link to image: https://images.nasa.gov/details/iss071e046284

As getting access to this research platform is costly and difficult, KTH has built a simulator that mimics the properties of the Astrobee [7]. The Space Robotics Lab (SRL), opened

this summer and one of the biggest of its kind in Europe, is part of the DISCOWER (Distributed Control in Weightless Environments) project. Currently, the lab houses three robots called freeflyers that move frictionlessly in two dimensions, thus imitating the three-dimensional Astrobees. The frictionless motion is achieved by using a very smooth and flat surface, allowing the robots to hover through air bearings on a very thin air film. Propulsion is accomplished, as with the Astrobee, through air thrust. The robots have been designed in dimensions, mass and maneuvering capabilities to be comparable to the Astrobee.

Even though the eventual target system is supposed to be the Astrobee, this work uses the freeflyers at SRL. This choice allows for the development and theoretical validation of control algorithms in a controlled environment that closely mimics key aspects of the Astrobee's operational conditions.

1.3 Failsafe and Robust Control

In the following, an overview of failsafe and robust control is given. The differences between both are highlighted and a literature review of the applications of failsafe control is made.

Dealing with uncertainty is an inherent, one could even say *the* inherent problem in control theory. From the very start, the goal was to control systems under modelling errors, noise or perturbing influence from the surroundings. With time, it was noticed that state-space models and control are not robust enough for all applications, which lead to the development of H∞ control in the 80s and 90s [8]. When the development of Model Predictive Controllers (MPC) started, soon methods for robust MPC were investigated such as tube-based MPC [9].

In recent years, the interest in controlling uncertain systems was even more extended: While state-space and MPC controllers assume that the model and environment are very similar, H∞ control and robust MPC allow for higher uncertainty. Since the start of this century, researchers investigated more and more not only robust, but even fail-safe control.

It is important to make a clear distinction between robust and failsafe control: Robust control considers systems that are subject to uncertainty, for example because of large noise. The system itself is however, explicitly or implicitly, assumed to be able to handle the control problem and the model is assumed to be sufficiently exact to design a suitable controller.

In failsafe control, in contrast, this is not necessarily given any more: If certain actuators fail, for example, additional constraints can appear that forbid certain motions. Alternatively, assumptions that were made for the control of the functional system may become invalid which can lead to the need of a reformulation of the model or the control objective. In some cases, a loss of controllability is possible. Thus, failsafe control deals with a much larger room for uncertainty and extends the efforts to address this uncertainty even further.

1.4 A Review of Failsafe Control

Failsafe control is mainly applied in high-risk applications. A lot of attention was given to automotive and aerial systems where human lifes can be at risk. The efforts can be split up into two categories: On the one hand, there is the branch of failsafe control that stems from underactuated and nonlinear systems. This kind of theory was mainly developed at the end of the last century and, in the context of failsafe control, applied to space systems. On the other hand, there is the modern failsafe control that builds on modern control theory. It uses optimization-based controllers, e.g. MPC in different variations, and the calculation of occupancy, resp. safe and unsafe sets.

A look at existing research shows that the potential of MPC to deal with failures has been noticed early: Due to the flexibility of the framework, "simply" the internal model needs to be updated [10]. The problem is rather to ensure stability under the new constraints which is seldomly done in a theoretically thorough way, as shown in the following.

The following sections give an overview of failsafe control by topic. The first results on failsafe control using MPC were achieved for aerospace systems in the early 2000s. The existing works mainly show proofs-of-concept without formal proofs. Later in the 2010s, the interest in autonomous driving rose. This led to the development of control strategies that could take the inherent insecurity into account which stems from the interaction with other traffic participants. Failsafety was achieved in the sense that the controllers could autonomously react to unexpected movements of others. The interest in failsafe control for UAVs and multicopters has been researched since the 2010s and is still of high interest today. The last field of research presented is the failsafe control of spacecrafts which peaked in the 90s and early 2000s and received some renewed interest in recent years.

1.4.1 Aerospace Systems

The application of failsafe control in aerospace systems has been a subject of research in the early 2000s. This field has primarily focused on addressing failures in aircraft control surfaces - i.e. the ailerons, elevators and rudders that provide roll, pitch and yaw control - and developing strategies for emergency maneuvers.

In [11], pre-stabilized MPC is applied to the El Al Flight 1862 crash scenario, highlighting that MPC might have prevented the accident. However, the authors admit that in practice, quickly finding the necessary parameters to make the controller work is difficult. The work also lacks rigorous proofs of stability and retracts to simulation results.

Another result from the start of nonlinear MPC theory is [12]. Here, different faults in an airplane are considered and stabilized using MPC techniques. The focus of the work is less on proving the stability of the proposed controllers and more on showing how the adaption of the MPC scheme can be used in general to deal with faults and failures.

More recently, an MPC scheme for extreme maneuvers, s.a. tight turns and flips, and emergency obstacle avoidance has been proposed [13]. The failure of control surfaces like ailerons, elevator, or rudder is modeled by reducing their maximum rate of change by

orders. While effective in simulations, the paper, as well as the others, shows no formal stability proof. The paper also claims the ability to track infeasible trajectories. While simulation results demonstrate this capability, the authors do not present a dedicated methodology for ensuring the tracking of infeasible trajectories. Instead, the approach relies on the inherent properties of MPC to get as close to a trajectory as possible. Consequently, the limitations and conditions under which tracking infeasible trajectories is achievable remain ambiguous.

1.4.2 Automotive Industry

As aforementioned, failsafety in the context of autonomous vehicles is often considered in terms of the behavior of the other traffic participants. An important topic here is failsafe trajectory planning (FTP), see [14, 15]. Safety is achieved based on analysis of reachability resp. occupancy prediction under the dynamics of other participants. More advanced solutions have combined stochastic MPC with FTP [16], enhancing the robustness of autonomous driving systems. Some researchers have explored the combination of different MPC algorithms, each with varying degrees of failsafety, performance, and optimality, to create more versatile control systems [17].

Besides of autonomous driving, there are also papers that treat actuator failures in vehicles: The advance of electric motors and cars allows for new vehicle architectures such as Four-Wheel Independent Drive/Steering systems (4WIDS) where each wheel is controlled independently by its own motor [18]. While this gives more degrees of freedom regarding the maneuverability of the vehicle, the risk of instability rises if one of the subsystems fails. Control methods that try to counteract this risk are reviewed in [18]. The methods used for ensuring maneuverability include classical nonlinear control s.a. Sliding Mode Control [19, 20, 21] or gain scheduling for linear parameter-varying systems [22].

Recently, also optimization-based solutions to the problem of actuator failures of 4WIDS vehicles have been proposed. In [23], Nonlinear Model Predictive Control (NMPC) is applied to four-wheeled vehicles under actuator faults. The faults are modeled as power steering failure (50% reduction in steering output) and uncertainty in the rear cornering stiffness (50% reduction in the rear cornering stiffness parameter). The goal is to bring the vehicle safely to the side of the road. In a first step, the fault is assessed and a finite state machine is used to determine an appropriate lane changing strategy. In the following, the NMPC controller is used to execute the strategy while taking the failure into account. The stability of these controllers, however, is not analyzed theoretically - instead, simulations and scenario testing are employed to assess and validate their performance and reliability.

1.4.3 Unmanned Aerial Vehicles (UAV) and Multicopters

Small multicopters have gained popularity since the end of the 2000s and soon failsafe controllers have been developed that can deal with the loss of actuators. In [24], feedback-based controllers are designed for quadrocopters under the loss of one, two or even three actuators. The idea is to analyze the systems under failures for possibly periodic solutions for certain variables. It is found that if certain control goals such as the control of the yaw

axis are given up on, the quadrocopter can still be kept in the air and safely landed. At the same time, a periodic solution can occur for the non-controlled variables. The control structure employed a cascaded design with an outer loop for position control and an inner loop for attitude control. This separation allowed for high-frequency attitude control to maintain stability while the slower outer loop guided the overall position.

Building upon this work, [25] develops a fault detection architecture that, when combined with the controller, resulted in a fully failsafe system. Both studies validated their designed controllers both through a proof of stability and experimental implementation on a real system, demonstrating the practical applicability of their approaches.

The problem considered in [26] is that controllers based on (N)MPC are potentially a good solution to dealing with actuator failures or other constraints. They give however cause for concern that the model needs to be fairly accurate to be suitable for MPC. Thus, the paper shows the development of an observer that is exact enough to be suitable for an observer-controller system that uses MPC. It concludes that it is possible to design suitably well-working observers, but shows stability only empirically through simulations.

One of the few studies to implement MPC methods on a real UAV system was conducted by Nan et al. [27]. They addressed the complete failure of one rotor in a quadcopter using NMPC. Their implementation allowed for quick switching from the fault-free to a faulty case by discarding yaw motion control (similar to [24]) and adjusting the cost matrices accordingly. The latter allows to abstain from reloading and initializing a completely new controller, making the switching process from the nominal to the faulty case very fast. While stability was established through tuning and testing rather than theoretical analysis, the results were impressive. The NMPC controller successfully leveraged nonlinear dynamics to stabilize the quadcopter even during agile flight maneuvers like loops.

A similar NMPC-based approach is applied in [28] for the failure of one rotor of a hexacopter. This paper includes an Extended Kalman filter for fault identification and equally considers the terminal cost as a tuning parameter without direct theoretical justification. Although the impact of a single motor failure is generally less severe for a hexacopter compared to a quadcopter, their work further demonstrated the versatility of MPC-based failsafe control strategies in multi-rotor UAV systems.

1.4.4 Space Systems

The control of spacecrafts under actuator failures is closely related to the control of underactuated spacecrafts, as a failed actuator can be turned off and the system becomes underactuated. Most literature on underactuated spacecraft control focuses on attitude control, primarily using nonlinear control methods to establish stability. These studies predominantly date from the 1980s to early 2000s.

There are two ways of controlling the attitude of satellites, either through momentum wheels or through thrusters [29, ch. 3.1]. Momentum wheel failures do not necessarily affect position control, whereas thruster failures impact both translational and rotational control. As thrusters affect both motions, the control is usually decoupled into resulting

forces and torques and a subsequent module, called the control allocator, restores the physical thrust forces. This is traditionally done for the nominal control without failures as well as for the underactuated one. However, the previously mentioned control for underactuated systems schemes suffer from two problems:

Firstly, it has been shown that for underactuated spacecrafts, the decoupling can lead to infeasible combinations of forces and torques that the control allocator may not be able to produce [30, 31]. And secondly, most of the proposed methods do not consider actuator constraints, thereby producing control signals that are not necessarily practically applicable, even without considering the control allocator.

Nevertheless, some of the results are cited here for the sake of completeness: The first results on stabilization of underactuated spacecraft dynamics were performed in [32]. There, statements on the controllability of rigid bodies with momentum exchange devices and gas thrusters were established. The result was that the attitude stabilization is only possible with three or more momentum exchange devices or two or more independent paired (jet) thrusters. Controllability in the case of thrusters further requires the two pairs, resp. four individual thrusters, to satisfy additional constraints on the geometry of the thruster positioning. The restriction to thruster pairs also implies that if one thruster of a thruster pair fails, the other (functioning) one also needs to be turned off.

The next few papers all build on these results of [32] and have the same implications.

First of all, it has been shown that if spacecrafts are only controlled with the two independent thrusters from [32], then the feedback cannot be smooth [33]. However, a controller based on geometric control theory is proposed, which controls two out of three Euler angles. Further efforts lead to a hybrid controller for underactuated spacecrafts with two momentum wheels [34]. This controller stabilizes any equilibrium attitude in finite time if the total angular momentum vector of the system is 0. Thus, it does not provide full stabilization but rather reorientation if the system is already at rest *before* the control law is applied. In another approach, an underactuated spacecraft is detumbled using a variable structure controller and afterwards reoriented using a series of rotations performed by linear feedback [35].

Newer results are a switching control scheme that globally asymptotically stabilizes the attitude and angular velocities of an underactuated spacecraft [36]. Another more recent work addresses the attitude control of an underactuated spacecraft using perturbed feedback linearization [37]. The inverse that arises during feedback linearization is damped, which yields stability within a domain of attraction. The piecewise-smooth control law stabilizes the system near zero error, but not exactly to zero. As such, it does not contradict the results of [33], although the remaining error can be made arbitrarily small.

As all of these papers build on [32], none of these can exploit the full potential of the actuators. In these approaches, when one thruster fails, the opposite thruster must also be deactivated, leading to suboptimal utilization of the remaining actuators.

It should be noted that while these control laws were developed, systems as the Astrobee were not yet thought of. The controllers were instead designed for satellites in Earth's

orbit, with very different requirements: First of all, the reaction does not need to be fast as satellites are typically far away from each other. Collision avoidance or other fast maneuvers are therefore not necessary. In fact, due to limited fuel generally slow maneuvers are preferred [29, ch. 3.1] wherefore actuator constraints are less important. The full utilization of all inputs is thus not as important. Additionally, the Astrobee belongs to a lightweight class of systems with fewer redundancies. For satellites, in contrast, hardware redundancies are more common and failures do not need to be compensated with advanced control method in all cases.

Developing explicit control laws capable of handling non-paired thrust actuators presents significant theoretical and practical challenges. To the best of the authors' knowledge, explicit methods addressing this issue have not yet been proposed. However, the development of implicit optimization-based controllers has yielded some promising results in the control of underactuated systems. These approaches actually exploit the full dynamics of the system and will be discussed in the following.

An analysis of the SPHERES project that preceded the Astrobees showed that the loss of a single actuator can already imply not only the emergence of nonholonomic constraints, but even the loss of small-time local controllability [30]. This happens even though the SPHERES have more than two remaining thruster pairs. Still, this result does not contradict [32] as the work considers the concurrent control of position and attitude together.

The author continued his work with the paper that comes closest to the issue treated in this work [31]. It considers the combination of translational and rotational control of spacecrafts and uses MPC to deal with the constraints. Its main disadvantage is that it uses the terminal set $\{\mathbf{0}\}$, i.e. it requires the optimizer to steer the system to the origin within the control horizon. It is well known that, while this is probably the most straightforward way of ensuring stability, this method suffers from small feasible sets and the need of very long prediction horizons. Thus, it is not always possible to implement on constrained hardware. Additionally, the controller does not consider tracking which can be advantageous for bringing a damaged system back to a point where it can be repaired.

In the last four years, Aguilar-Marsillach et al. [38, 39, 40, 41] have published several papers on spacecraft rendezvous in the case of actuator failures. The central concept in these papers is the calculation of unsafe sets that will lead to a collision if one or multiple actuators fail. This is achieved using reachability analysis and the calculation of backwards reachable sets to determine initial states that would result in collision. The papers explore different types of safety, such as passive safety (where the chaser naturally drifts past the chased) or abort safety (where collision is avoided using remaining thrusters). They also consider various scenarios, including elliptical orbits and near-rectilinear halo orbits around the Moon, as well as methods to reduce fuel consumption.

While these papers do not provide a formal stability proof, established results strongly suggest that finding a proof of stability for the nominal system is possible. However, for the system under failures this is no longer guaranteed, as will become clear in the course of this work.

In the context of the papers, this lack of formal stability is however not a problem: Their objective is to find a controller that either guarantees to reach the target, or that aborts the mission while preventing damage to the chased spacecraft if a failure occurs. What happens to the spacecraft that is experiencing the failures is not the primary focus of their work. Thus, although these papers are very interesting in general, they address a different question than the one considered in this work. It can be noticed that these papers, based on the theory used, relate most to the ones on FTP for autonomous vehicles.

1.4.5 Conclusion

Generally, both methods from nonlinear control as well as MPC have been applied for failsafe control under actuator failure in the last decades. If the stabilization under failure is done with feedback-methods, more system analysis is usually performed and the resulting controllers can be shown to be stable. One of their shortcomings is however that a lot of them do not consider actuator saturation.

In recent years, the development went towards MPC-based solutions due to the ease of incorporating constraints. While these controllers prove to work in practice, they usually lack stability guarantees or if not, suffer from high computational requirements. As will be shown in this work, it is possible to ensure the stability of an MPC scheme under actuator failures with comparatively little online computational load. In order to do this, however, the control designer has to deal with the underlying, nonlinear and underactuated model.

It has also been shown that few failsafe control methods exist for the modern class of small autonomous space robots. Existing methods mostly target satellites with different requirements that are not designed to perform fast evasion maneuvers and lack the possibility to use the full set of actuators. Despite its importance for the safety of future missions, the control of mobile autonomous space robots is thus underexplored.

Based on these results, the objective of the thesis are shown in the following section, followed by the main contributions of this work.

1.5 Contributions

The last section motivated a new type of small, agile space systems that handle routine and maintenance tasks. Their potential for future space missions has been underlined concerning the reduction of risk and costs of space missions. Furthermore, it has been shown that current control schemes cannot effectively handle actuator failures for these kinds of systems which motivates the need to develop such methods.

Objectives In order to tackle this problem, methods are developed to handle actuator failures of the freeflyers, thus developing the fundamentals of fail-safe control for future space robotic systems. The objective is to design a controller that can avoid crashes even under actuator failures and is able to autonomously navigate the system back to a place where it can be repaired. The original mission is aborted and original control objectives can be disregarded. All this should be achieved while simultaneously being able to show the stability of the developed controller. The latter is necessary to formally ensure mission

safety and guarantee the system's reliable operation during the return journey. Such a system can then become part of a truly autonomous space station.

Scope Throughout the work it is assumed that the actuator failure is known. It has been shown in the previous section that such methods exist and are already used for MPC schemes under actuator failures. The development of fault detection methods is therefore not part of this thesis.

Overview of the work In order to fulfill the objectives, the work uses the following steps:

Step 1 Modeling of the freeflyers with and without actuator failures (Chapter 3).

Step 2 Deriving properties that hold for all combinations of actuator failures (Section 3.3).

It turns out that under different combinations of actuator failures, different constraint sets for the resulting forces and torque follow of which some are nonholonomic. Due to the quantity of different possible nonholonomic constraint cases, it is shown that proving stability for every single case may not be infeasible, but at least practically undesirable. Instead, it is shown that if the system is brought into a small scale orbit, called micro-orbit, all possible cases for up to three failed actuators can be treated. Some cases for more than three actuators are also possible to control.

Step 3 Proposition of micro-orbiting and derivation of the orbit model (Chapter 4).

Based on this model, three different controllers are presented of which all control the position as well as the attitude. The first one is a feedback linearizing controller that can stabilize the system at a desired set point under satisfaction of the actuator constraints. The other two controllers are NMPC controllers with the different terminal ingredients. The first one is designed, borrowing from established MPC controllers, using a linear terminal controller after a feedback linearization step.

As it was found that the resulting terminal set of this controller is too small, another NMPC controller was designed. This one uses a non-linear terminal controller, for which a terminal cost and set are derived. The terminal controller uses a feedback linearizing controller with an added explicit MPC controller where the latter controls the resulting linearized system under actuator constraints. Within the bounds of formulation of explicit MPC and feasibility, the terminal set of the NMPC controller is quasi unbounded. It should be noted that to the best of the author's knowledge, such a controller has not yet been proposed.

Step 4 Derivation of three different controllers, one of them using explicit MPC to derive a large terminal region (Chapter 5).

Finally, the controllers are implemented and simulated. Based on the simulations, their performance is evaluated.

Step 5 Simulation and evaluation of the controller performance (Chapter 6).

Contribution The main contributions of this work are therefore twofold:

- Find a model and operating mode - i.e. micro-orbiting - such that it is possible to handle the nonholonomic constraints appearing from actuator failures with guaranteed stability.

Various controllers are possible for implementing this strategy: the controllers presented in this paper, of course, but also controllers as described in Remark 28, for example. The formulation for micro-orbiting for the 3D case is currently under development.

- As other MPC techniques proved insufficient to control the system, a method using explicit MPC in the terminal controller was developed. This leads to a quasi unbounded terminal constraint only limited by feasibility and the bounds of formulation of explicit MPC.

This kind of controller is not limited to space robotics and can be applied to other systems, although their theoretical capabilities and restrictions have yet to be explored. Some known limitations are discussed in Section 5.3.3.

> **Remark 1:** It should be noted that existing results from Section 1.4.4 that keep spacecrafts rotating around an axis have nothing to do with micro-orbiting. They do not rotate on an orbit but around themselves. Also, the objective is not to overcome the nonholonomic constraints in order to maintain the position maneuverability. Instead, limited controllability of solely the attitude is accepted and the rotating axes are uncontrolled.

The rest of this work is structured as follows: After a short overview over the mathematical notation in the next section, the fundamentals are presented in Chapter 2. Besides of some mathematical foundation, the discussion mainly considers results on MPC. In the following, the key steps as described above are shown. The thesis closes with a summary of the results and looks ahead to questions that are still open.

1.6 Mathematical Notation

This work uses the following mathematical notations: The transpose of a matrix is denoted by $(\cdot)^T$. $\mathbf{0}_{n\times m}$ represents a n-by-m zero matrix and $\boldsymbol{I}_{n\times n}$ a n-by-n identity matrix. A diagonal matrix is expressed as:

$$\operatorname{diag}\begin{pmatrix} a_1 & a_2 & \dots \end{pmatrix} = \begin{pmatrix} a_1 & 0 & \dots \\ 0 & a_2 & 0 \\ \vdots & 0 & \ddots \end{pmatrix}$$

Matrix definiteness is represented by $\boldsymbol{A} \succ 0$ and $\boldsymbol{A} \succeq 0$ for positive definiteness resp. semidefiniteness. For vectors and matrices, $\|(\cdot)\|_2$ represents the Euclidean (L2) norm.

The set of non-negative integers (including zero) is symbolized by $\mathbb{N}_0^+$. Named sets are denoted with calligraphic letters. In the context of sets, the symbol $\partial\mathcal{U}$ signifies the boundary and $\operatorname{Int}(\mathcal{U})$ the interior of a set $\mathcal{U}$. In the context of functions, ∂ indicates a partial derivative.

Chapter 2

Fundamentals

After the statement of the problem and the contributions made in this work, some necessary foundations are presented. The chapter starts in Section 2.1 with mathematical foundations on polytopes and set operations that are useful in their context. In the following, some definitions of system properties are given in Section 2.2. An introduction to MPC and explicit MPC is provided in Section 2.3 and Section 2.4, respectively.

2.1 Mathematical Fundamentals

Convex polytopes are sets that can be described as linear inequalities. They are widely used in mathematics and engineering, e.g. in convex optimization, but also especially in Model Predictive Control. The notation as linear inequality is called the H-representation and can be given as [42, Ch. 2.2.4]

$$\mathcal{P} : \{x \in \mathbb{R}^{n_x} | \boldsymbol{A}x \leq \boldsymbol{b}\}. \tag{2.1}$$

If n_c is the number of inequality constraints, $\boldsymbol{A}$ is here a constant matrix of size $\mathbb{R}^{n_c \times n_x}$ and $\boldsymbol{b} \in \mathbb{R}^{n_c}$ is a vector.

Alternatively, the polytope can also be given in the V-representation. There, the corner points of the polytope, the vertices, are used to describe the set. Every point that is "between" a number of vertices belongs to the set. Both representations can be converted into each other and there are freely available computer programs that can do this automatically. The H-representation is mainly used in this work, but the V-representation is also useful, e.g. to calculate linear projections of a polytope.

The Minkowski set summation and Minkowski (Pontryagin) difference are operations that allow to "grow" resp. "shrink" one set by another. Using the two sets $\mathcal{U}, \mathcal{V} \subset \mathbb{R}^{n_x}$, the Minkowski sum can be defined as [9]

$$\mathcal{U} \oplus \mathcal{V} = \{u + v | u \in \mathcal{U}, v \in \mathcal{V}\}$$

and the Minkowski difference as

$$\mathcal{U} \ominus \mathcal{V} = \{x | x \oplus \mathcal{V} \subseteq \mathcal{U}\}.$$

A Minkowski difference that is particularly easy to calculate is $\mathcal{P} \ominus \mathcal{S}(r)$: Let $\mathcal{P}$ be a polygon as in Eq. (2.1) and $\mathcal{S}(r) : \{x | x^T x \leq r\}$ be a sphere, then each inequality needs to be parallelly shifted inwards by r. This is easy to see by the definition of the Minkowski difference and the fact is visualized in Figure 2.1.

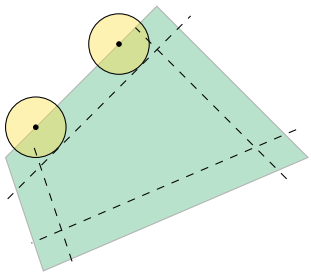


Figure 2.1: Minkowski difference $\mathcal{P} \ominus \mathcal{S}(r)$ with polygon $\mathcal{P}$ and sphere $\mathcal{S}(r)$.

The resulting polytope can be calculated as follows: Interpret the polytope in H-representation as the combination of n_c linear inequalities $a_i x \leq b_i, i \in 1, \ldots, n_c$. Here, a_i are the row vectors of $\boldsymbol{A}$ and b_i the corresponding elements of $\boldsymbol{b}$. Then the boundary of each inequality can be written as a plane in $\mathbb{R}^{n_x}$ in the form $a_i(x - v_i) = 0$ where supporting vectors v_i are introduced that satisfy $a_i v_i = b_i$.

Each plane is then moved inwards in the direction of the normal vector a_i^T of the plane by distance r.

$$0 = a_i \left(x - v_i + \zeta(a_i, v_i) r \frac{a_i}{\|a_i\|} \right) \tag{2.2}$$

The correct sign to move towards the set can be calculated as function $\zeta(a_i, v_i)$ with any point $\iota \in \text{Int}(\mathcal{P})$ as

$$\zeta(a_i, v_i) = -\text{sign}(a_i(v_i - \iota)).$$

By multiplying Eq. (2.2) and performing this operation for all rows, a new matrix inequality can be calculated. The matrix $\boldsymbol{A}$ will remain unchanged, but the vector $\boldsymbol{b}$ is replaced.

> **Remark 2:** Observe in Section 2.1 that r becomes a single direct parameter of the resulting matrix inequality - all other values can be calculated directly. This will become useful in Section 5.3.3.

Very similar results can be obtained if, for example, the set $\mathcal{S}(r)$ is replaced by a line segment, i.e. the set $\mathcal{L} = \{t\boldsymbol{w} | \boldsymbol{w} \in \mathbb{R}^{n_x}, t \in (-t_{\max}, t_{\max}) \subset \mathbb{R}\}$ with a fixed $\boldsymbol{w} \in \mathbb{R}^{n_x}$. In

this case, the expression can be given by

$$\mathcal{P} \ominus \mathcal{L} = \{x \in \mathbb{R}^{n_x} | 0 = a_i \left(x - v_i + \zeta(\boldsymbol{w}, v_i)\boldsymbol{w}\right)\}$$

As a last remark, it follows directly from the definition of the Minkowski sum and difference that for any $\mathcal{S}(r_1)$, the relationship $\mathcal{S}(r_1) \subseteq \mathcal{P} \ominus \mathcal{S}(r_2)$ implies $\mathcal{S}(r_1 + r_2) \subseteq \mathcal{P}$, resp. $\mathcal{S}(r_1) \oplus \mathcal{S}(r_2) \subseteq \mathcal{P}$.

2.2 Control Fundamentals

In this work, the knowledge of certain fundamentals of linear control theory is presumed. This includes the notion of Lyapunov stability for continuous and discrete-time systems, the continuous and discrete-time Lyapunov equation and the Riccati equation for discrete-time systems, including their conditions for solvability. In case of questions regarding these topics, the book of Hespanha can be warmly recommended [43].

In the following, some definitions are given that consider certain system properties and the feasibility of trajectories with respect to system dynamics. A system that is nonholonomic has certain nonintegrable constraints. These can be defined as follows [44, p. 242 ff.]:

Definition 1 (Holonomicity and Nonholonomicity): Consider a system

$$\dot{x} = f(x_1, x_2, \ldots, t)$$

with constraints

$$g(x_1, x_2, \ldots, t) = 0$$

where f, g are smooth functions. These constraints are called *holonomic* constraints. Every constraint that is *not* expressible like this is called a *nonholonomic* constraint. In other words, nonholonomic constraints are nonintegrable.

A system that has nonholonomic constraints is called a nonholonomic system and the property itself is called *nonholonomicity*.

It is important to know if nonholonomic constraints exist as Brockett's Theorem states that nonholonomic systems can not be stabilized using smooth time-invariant feedback [45].

Dynamic feasibility is defined, following [46], as follows:

Definition 2 (Dynamic Feasibility): Let $\dot{x} = f(x, u)$ be the dynamics of a system with input constraints $u \in \mathcal{U}$ and state constraints $x \in \mathcal{X}$. The trajectory $\chi(t)$ is called *continuously dynamically feasible* if

- it is continuously differentiable and $\chi(t) \in \mathcal{X}$ for all $t \geq 0$

- there exists the input $\tilde{u}(t) \in \tilde{\mathcal{U}}$ that is necessary to perfectly follow the trajectory for all $t \geq 0$, i.e.
- $\chi(t)$ satisfies the system dynamics $\dot{\chi}(t) = f(\chi(t), \tilde{u}(t))$ for all $t \geq 0$ and
- the set of remaining control inputs $\bar{\mathcal{U}} = \mathcal{U} \ominus \tilde{\mathcal{U}}$ contains zero in its interior.

It is called *discretely dynamically feasible* if

- it satisfies the system dynamics $\chi_{(k+1)\delta} = f(\chi_{k\delta}, \tilde{u}_{k\delta})$ at the sampling times $k\delta$ with the step k and the sampling period δ,
- $\chi_{k\delta} \in \mathcal{X}$ and $\tilde{u}_{k\delta} \in \tilde{\mathcal{U}}$ for $i \in \mathbb{N}_0^+$,
- the set of remaining control inputs $\bar{\mathcal{U}} = \mathcal{U} \ominus \tilde{\mathcal{U}}$ contains zero in its interior and
- $\chi(t)$ is continuously differentiable for all $t \in [k\delta, (k+1)\delta]$.

The interpretation of this is that on the one hand, there is enough control action to perfectly steer the system along the trajectory. Additionally, there must always be some spare degree of freely choosable control input to control the system towards the trajectory under errors.

As a last note, the solution to the differential equation of a linear, time-continuous autonomous system with $x \in \mathbb{R}^{n_x}$, $\boldsymbol{A} \in \mathbb{R}^{n_x \times n_x}$ and dynamics $\dot{x} = \boldsymbol{A}x$ can be given explicitly and is $x(t) = e^{\boldsymbol{A}t}x(0)$ for every $t \geq 0$ [47]. For a discrete-time autonomous linear system $x_{k+1} = \boldsymbol{A}x_k$ and same dimensions, the solution is $x_k = \boldsymbol{A}^k x_0$ for all $k \in \mathbb{N}_0^+$.

2.3 Classic Model Predictive Control

Model Predictive control (MPC) is a modern optimization-based control technique. Its main difference to traditional approaches is its ability to handle a wide variety of constraints. MPC was originally used in process industry with slow dynamics, but has been extended in the previous decades to fast-sampled systems. Today, MPC is considered a mature technology with applications in many different areas [48].

MPC is an optimal control method in the sense that a constrained optimization problem is formulated and solved at each sampling step. Depending on an initial state, an optimal input sequence is computed that minimizes the control cost under satisfaction of input and state constraints. In general, this solution cannot practically be found in a closed form. Once the optimum has been found, only the first element of the sequence is applied to the system before a new optimization with an updated initial state is computed. This procedure is repeated for every step, leading to a so-called receding horizon strategy.

There are many different variants and formulations of MPC for a wide range of applications. In this place, the standard formulation is presented, an extension for the tracking of trajectories and finally explicit MPC, a closed-form solution for linear MPC.

2.3.1 Standard Formulation

The standard MPC is the common control problem of point stabilization at the origin. It optimizes both the error as well as the control effort and takes input and state constraints

into account. The following statements are based on the overviews presented in the PhD theses [49, 50, 51].

Consider a system with $x_t \in \mathbb{R}^{n_x}$, $u_t \in \mathbb{R}^{n_u}$ and the system dynamics

$$x_{t+1} = \Phi(x_t, u_t). \tag{2.3}$$

The predicted state and input k steps into the future based on time step t are $x_{t+k|t}$, $u_{t+k|t}$ and will be abbreviated as x_k, u_k if the context is clear. The predicted trajectories of state and input up to $N-1$ with the finite horizon N are given in matrix-form as

$$\begin{aligned} X &= \left(x_k^T, x_{k+1}^T, \ldots, x_{k+N-1}^T\right)^T \text{and} \\ U &= \left(u_k^T, u_{k+1}^T, \ldots, u_{k+N-1}^T\right)^T, \end{aligned} \tag{2.4}$$

respectively. The constraints are

$$x_k \in \mathcal{X} \subseteq \mathbb{R}^{n_x}, \quad \forall k \in \mathbb{N}_0^+, \tag{2.5}$$

$$u_k \in \mathcal{U} \subseteq \mathbb{R}^{n_u}, \quad \forall k \in \mathbb{N}_0^+ \tag{2.6}$$

and can for example be linear inequalities, in which case $\mathcal{X}$ and $\mathcal{U}$ are polytopes.

The (scalar) cost is defined as

$$V_N(x_0) = \sum_{k=0}^{N-1} l(x_k, u_k) + l_N(x_N) \tag{2.7}$$

with the stage costs $l(x_k, u_k)$ and terminal cost $l_N(x_N)$. It is assumed that the stage cost is convex, $l(0,0) = 0$ and there exists a class κ_a function[1] $\alpha_l(\cdot)$ such that $l(x,u) \geq \alpha_l(\|x\|)$. A common choice, which is also employed in this work, is a quadratic function with symmetric, positive definite cost matrices Q_x and Q_u:

$$l(x_k, u_k) := x_k^T Q_x x_k + u_k^T Q_u u_k \tag{2.8}$$

The terminal cost $l_N(x_N)$ should satisfy similar constraints concerning positive definiteness and the behavior at the origin, but these properties are discussed later in detail when the stability of MPC is concerned.

This allows to formulate the standard MPC problem:

$$\underset{U}{\text{minimize}} \quad V_N(x_0) = \sum_{k=0}^{N-1} l(x_{t+k|t}, u_{t+k|t}) + l_N(x_{t+N|t}) \tag{2.9a}$$

$$\text{subject to} \quad x_{t+k+1|t} = \Phi(x_{t+k|t}, u_{t+k|t}) \quad k = 0, \ldots, N-1, \tag{2.9b}$$

$$(x_{t+k|t}, u_{t+k|t}) \in \mathcal{X} \times \mathcal{U} \qquad k = 0, \ldots, N-1, \tag{2.9c}$$

$$x_{t+N|t} \in \mathcal{X}_{\text{term}}, \tag{2.9d}$$

$$x_{t|t} = x_0 \tag{2.9e}$$

[1] A function $\alpha_\kappa : [0, a) \to [0, \infty)$ belongs to class κ_a if it is strictly increasing and $\alpha_\kappa(0) = 0$.

where the terminal constraint (2.9d) with the terminal set $\mathcal{X}_{\text{term}}$ can be used together with $l_N(x_N)$ to enforce stability as will be shown soon. The set of feasible control sequences is implicitly defined as $\{U|\exists X$ s.t. (2.9b) - (2.9e) hold$\}$ while the set of feasible initial states is the one that makes the former set nonempty.

By definition of the problem, the satisfaction of the constraints is only directly guaranteed until $k = N - 1$. Additionally, this definition itself does not make any statements on the stability of the resulting receding horizon scheme. While multiple approaches exist to guarantee stability of the controller, the most widely used is the terminal cost and terminal set approach.

The key idea is based on the finding that for the *infinite-horizon* MPC problem, the system is asymptotically stable and the problem recursively feasible [52]. Therefore, $l_N(\cdot)$ is chosen so that $V_N(\cdot,\cdot)$ is approximately equal to or an upper bound on the infinite horizon cost in a region around the origin. The terminal set is chosen to be a subset of this region such that all constraints are satisfied within the subset. One way to determine a terminal cost and set is using a terminal controller. This is a known controller that can locally stabilize the system within a certain region. For linear systems, for example, the infinite-horizon LQR-controller and its corresponding cost function can be used.

Before a theorem on stability of MPC can be given, the definitions of recursive feasibility and control invariant sets are necessary.

Definition 3 (Recursive Feasibility): The MPC problem (2.9) is *recursively feasible* if the existence of a feasible solution with initial state x_k implies that the MPC problem is also recursively feasible if it is initialized with x_{k+1}.

An important statement on recursive feasibility can be made using the notion of control invariant sets [53]:

Definition 4 (Control Invariant Set): The set $\mathcal{C} \subseteq \mathbb{R}^n$ is positively control invariant for the dynamics in Eq. (2.3) and the control constraint $u_k \in \mathcal{U}$ if

$$x_k \in \mathcal{C} \Rightarrow \exists\{u_k, u_{k+1}, \ldots\} \text{ with } u_k \in \mathcal{U} \text{ such that } x_k \in \mathcal{C}\ \forall k \in \mathbb{N}_0^+ \tag{2.10}$$

Theorem 1: If the terminal set $\mathcal{X}_{\text{term}}$ is control invariant, then the MPC problem (2.9) is recursively feasible.

Stability of a nonlinear MPC scheme can be guaranteed as follows [54, 55]:

Theorem 2 (Stability of Model Predictive Control): Consider the nonlinear system $x_{k+1} = \Phi(x_k, u_k)$ subject to the constraints $x_k \in \mathcal{X}$, $u_k \in \mathcal{U}$ for all $k > 0$ under the Model Predictive Control (2.9). Consider the conditions

- The stage cost $l(x_k, u_k)$ is continuous, $l(0,0) = 0$ and lower-bounded by a class κ_∞ function α_1 such that $l(x_k, u_k) \geq \alpha_1(\|x_k\|)$ for all x_k, u_k.

- The terminal cost $l_N(x_N)$ satisfies $l_N(0) = 0$, $l_N(x) > 0 \; \forall x \neq 0$ and $l_N(x_N)$ is continuous at $x_N = 0$.
- $\mathcal{X}$, $\mathcal{U}$ and $\mathcal{X}_{\text{term}}$ are closed and contain zero in their interior.
- $\mathcal{X}_{\text{term}}$ is control invariant under the given dynamics and constraints.
- For all $x \in \mathcal{X}_{\text{term}}$, at least one $u \in \mathcal{U}$ exists such that

$$l_N(\Phi(x, u)) - l_N(x) + l(x, u) \leq 0. \tag{2.11}$$

If all of the conditions are met, then the MPC controller asymptotically stabilizes the system, i.e. $x_k \to 0$ as $k \to \infty$.

2.3.2 Tracking Trajectories with MPC

In the standard MPC scheme, the system was controlled to a setpoint and the setpoint was implicitly set to zero: $e_k = x_k - \mathbf{0} = x_k$. If instead a trajectory is supposed to be tracked, the definition of the error changes to

$$e_k = x_k - \chi_k \tag{2.12}$$

where χ_k with $k \in \mathbb{N}_0^+$ is the trajectory to be tracked. This means that the error dynamics depend explicitly on time, which needs to be addressed specifically. This is the next topic discussed, with the results taken from [56] if not otherwise noted.

The tracking problem is here formulated in terms of the states. In this case, a necessary assumption is

(a1) The reference lies within the feasible set of states, $\chi_k \in \mathcal{X} \; \forall k \geq 0$.

Additionally,

(a2) The reference χ_t is dynamically feasible in the sense of Definition 2.

(a3) The input $\tilde{u}(t)$ (ref. Definition 2) that is necessary to follow the trajectory perfectly is known a priori.

Remark 3: An alternative formulation with a reference in the output values of the system would lead to the assumption

(a1) Alternative The reference χ_t is contained in the pointwise image of $\mathcal{X}$ under the output map.

Introducing the new input $\bar{u}_t = u_t - \tilde{u}_t$ allows to formulate the tracking MPC problem

$$\begin{aligned}
\underset{\bar{U}}{\text{minimize}} \quad & \sum_{k=0}^{N-1} l(e_{t+k|t}, \bar{u}_{t|t+k}) + l_N(e_{t+N|t}) && && \text{(2.13a)}\\
\text{subject to} \quad & x_{t+k+1|t} = \Phi(x_{t+k|t}, \bar{u}_{t|t+k} + \tilde{u}_{t|t+k}) && k = 0, \dots, N-1, && \text{(2.13b)}\\
& e_{t+k|t} = x_{t+k|t} - \chi_{t+k|t} && k \in \mathbb{N}_0^+, && \text{(2.13c)}\\
& (x_{t+k|t}, u_{t+k|t}) \in \mathcal{X} \times \mathcal{U} && k = 0, \dots, N-1, && \text{(2.13d)}\\
& e_{t+N|t} \in \mathcal{X}_{\text{term}}, && && \text{(2.13e)}\\
& x_{t|t} = x_0 && && \text{(2.13f)}
\end{aligned}$$

Note that this is essentially a compensation of the time-varying part such that the remaining MPC problem is just a setpoint stabilization of the error through the remaining input $\bar{u}_t$.

The following assumptions ensure convergence of the tracking MPC [57, 58]:

(a4) The stage cost $l(e_k, \bar{u}_k)$ is continuous, $l(0,0) = 0$ and lower bounded by a class κ_∞ function α_m such that $l(e_k, \bar{u}_k) \geq \alpha_m(\|e_k\|)$ for all $(e_k, \bar{u}_k)$.

(a5) The terminal cost $l_N(e_N)$ is positive semi-definite and continuously differentiable in e_k.

(a6) The terminal set $\mathcal{X}_{\text{term}} \subseteq \mathcal{X}$ is compact and time-varying.

(a7) For all $\tilde{x} \in \mathcal{X}_{\text{term}}$ and the considered sampling time $\delta > 0$, there exists a piecewise continuous admissible input $u_{\mathcal{X}_{\text{term}}} \in \mathcal{U}$, such that for all $\tau \in [0, \delta)$,

$$\frac{\partial l_N}{\partial e} \cdot \Phi(e(\tau), u_{\mathcal{X}_{\text{term}}}(\tau)) + l(e(\tau), \bar{u}(\tau)) \leq 0 \tag{2.14}$$

and the closed-loop solution fulfills $x(\tau) = x(\tau, \tilde{x}|u_{\mathcal{X}_{\text{term}}}) \in \mathcal{X}_{\text{term}}$, i.e. the terminal region is control invariant.

Note that the last assumption is formulated in terms of the continuous-time dynamics and not the discrete ones.

The main difference compared to setpoint stabilization is that $\mathcal{X}_{\text{term}}$ is in general time-dependent. A way to construct this is given in [57, 58]. If instead suitable assumptions on the trajectory are made, it is possible to construct constant terminal sets at the cost of conservativeness:

Let $\tilde{\mathcal{U}}$ and $\bar{\mathcal{U}}$ be

$$\tilde{\mathcal{U}} = \{\tilde{u}_k | \chi_k \text{ perfectly followed by } \tilde{u}_k, k \in \mathbb{N}_0^+\} \tag{2.15}$$

$$\bar{\mathcal{U}} = \mathcal{U} \ominus \tilde{\mathcal{U}}. \tag{2.16}$$

Furthermore, choose $\mathcal{X}_{\text{term}}$ such that it is control invariant for the system dynamics and the control constraint $\bar{\mathcal{U}}$. Then, $\mathcal{X}_{\text{term}}$ guarantees recursive feasibility for problem (2.13) due to Eq. (2.16) and Theorems 1 and 2.

The given assumptions ensure closed loop stability according to the following theorem:

Theorem 3 ([57, 58]): If the optimal control problem (2.13) is feasible for the initial time instant and the stage cost, the terminal cost and the terminal constraints satisfy assumptions (a1) to (a7), then (2.13) is recursively feasible and the tracking error e_k converges to zero.

2.4 Explicit MPC

Up to now, the notion of MPC was always valid for both linear and nonlinear systems. The solution for each step was implicit and only retrieved by numerically optimizing for a given starting point. If the consideration is however restricted to linear systems with convex polytopic constraints, the controller can be retrieved in an *explicit* form. The control law turns out to be piecewise linear and continuous with a piecewise quadratic and continuous cost function. This section is adapted from the seminal paper [59]. For a good overview over applications and possibilities in explicit MPC (eMPC), the reader may also be referred to [60].

Consider the discrete-time linear time-invariant system

$$\begin{aligned} x_{t+1} &= Ax_t + Bu_t \\ y_t &= Cx_t \end{aligned} \tag{2.17}$$

and formulate the optimization problem as

$$\begin{aligned} \underset{U}{\text{minimize}} \quad & V(U, x_t) = x_{t+N|t}^T P x_{t+N|t} + \sum_{k=0}^{N-1} [x_{t+k|t}^T Q_x x_{t+k|t} + (u_{t+k|t})^T Q_u u_{t+k|t}] \\ \text{subject to} \quad & x_{t+k|t} \in \mathcal{X} \qquad k = 1, \dots, N, \\ & u_{t+k|t} \in \mathcal{U} \qquad k = 0, \dots, N-1, \\ & x_{t|t} = x_t, \\ & x_{t+k+1|t} = Ax_{t+k|t} + Bu_{t+k} \quad k \geq 0, \\ & x_{t+N|t} \in \mathcal{X}_{\text{term}} \end{aligned} \tag{2.18}$$

where $\mathcal{X}$ and $\mathcal{U}$ are polytopes defined by linear inequalities. A good choice for P is the solution of the discrete-time algebraic Riccati equation

$$P = A^T P A - (A^T P B)(Q_u + B^T P B^{-1}(B^T P A) + Q_x$$

which also directly gives a terminal controller

$$\kappa(x) = -Q_u^{-1} B^T P x.$$

The terminal set can be chosen as the largest control invariant set that satisfies the input and state constraints.

The key idea of eMPC is twofold: First of all, it is noticed that up to step N, all inputs, states and outputs can be written in a block form such as

$$X = \begin{pmatrix} I \\ A \\ A^2 \\ \vdots \\ A^{N-1} \end{pmatrix} x_t + \begin{pmatrix} 0 & \dots & & 0 \\ B & 0 & \dots & 0 \\ AB & B & \dots & 0 \\ \vdots & \vdots & \ddots & 0 \\ A^{N-2} & A^{N-3} & \dots & B \end{pmatrix} U \tag{2.19}$$

where the definitions of X and U remain as given in Eq. (2.4). Also the constraints and the objective function of the MPC problem can be written in block form which leads to the expression

$$\begin{aligned} \underset{U}{\text{minimize}} \quad & V(x_t) = \frac{1}{2} x_t^T Y x_t + \frac{1}{2} U^T H U + x_t^T F U \\ \text{subject to} \quad & GU \leq W + E x_t \end{aligned} \tag{2.20}$$

where Y, H, F, G and E follow by direct computation.

The second important observation made is that the expression in Eq. (2.20) can in fact be solved parametrically using the Karush-Kuhn-Tucker (KKT) conditions for optimality. This is achieved using a state transform

$$z = U + H^{-1} F^T x_t \tag{2.21}$$

with which Eq. (2.20) can be equivalently rewritten as

$$\begin{aligned} \underset{z}{\text{minimize}} \quad & V_z(x_t) = \frac{1}{2} z^T H z \\ \text{subject to} \quad & Gz \leq W + S x_t. \end{aligned} \tag{2.22}$$

Using the KKT optimality conditions for one *specific* point in state-space, a convex polytopic set of points can be found in which all points have the same active constraints. A constraint is active if the corresponding row in the condition Eq. (2.22) can be written as an equality. Within this set, the control law and cost functions turn out to be linear resp. quadratic.

As the ultimate goal is to find the control laws for all feasible states, an algorithm is proposed based on these findings: Starting from an arbitrary point in state space, the set with identical active constraints is identified (depicted in orange in the left panel of Figure 2.2). The algorithm then proceeds by inverting one constraint at a time to generate new candidate sets. For each new set, a point within its boundaries is selected, and the active constraints for that point are determined. This process is iterated until the entire state space has been covered and all control laws have been determined.

Once the whole state space has been covered, all control laws are available. In practice, for applying the eMPC controller, first the region is checked in which the current state lies and subsequently, the respective control law is applied.

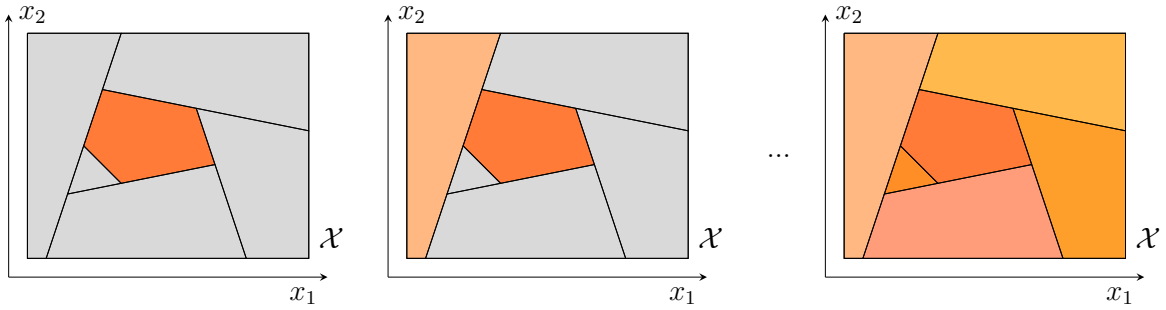


Figure 2.2: Progression of the explored regions in explicit MPC, starting from the set in the center. Drawn in adaption of [59].

While the method allows to solve constrained optimal control problems in a parametric form, the exploration algorithm can possibly take a lot of time. Especially for large horizons and high-dimensional spaces, it scales unfavorably. This is why eMPC is usually only applied in smaller problems. Another problem can be that checking for the correct set might be cumbersome for a lot of active sets and for controllers with limited memory space, even storing all possible sets can become a problem. Even though there are several proposals to resolve both problems [61, 62, 63, 64], no final solution to all of these problems has been found yet.

2.5 Conclusion

This chapter has discussed the necessary fundamentals of this work. After some mathematical properties of polytopes and Minkowski sums, the definition of nonholonomic constraints was given. Next, the dynamic feasibility of trajectories was defined and the chapter concluded with two larger sections considering MPC and eMPC.

Next, the freeflyers are modeled and the effect of actuator constraints on their dynamics is discussed. The result will be that the faulty system has nonholonomic constraints and reduced controllability.

Chapter 3

Modelling and System Analysis

In this chapter, the freeflyers will be modeled and analyzed. A focus lies on the analysis of their behavior under actuator failures. The chapter starts with a description of the system in Section 3.1 and presents the system equations in Section 3.1.2. In that section, it is explained how the input space is structured and subsequently in Section 3.1.3, the effect of actuator failures on the input space is analyzed.

The result is that for some scenarios, existing control methods are able to deal with the failures. For other scenarios, this is not the case any more as nonholonomic constraints appear. Thus, a more formal definition of the control problem that allows for a distinction between already solved and unsolved cases is given in Section 3.2. The last section of this chapter discusses the occurring challenges of ensuring stability under actuator failures.

3.1 Modeling

3.1.1 Actuator Failures

As stated in Section 1.2, the freeflyers at SRL serve as the simulation platform for this work. The robots, as shown in Figure 3.1, hover on air-bearings on a sufficiently smooth surface. Therefore, they behave as the planar version of a 3D Astrobee in microgravity.

The freeflyers are not only made hovering through air pressure, but as Astrobees, they are also propelled by it. By pushing out air into one direction, the system accelerates into the opposite one.

In the case of the Astrobee, the necessary air pressure is created through a central compressor. At several points on the robot, so-called dual flapper nozzles are installed [65], see Figure 3.2. These gate-shaped structures open and close to cover the opening of the outflowing air, determining the resulting force through the width of the free area. Through the positioning of several nozzles on different sides of the robot, maneuverability into all directions and orientations is ensured.

The freeflyer, in contrast, uses an air tank and solenoid valves that open and close discretely and is controlled using Pulse Width Modulation (PWM) [7, p. 62]. Also

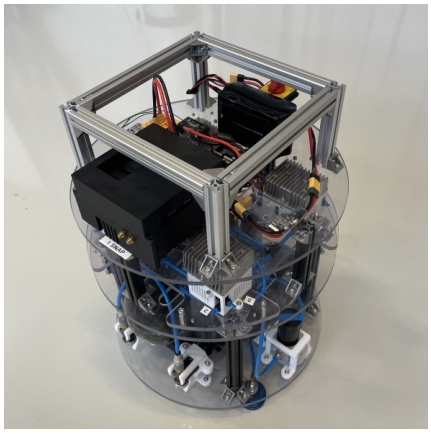


Figure 3.1: The freeflyers at the KTH Space Robotics Lab (SRL).

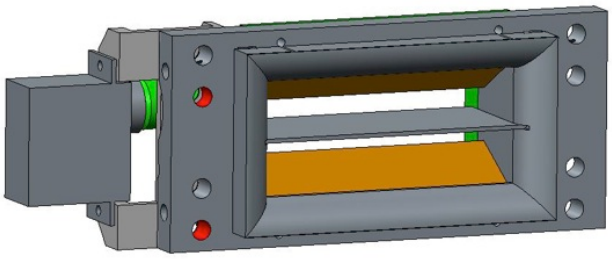


Figure 3.2: The dual flapper nozzle used in the Astrobee. The currently half-opened gates are colored in orange. Image taken from [65].

here, there are multiple independent thrusters on different sides of the robot so that the full motion becomes possible. Figure 3.3 shows the resulting forces for the freeflyer.

The difference in thruster construction affects the behavior under actuator failure: For the freeflyers, the valve is in two discrete cases either fully open or fully closed, i.e. either the maximal force is applied on the robot or no force at all. This naturally also holds for actuator failures.

For the Astrobee in turn, there is a range of continuous failure scenarios that can possibly occur: On the one hand, the compressors could fail, the motors actuating the nozzles break or the feed lines for the thrusters could break. But also the nozzles could get stuck or the communication between the controller and one thruster could be disturbed. In this case, the uncontrollable force could be anywhere between zero and the maximal thrust. In order to maintain generalizability to the Astrobee, the latter case is used in the work. For later reference, the following definition is made:

Definition 5 (Actuator failure): An actuator failure, in the context of this thesis, is the failure of an actuator that renders the actuator uncontrollable. It appears

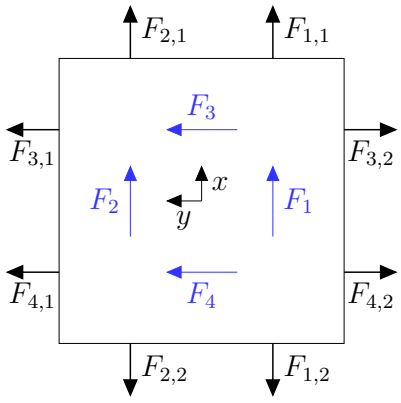


Figure 3.3: The thrust forces acting on the robot.

unpredictably, can affect any of the actuators and multiple failures can be present at the same time. Every failed actuator can propel the system by a non-manipulable force between zero and the maximum thrust and the force is constant from the time it occurs. In accordance with Section 1.5, failures are known.

Remark 4: Even though it is important to model all possible failure-cases, the analysis later in this chapter shows that exactly the extreme scenarios (fully opened/closed) are the ones of the highest interest.

3.1.2 System Equations With and Without Actuator Failures

With Definition 5, the system can be modeled with and without actuator failures. As any manipulable value that can appear for the working actuators can also appear for the non-manipulable failed actuators, it is especially important how the inputs are modeled. Therefore, the following section does not only present a model of the system, but also argues for the chosen formulation.

In order to be able to state the system equations, a couple of variables need to be introduced first: The states x_1 and y_1 denote the position in x- and y-coordinates, x_2 and y_2 in turn are the velocities in the respective directions. α is the orientation as angle w.r.t. the global x-axis and ω is the angular velocity. The time-dependence of these values is not written down explicitly.

Besides of the states, the system constants are the mass m, the moment of inertia J and the lever d of each thruster w.r.t. the center of mass. The matrix $\mathfrak{R}$ is a rotation matrix of the form

$$\mathfrak{R} = \begin{pmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{3.1}$$

This stated, the continuous-time dynamics can directly be derived from Newton's law as follows:

$$f(x) = \begin{pmatrix} \dot{x}_1 \\ \dot{y}_1 \\ \dot{\alpha} \\ \dot{x}_2 \\ \dot{y}_2 \\ \dot{\omega} \end{pmatrix} = \begin{pmatrix} x_2 \\ y_2 \\ \omega \\ 0 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} \mathbf{0}_{3\times 3} \\ \mathfrak{R} \end{pmatrix} \begin{pmatrix} \frac{1}{m} & 0 & 0 \\ 0 & \frac{1}{m} & 0 \\ 0 & 0 & \frac{1}{J} \end{pmatrix} \left(\begin{pmatrix} F_x \\ F_y \\ T \end{pmatrix} + \boldsymbol{F}_{\text{fault}} \right) \tag{3.2}$$

In this model, the inputs to the system are the resulting forces and torque F_x, F_y and T in the robot's frame of reference. $\boldsymbol{F}_{\text{fault}}$ is the faulty force that stems from failed actuators and, in anticipation of Eq. (3.7), $\boldsymbol{F}_{\text{fault}} \in \mathcal{U}_{\text{fault}}$. Compared to Figure 3.3, these are not the physical inputs $F_{i,j}$ with $i \in 1, \dots 4$ and $j \in 1, 2$ from the thrusters. The relationship

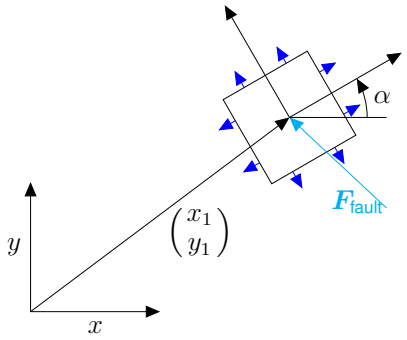


Figure 3.4: Schematic of the freeflyers.

between the resulting forces and the physical thrust inputs is simply a linear map

$$\boldsymbol{F} = \begin{pmatrix} F_x \\ F_y \\ T \end{pmatrix} = \boldsymbol{D} \begin{pmatrix} F_{1,1} \\ F_{1,2} \\ F_{2,1} \\ F_{2,2} \\ F_{3,1} \\ F_{3,2} \\ F_{4,1} \\ F_{4,2} \end{pmatrix} = \boldsymbol{D}\boldsymbol{F}_{i,j} \tag{3.3}$$

and can be given as

$$\begin{pmatrix} F_x \\ F_y \\ T \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 \\ d & -d & -d & d & d & -d & -d & d \end{pmatrix}}_{\boldsymbol{D}} \boldsymbol{F}_{i,j} \tag{3.4}$$

where $\boldsymbol{D}$ has full row rank.

Using the representation of the input as $\boldsymbol{F}_{i,j}$ directly for the control would be straightforward on one side: It contains the thruster forces explicitly and can therefore be readily applied without extra steps. Also, the set of inputs can be described easily as box constraints:

$$F_{i,j} \in \mathcal{U}_{i,j} = \begin{cases} \{0\} & \text{if thruster } i,j \text{ fails} \\ [0, F_{max}] & \text{if functional} \end{cases} \tag{3.5}$$

and the closed input set $\mathcal{U}_{\text{phys}}$ is the Cartesian product of all $\mathcal{U}_{i,j}$, i.e.

$$\mathcal{U}_{\text{phys}} = \mathcal{U}_{1,1} \times \mathcal{U}_{1,2} \times \ldots \times \mathcal{U}_{4,2} \tag{3.6}$$

Remark 5: Equivalently, it follows that $\boldsymbol{F}_{\text{fault}}$ is contained in

$$\boldsymbol{F}_{\text{fault}} \in \mathcal{U}_{\text{fault}} = \boldsymbol{D}\boldsymbol{F}_{i,j}^{\text{fault}} \tag{3.7}$$

where

$$\boldsymbol{F}_{i,j}^{\text{fault}} = \begin{pmatrix} F_{1,1}^{\text{fault}} & F_{1,2}^{\text{fault}} & \dots & F_{4,2}^{\text{fault}} \end{pmatrix}^T \tag{3.8}$$

with

$$F_{i,j}^{\text{fault}} \in \begin{cases} \{0\} & \text{if thruster } i,j \text{ is functional} \\ [0, F_{max}] & \text{if thruster has failed.} \end{cases} \tag{3.9}$$

The disadvantage of this representation, however, is that $\boldsymbol{F}_{i,j}$ is by Eq. (3.3) not uniquely expressible in terms of $\boldsymbol{F}$. In other words, there are multiple combinations of the eight thrust inputs that result in the same motion. The minimum energy solution would be the best combination as it prolongs the operating duration, but including this directly into a control law adds another level of complexity.

It is therefore common to decouple the problem of finding a controller using the resulting input ($\boldsymbol{F} \in \mathbb{R}^3$ in this case) from finding a physical realization through the actual actuators ($\boldsymbol{F}_{i,j} \in \mathbb{R}^8$). The latter step is called control allocation, see Section 1.4.4. Albeit the main problem considered in this work is the question of finding a suitable control law, a control allocator was developed so that the control law becomes applicable. The allocator finds the minimum-energy realization of the control input and can be found in Appendix A.

Remark 6: Using a Moore-Penrose pseudoinverse for control allocation might be the first intuition as Eq. (3.3) is a linear map. This is not sufficient, though, as it does not take the constraints into account. The analytical solution for the lowest-cost realization of $\boldsymbol{F}$ through $\boldsymbol{F}_{i,j}$ can be calculated as shown in Appendix A. Another procedure based on iteratively solving the Pseudoinverse until all constraints are satisfied is also possible and presented in [30].

This simplification on one side comes with a complication on the other side: While Eq. (3.3) describes how to map from $\boldsymbol{F}_{i,j}$ to $\boldsymbol{F}$ and Appendix A describes a mapping into the other direction, also the whole sets of inputs need to be mapped: The projection of the box-constrained input set of $\boldsymbol{F}_{i,j}$ does not yield a box-constraint for $\boldsymbol{F}$ any more, but, anticipating Section 3.1.3, rather a polytope. Also the influence of a failing actuator on the input set in $\mathbb{R}^3$ is not as immediately clear as for the higher-dimensional input. Nevertheless, this trade-off is considered advantageous given the analytical benefits and the simplified controller.

The question of how failing actuators influence the input set is addressed in the following section. This will make it possible to sort out between the scenarios that are easier and the ones that are harder to control. Finally, this will permit to give a complete statement of the control problem.

3.1.3 Effects of Actuator Failures on $\mathcal{U}_{\text{rem}}$

The subsequent section analyzes how actuator failures affect the maneuverability of the freeflyers. One of the problems that makes the control of systems under actuator failures difficult is that the combinatorics of the possible failures lead to a large number of

individual cases to be considered. While it might look at the first glance as if the value of $\boldsymbol{F}_{\text{fault}}$ was a mere difference of parameters, the coming sections show that the system behavior and consequently also the necessary control can differ greatly depending on which actuators fail. Therefore, it is in a first step of high importance to reduce the numbers of cases to the lowest amount possible.

For later reference, a fault scenario is defined as follows:

Definition 6 (Fault Scenario): Consider a freeflyer under the dynamics (3.2) and (3.3) with eight inputs. Each of the actuators $(1,1),(1,2),\dots(4,2)$ can be in a failed (as of Definition 5) or functioning state. A fault scenario describes the specific combination of failed and functioning actuators at a given time.

Finding redundant fault scenarios In a first step of analysis, redundant problem formulations can be found: Intuitively, it does not make a difference for the maneuverability if either $F_{1,1}$, $F_{3,1}$, $F_{2,2}$ or $F_{4,2}$ are affected (see Figure 3.3 for clarity) - simply by rotating the coordinate system in 90° steps, one case can be modeled by one another. Additionally, if the other four thrusters are affected individually, then it is the same pattern as before just with changed signs of the resulting forces and torque.

Conceptually, if any single actuator fails, all fault scenarios therefore have the same effect on the fault induced on the system and the remaining degrees of freedom. These scenarios are therefore redundant, in the sense of the following definition:

Definition 7 (Redundant Fault Scenarios): A set of fault scenarios is called redundant if the set $\mathcal{U}_{\text{phys}}$ of remaining available inputs for each scenario can be transformed into $\mathcal{U}_{\text{phys}}$ of the others purely by rotation and reflection.

The problem of finding redundant fault scenarios automatically and for any number of errors is a geometrical one: The redundancies appear through a geometrically "similar" (i.e. rotated and reflected) arrangement of the failed and functioning actuators. It can thus also be solved through with geometrical methods, namely an approach that relies on finding congruent polytopes. Two polytopes (or more general, subsets) are said to be congruent if one is the image of the other via an isometry [66], i.e. the sets are rotationally and reflectively invariant.

Proposition 1 (Redundancy of fault scenarios): When a freeflyer experiences multiple actuator failures (as of Definition 5), redundant scenarios (ref. Definition 7) may occur. These redundancies arise from the same *relative* positioning of the failed actuators, where two scenarios differ only by rotation or reflection. To identify such redundant scenarios, polygons can be defined whose corners lie at the geometrical position of the failing actuators. If two polygons are congruent, then their corresponding fault scenarios are similar and can be treated with the same control approach.

No. actuator failures	No. of overall scenarios	No. of unique scenarios
1	8	1
2	28	6
3	56	7
4	70	13
5	56	7
6	28	6
7	8	1

Table 3.1: Count of overall combinations of actuator failures per number of failures and number of rotationally and reflectively unique cases.

Proof As the similarity is a purely geometrical feature based on the position of the thrusters, any set of failing thrusters can be interpreted as a polytope in V-representation. Therefore, algorithms for checking congruence find similar scenarios. As the difference between similar scenarios is just one in reflection and orientation, a different reference system can be introduced in which one similar scenario looks exactly as the other one. This, however does not change the control task so the same method can be applied. ∎

Methods for finding congruent polytopes are well-established and can for example be found in [66]. With this approach, the number of cases that need to be considered can be greatly reduced: The overall number of scenarios of n failures is $\frac{8!}{n!(8-n)!}$ using the combinatorics of the binomial distribution. Table 3.1 shows how many rotationally and reflectively unique scenarios can be found: The number of overall scenarios from 0 to 8 faults is 256 while there are only 43 unique scenarios.

In the rest of the work, only unique failure scenarios will be regarded. Looking at these individually allows to generalize the problems further into four fault classes. For them, some properties of the input sets can be derived.

Different fault cases and their implications As from Eq. (3.3) follows, the map between the thruster forces and the resulting forces and torque is linear. Also, the constraints on the thruster forces are given as box constraints (see Eq. (3.5)). Due to linearity, the resulting set $\mathcal{U}_{\text{rem}}$ will be a polytope as well [67, p. 16 f.]. It can be calculated by transforming the set into V-representation, transforming each vertex and then finding the H-representation again.

As defined above, $\mathcal{U}_{\text{phys}}$ is the box-constrained set of physical inputs in $\mathbb{R}^8$. Additionally, two more sets are defined: $\mathcal{U}_{\text{rem}}$ is the remaining set of the manipulable inputs after one or multiple failures occurred[1].

$$\mathcal{U}_{\text{rem}} = \boldsymbol{D}\mathcal{U}_{\text{phys}} \tag{3.10}$$

[1]The non-manipulable, i.e. failed actuators form the faulty input force $\boldsymbol{F}_{\text{fault}}$.

The resulting set $\mathcal{U}_{\text{res}}$ represents the possible resulting thrust forces under control *and* actuator failures.

$$\mathcal{U}_{\text{res}} = \mathcal{U}_{\text{rem}} \oplus \boldsymbol{F}_{\text{fault}}. \tag{3.11}$$

Note that from $\mathcal{U}_{\text{phys}}$ being closed, it follows directly that $\mathcal{U}_{\text{rem}}$ and $\mathcal{U}_{\text{res}}$ are also closed.

$\mathcal{U}_{\text{res}}$ has a great analytical value as the acceleration of the system depends linearly on this resulting force (ref. Eq. (3.2)). In other words, $\mathcal{U}_{\text{res}}$ determines the set of possible accelerations and therefore the possible movements of the robot.

In the following, four different fault cases will be distinguished:

C1 Zero is contained in the interior of $\mathcal{U}_{\text{res}}$.

C2 Zero is on the boundary of $\mathcal{U}_{\text{res}}$.

C3 Zero is not contained in $\mathcal{U}_{\text{res}}$ and at least one vector $\begin{pmatrix} f_x & f_y & 0 \end{pmatrix}^T$ is contained in the interior of $\mathcal{U}_{\text{res}}$ where f_x, f_y are arbitrary.

C4 Neither zero nor an arbitrary vector $\begin{pmatrix} f_x & f_y & 0 \end{pmatrix}^T$ are contained by $\mathcal{U}_{\text{res}}$ in its interior.

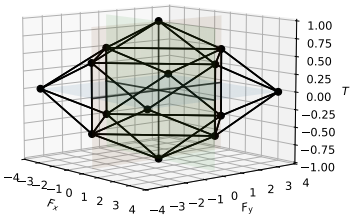
> **Remark 7:** Without in-depth theoretical justification, $\begin{pmatrix} f_x & f_y & 0 \end{pmatrix}^T \in \text{Int}(\mathcal{U}_{\text{res}})$ implies that the orientation is still controllable even if the position is not.

In order to build intuition, an example for each of the cases is given and later a general analysis is presented. $\mathcal{U}_{\text{res}}$ for the case without any faults is shown in Figure 3.5a and it is clear that this is case C1. If thruster $(1,1)$ fails with an uncontrollable thrust of $F_{\max}$, still C1 emerges as shown in Figure 3.5b. Case C2 can for example occur if thrusters $(1,1)$ and $(2,1)$ get stuck at $F_{\max}$ and case C3 emerges if additionally $(1,2)$ gets stuck at zero, see Figures 3.5c and 3.5d. If thrusters $(1,1)$, $(2,2)$, $(3,1)$ and $(4,2)$ all get stuck at $F_{\max}$, case C4 arises.

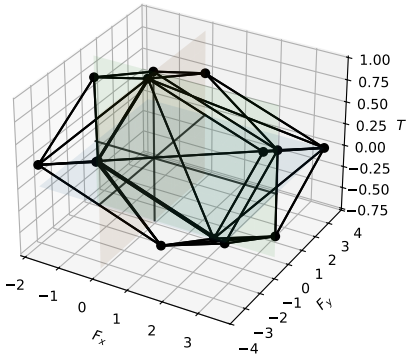
In the author's view, these plots clarify what the definitions of the different fault cases already contain: Case C1 is, as demonstrated, almost the same case as if the system has no actuator failures. The system can accelerate into every direction and is hence holonomic and small-time locally controllable. Conceptually, one could therefore think of a controller that compensates for the faulty force and then applies the same strategy as for the nominal, fault-free system with adjusted parameters. This case is thus not of particular interest for this work.

> **Remark 8:** In both cases, the system has full local controllability. Thus, the infinite-horizon LQR controller can be used to ensure stabilization locally by deriving the terminal cost and set from it.

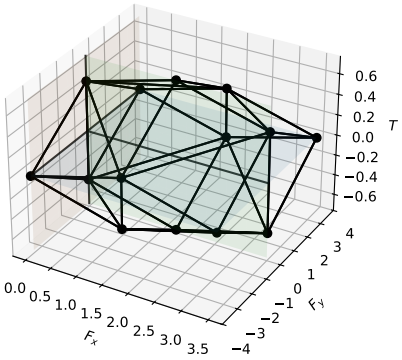
In case C2, however, the system becomes nonholonomic: For the case given in Figure 3.5c, the system cannot accelerate any more into the local negative x-direction. A compensation



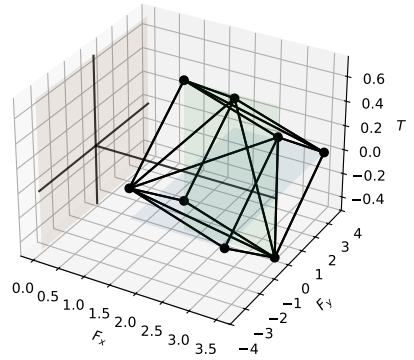
(a) No actuator failures: Case C1.



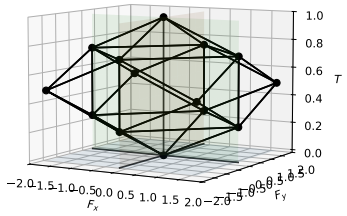
(b) $F_{1,1} = F_{\max}$: Case C1.



(c) $F_{1,1} = F_{2,1} = F_{\max}$: Case C2.



(d) $F_{1,1} = F_{2,1} = F_{\max}, F_{1,2} = 0$: Case C3.



(e) $F_{1,1} = F_{2,2} = F_{3,1} = F_{4,2} = F_{\max}$: Case C4.

Figure 3.5: Examples for $\mathcal{U}_{\text{res}}$ if different actuators fail. Observe where the origin falls to with respect to the interior of the polytopes.

of the uncontrollable input is therefore not possible any more and a new control design needs to be found.

> **Remark 9:** It should however be noted that according to the system equation Eq. (3.2), the system can accelerate into the *global* negative x-direction: As the input forces are multiplied with the nonlinear rotation matrix $\mathfrak{R}$, the orientation of

the system - not the actuator faults themselves - determines in which direction it can and cannot accelerate globally.

It may also be noted that the thrusters opposite to the failing ones can be chosen to cancel out the failing ones. This would be equivalent to a system that is only propelled by the thrusters in the y-direction. Analysis of the resulting system equations show that this is almost the same description as a unicycle, to be more precise the same as a unicycle on a ground without friction.

In contrast to case C2, C3 is also nonholonomic but in this case it is not possible any more to have zero acceleration into the x-direction. This means that point stabilization is not only from a practical, but even from a theoretical point infeasible as no compensation can be found s.t. $f(x = 0, u = \text{const.}) = 0$. As also observed in Remark 9, the acceleration in the global coordinate system however also depends on the orientation, expressed in Eq. (3.2) through $\mathfrak{R}$. And additionally, the orientation can still be controlled: The subsystem

$$\begin{pmatrix} \dot{\alpha} \\ \dot{\omega} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ \omega \end{pmatrix} + \begin{pmatrix} 0 \\ \frac{1}{J} \end{pmatrix} T \tag{3.12}$$

is just a double integrator with input T. This is why there is a distinction is to be made between cases C3 and C4: In C3, the orientation can still be used to "steer" the system into a certain direction while in C4, this is not possible any more.

Remark 10: One interpretation of these results is the following: The actual underlying difficulty of the problem is not, that the system is disturbed by a constant force. The problem is instead that the magnitude of the disturbing force lies within the range of the other actuators - with the implication that while it is possible to cancel it out, no control authority is left in that case.

Remark 11: This also means that this work cannot apply common strategies for robust control as the bare mitigation the influence of the error is not possible. Conceptually, it is therefore expedient to think of all the different error scenarios as a class of systems with common characteristics and different manifestations. The task is to find a control scheme that can control all systems in this class.

Relationship between actuator failures and fault cases The implications of the different fault cases have been investigated in the previous paragraph by means of examples. The interpretation for each case stands however in general. It is furthermore of interest to find when each case can appear based on the number of actuator failures. Some conditions can be given in the following lemma:

Lemma 1: The relationship of the number of failing actuators (as of Definition 5) and the possible occurrence of fault cases C1-C4 for the freeflyers is as follows:

- If there is no or one actuator failing, C1 appears always.

No. actuator faults	C1	C2	C3	C4
1	always	-	-	-
2	✓	faults at 0 or max	-	-
3	✓	✓	✓	-
4	✓	✓	✓	faults at 0 or max
5+	✓	✓	✓	✓

Table 3.2: Overview of the fault cases that can occur for different numbers of actuator failures. ✓ means that the fault can occur for certain combinations of faults, hyphen means that it can never. The textual statements are necessary but not always sufficient conditions that the case emerges.

- If there are two actuators failing, C1 or C2 appear always.
- If there are three actuators failing, cases C1-C3 can appear.
- Only if there are four or more actuators failing, all cases including C4 can appear.

A slightly more detailed overview is displayed in Table 3.1.

Proof sketch For the sake of brevity, the proof is not written down explicitly. It follows directly from two properties: First of all, the absolute value of $\boldsymbol{D}$ for each row is the same (if not zero), only the sign can be changed. Furthermore, the possible thrust for each thruster is in $[0, F_{\max}]$. Consider only the first row of $\boldsymbol{D}$: The set of possible values for the resulting force in x-direction is then given as

$$\mathcal{U}_{1,1} \oplus (-\mathcal{U}_{1,2}) \oplus \mathcal{U}_{2,1} \oplus (-\mathcal{U}_{2,2}) \oplus \boldsymbol{F}_{\text{fault}}[1] \tag{3.13}$$

with the definition of $\mathcal{U}_{i,j}$ from Eq. (3.5) and $\boldsymbol{F}_{\text{fault}}[1]$ being the first element of $\boldsymbol{F}_{\text{fault}}$.

Playing through the different (unique) scenarios of broken thrusters leads to the resulting set containing zero inside, on the boundary or not containing it. Finally using the equivalent argument for the other rows yields the result of Lemma 1. ∎

3.2 Problem Statement

The preceding results from this chapter give the means for the full problem statement: In Section 3.1, a mathematical model of the freeflyers is provided. The actuator bounds are derived in Section 3.1.3, where also the four fault cases are defined that need to be considered. Further analysis shows that case C1 is easy to handle with a slight adaption of the control for the nominal case. Cases C2 and C3 are shown to be not controllable with the existing method due to the nonholonomicity and partial loss of controllability. Also case C4 is shown to be not controllable with the existing method, but also to be less promising to be possibly controllable at all.

Based on this, the problem statement follows:

Definition 8 (Problem statement): Consider a freeflyer with the dynamics (3.2) under actuator failures. Assume one or multiple actuator failures as of Definition 5 are present and the fault scenario (ref. Definition 6) is of the fault cases C2 and C3.

Let the freeflyer be subject to the feasible state constraints $x \in \mathcal{X}$ for all $x(t), t \geq 0$ resp. $x_k, k \in \mathbb{N}_0^+$. Additionally, let the input constraints under actuator failures be $u \in \mathcal{U}_{\text{rem}}$ for all $u(t), t \geq 0$ resp. $u_k, k \in \mathbb{N}_0^+$ with $\mathcal{U}_{\text{rem}}$ as given in Eq. (3.11).

Assume a trajectory χ is continuously resp. discretely dynamically feasible as of Definition 2. Assume further that it satisfies the state constraint $\mathcal{X}$ within an neighborhood $\mathcal{N} = \{x \in \mathbb{R}^{n_x} | \|x - \chi\| \leq \rho\} \subseteq \mathcal{X}$ for all t/k in the trajectory's time domain where $\rho > 0$ is constant.

The control problem consists of bringing the freeflyer to $\mathcal{N}$ and keeping it inside of it while satisfying the constraints, i.e. the tracking error e converges to $e \leq \rho$ under satisfaction of $x \in \mathcal{X}$ and $u \in \mathcal{U}_{\text{rem}}$.

According to Lemma 1, this means that if the problem is solved, the freeflyers can be guaranteed to be partially controlled for at least three actuator failures. As C1- C3 also appear for some of the scenarios with four or more actuator failures, this is however not exhaustive and there are more recoverable combinations.

Remark 12: It is in most cases not desirable to continue operating the system even under actuator failures. A controller satisfying the given definition is however capable of avoiding crashes on a short horizon and to autonomously move to a place where necessary repairs can be undertaken and help is available.

3.3 System Analysis

In the previous section, the problem statement has been given. The purpose of the concluding part of the chapter is to provide some more analysis in preparation of the later proposed controllers. Different potential control strategies are discussed and assessed regarding their performance as well as practicability.

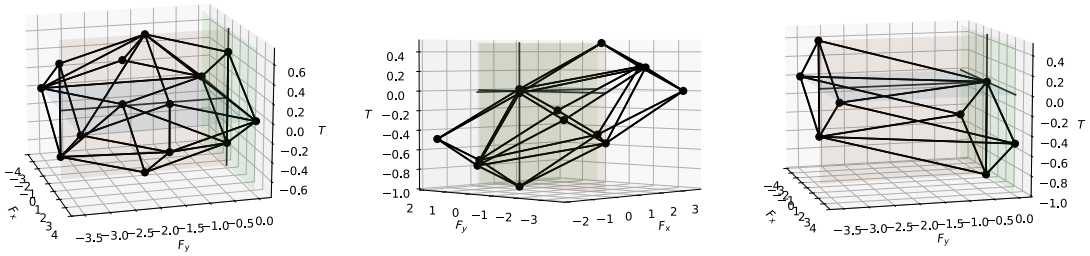
The first direct implication that follows from the nonholonomicity is that nonholonomic systems can never be stabilized by a linear feedback law [68]. Linearization of the system and using a local control law is thus impossible. Additionally, Brockett's Theorem states that nonholonomic systems cannot even be point-stabilized using smooth (nonlinear) time-invariant feedback [45]. Hence, classical feedback-linearization is also not possible in this case.

Brockett's Theorem, however, leaves some other possibilities for stabilization [69]: Non-smooth and time-varying feedback. Leaving general time-varying feedback out of the scope for now, another well-established method would be Sliding Mode Control which gives a non-smooth control law. Still, this control scheme requires $\mathbf{0} \in \text{Int}(\mathcal{U}_{\text{res}})$, see [70, p. 563f.] where $\mathbf{0} \in U$ for an *open* set U is needed. As of the definition of C2, this is not fulfilled with the current set of actuators.

Sliding Mode Control has indeed been applied to nonholonomic systems. In order to make this work, the aforementioned condition is necessary to be satisfied which is possible only if certain actuators are taken out of the control. This means that as in the control schemes discussed in the introduction, not only a broken thruster has to be taken out of control, but also its counterpart on the other side. Thus, the controller suffers from conservativeness, as not the whole input can be used.

Additionally, further results even indicate that the system can become small-time locally uncontrollable for certain combinations of failures [30]. In this case it is not possible to use Sliding Mode Control any more as the assumption that the system is controllable in the region of the Sliding Mode (which contains zero) is violated.

Accordingly, a different non-standard, nonlinear and possibly time-varying feedback law would need to be found. In fact, the requirements are even tighter: It has already been observed that case C2 means that nonholonomic constraints appear. So far, the actual form of the constraints has however not been regarded yet. And as you can see from Figure 3.6, the constraints can be quite different for different cases: Figure 3.6a only has nonholonomic constraints on F_y, while the nonholonomic constraint in Figure 3.6b holds for a plane that contains the origin and involves F_x, F_y and T. For Figure 3.6c, there are even three planes meeting at the origin and therefore posing nonholonomic constraints. The statement from the beginning of this paragraph therefore needs to be corrected: In fact, *multiple* non-standard, nonlinear and possibly time-varying feedback laws would need to be found.



(a) $F_{3,1} = F_{\max}$, $F_{3,1} = F_{\max}$ (b) $F_{3,2} = F_{\max}$, $F_{1,1} = 0$ (c) $F_{3,2} = F_{\max}$, $F_{3,1} = 0$

Figure 3.6: $\mathcal{U}_{\text{res}}$ for different combinations of 2 failing actuators of class C2.

While it might be possible to find different controllers so that every case that can occur is satisfied, this would turn out to be a very complicated and cumbersome work: A search on Google Scholar for "unicycle control" gives around 22 900 results that date back from the 1980s until today. Only for this single and also very simple nonholonomic system, massive research efforts have been made. While the problem can be seen as solved today, this shows how difficult it can be to derive a working nonholonomic controller even for one single system, let alone multiple ones.

The significant challenges that come with controlling the system in a failure case of C2 become apparent by this analysis. This becomes even more evident if systems of C3 are supposed to be controlled: As aforementioned, $\mathbf{0} \in \mathcal{U}_{\text{res}}$ is required so that the desirable steady states and linear trajectories are feasible. By definition, this requirement is not satisfied and trade-offs are necessary to guarantee at least some level of control.

This is the reason why the problem is approached from a slightly different angle in this work: Instead of controlling all system states, the attitude is disregarded and it is considered sufficient if the position can be kept in a certain region around the zero error. It turns out that in this case a new system formulation can be found which does not suffer from the discussed problems, and both C2 and C3 can be treated.

The proposed solution is called micro-orbiting and has to the best of the author's knowledge not been proposed earlier. It is based on bringing the system into a small-scale orbit around a defined point, and using the orbit dynamics to cancel out the partially compensated error.

> **Remark 13:** The question might come to mind why an explicit controller is needed at all: MPC can, ultimately, take any constraints into account and find the optimal solution. The problem is that, as described in Section 2.3, the control horizon is limited and in order to provide stability, a terminal condition is necessary that allows to conclude that. Requiring the error to be zero at the end of the horizon would theoretically suffice, but poses a practical problem regarding the feasibility if the horizon is too small. Otherwise, the knowledge of a valid terminal condition is equivalent to the knowledge of local stability or a locally stabilizing control law. Therefore, even if the actual used controller is of MPC type, a terminal controller is sought that ensures stability for all time steps beyond the prediction horizon.

Chapter 4

Gaining Holonomicity through Micro-Orbiting

The previous chapter demonstrated the challenges in controlling the system as initially formulated. This chapter proposes a strategy to circumvent these issues by modifying the control objective. Despite this modification, the problem statement outlined in Definition 8 remains satisfied. While the strategy does not yet incorporate specific controllers (which will be presented in Chapter 5), it yields a controllable model, thus bridging the gap between modeling and control.

The basic idea is to bring the system into a small-scale orbit, called micro-orbit, around a defined point. By exploiting the orbit dynamics, the uncontrollable error can be partially compensated and a holonomic system can be derived.

4.1 Planar Circular Movements

To introduce the concept of micro-orbiting, the kinematics of a body in planar movement are presented. This serves to show which accelerations appear when a body rotates around a center and which forces are necessary to maintain the motion.

The following equations are adapted from [71] and a drawing of the system is shown in Figure 4.1: Introduce two points A and B with positions $\boldsymbol{v}_A$ and $\boldsymbol{v}_B$. Let B be the center of a rigid body and A, not lying on the body, have a constant distance r to B. Introduce the local time-dependent coordinate system with unit vectors $\boldsymbol{e}_r$ and $\boldsymbol{e}_\varphi$ such that the connection vector between A and B becomes $r\boldsymbol{e}_r$. Let ω be the angular velocity of the rigid body and m its mass. Then it holds

$$\begin{aligned} \boldsymbol{v}_B &= \boldsymbol{v}_A + r\boldsymbol{e}_r \\ \dot{\boldsymbol{v}}_B &= \dot{\boldsymbol{v}}_A + \omega r\boldsymbol{e}_\varphi \\ \ddot{\boldsymbol{v}}_B &= \ddot{\boldsymbol{v}}_A + \dot{\omega} r\boldsymbol{e}_\varphi - \omega^2 r\boldsymbol{e}_r. \end{aligned} \tag{4.1}$$

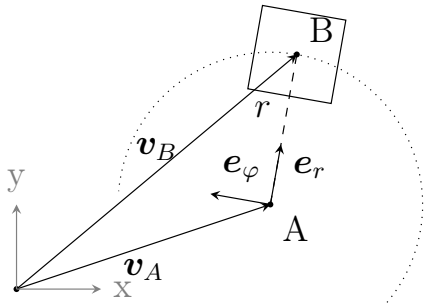


Figure 4.1: Drawing of a rigid body in planar circular motion as described in Eq. (4.1).

This is so far only a kinematic description of a motion, the motion will however not arise by itself. In order to realize it for a physical object, an external force needs to be present to keep B on the orbit around A.

Restrict the consideration now to a rotation of B around A with constant angular velocity ω_0: In this case, it is clear by Newton's law that the necessary force to keep the body on the orbit around A is $-m\omega_0^2 r\boldsymbol{e}_r$. This is a constant force that always directs towards A, i.e. the center of the circular path that B is following.

> **Remark 14:** Be careful to not confuse kinematics and dynamics here: The kinematics presented in Eq. (4.1) show that the accelerations $\dot{\omega} r\boldsymbol{e}_\varphi$ and $-\omega^2 r\boldsymbol{e}_r$ need to be present so that the described motion is possible. If there is no physical constraint (i.e. A is on the body), these accelerations would *not* occur without influence from the outside - according to Newton's law, the body would keep its inertia and follow a straight motion into the current direction. The accelerations must instead be applied *externally* by the forces acting on the body.

Eq. (4.1) only considers the position, not the orientation of the body: The rotational inertia will - once again following Newton's law - stay constant without external torques applied. This means the following: Consider both the angular velocity of the body around itself and the angular velocity of the body around the center A are ω_0. Then, orientation of the body towards A will not change as long as the two motions are sustained. This happens as the two changes in relative orientation cancel each other out.

The implication is that if (1) both angular velocities are the same and constant at some point, (2) no external torques are applied and (3) the necessary force $-m\omega_0^2 r\boldsymbol{e}_r$ acts on the body, then this force will be constant in the local coordinate system of the body. Also, the body will stay on the circular trajectory. In other words, the body experiences a constant force relative to its own coordinate system, but maintains the rotation around A.

The observation can be extended to orbits around a moving center A: As of Eq. (4.1), the inertial accelerations and the accelerations of the center can be summed up. Accordingly, the body only needs to additionally experience the same forces and accelerations as if it was directly moving along the trajectory. To describe the resulting movement in intuitive words, the path of the body then becomes a spiral around a trajectory.

To conclude, there is only one locally constant force and zero torque necessary to keep a rigid body on a suitable orbit once it has entered it. If the body should orbit around a point moving along a trajectory, simply another force should be added which realizes the accelerations of the trajectory without the orbit.

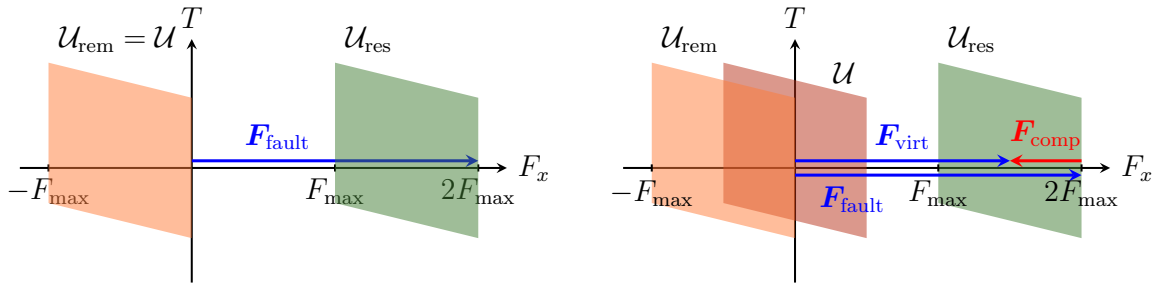
> **Remark 15:** The dynamics, for the case of a static center, are exactly the same as the ones for satellites on a big scale: In their case, the force acting on them is gravity while the radius is determined by their altitude. It is as well similar to what happens to the body of a passenger on a swing carousel.

4.2 Micro-Orbiting

As described in Section 3.3, the constant faulty force makes it impossible to keep a robot with fault case C3 at a defined setpoint. In the light of the foregoing section, something else is however possible: As the uncontrollable faulty force is constant, it can keep the system on a suitable orbit around the setpoint. This is the main idea of micro-orbiting. Additionally, with a partial compensation of the uncontrollable faulty force, the problem of nonholonomicity can be circumvented that appears for both C2 and C3. These thoughts will be extended in the coming section.

The thought that the uncontrollable force can keep the system on an orbit around a point is rather straightforward. As it is constant and the orientation and angular velocity can still be controlled in cases C2 and C3, a suitable orbit can be found and maintained. But while this enables the system even for C3 to stay in a certain region - namely a circle with radius r - around a setpoint, nonholonomicity can still occur:

Consider for example the case shown in Figure 4.2[1] where $F_{1,1} = F_{2,1} = F_{\max}$ and $F_{1,2} = 0$. In this case, $\boldsymbol{F}_{\text{fault}} = \begin{pmatrix} 2F_{\max} & 0 & 0 \end{pmatrix}^T$ and $F_x \in [F_{\max}, 2F_{\max}]$ for all $F_x \in \mathcal{U}_{\text{res}}$. If a suitable micro-orbit is found so that $\boldsymbol{F}_{\text{fault}}$ is completely compensated by the rotational accelerations, $\mathcal{U}_{\text{rem}}$ can be calculated according to Eq. (3.11). This means that $F_x \in [-F_{\max}, 0]$ for all $F_x \in \mathcal{U}_{\text{rem}}$, i.e. the controllable inputs are still nonholonomic.



(a) Orbiting with $\boldsymbol{F}_{\text{fault}}$ (b) Orbiting with $\boldsymbol{F}_{\text{virt}}$

Figure 4.2: Relationship between the different input sets and vectors for $F_{1,1} = F_{2,1} = F_{\max}$ and $F_{1,2} = 0$. Displayed are projections on the F_x-T-plane.

[1] an extension of the example in Section 3.1.3, see Figure 3.5d

It turns out that this can be solved easily by introducing a partial compensation of the uncontrollable input. In terms of the example, consider what would happen if the amount of force directed to orbiting would not be $\begin{pmatrix}2F_{\max} & 0 & 0\end{pmatrix}^T$, but $\begin{pmatrix}1.5F_{\max} & 0 & 0\end{pmatrix}^T$: Then, in an adaption to Eq. (3.11), $F_x \in [-0.5F_{\max}, 0.5F_{\max}]$ for F_x in the resulting input set $\mathcal{U}$. To realize this, $\boldsymbol{F}_{\text{fault}}$ simply needs to be compensated by $0.5F_{\max}$.

> **Remark 16:** Keep in mind that additions in $\mathcal{U}_{\text{res}}$ or $\mathcal{U}_{\text{rem}}$ do not directly correspond to changes in actuator thrusts of one distinct actuator. Instead, if a resulting force is within these sets, there is at least one way (but maybe multiple) of realizing this resulting force.
>
> In terms of the example, it would not suffice to only compensate using thruster $(2, 2)$ - in that case, the robot would experience an unwanted torque. Consequently, the remaining actuators must counteract this torque, making the physical realization of the compensation less apparent. This underlines the advantages of using a control allocator as mentioned before and described in Appendix A.

This thought can be generalized and systematized:

Definition 9 (Virtual faults): Consider a freeflyer under actuator failures (as of Definition 5): The virtual fault $\boldsymbol{F}_{\text{virt}} \in \mathcal{U}_{\text{res}}$ is the *remaining* uncontrollable thrust after a partial compensation $\boldsymbol{F}_{\text{comp}} \in \mathcal{U}_{\text{rem}}$ of the *actual* (uncontrollable) faulty thrust, i.e.

$$\boldsymbol{F}_{\text{virt}} = \boldsymbol{F}_{\text{fault}} - \boldsymbol{F}_{\text{comp}}. \tag{4.2}$$

The remaining set of available inputs is then

$$\mathcal{U} = \mathcal{U}_{\text{res}} \ominus \boldsymbol{F}_{\text{virt}}. \tag{4.3}$$

Proposition 2 (Micro-orbiting for fault-mitigation): Consider a freeflyer under actuator failures causing case C2 or C3 and a compensation force $\boldsymbol{F}_{\text{comp}}$ such that the virtual fault $\boldsymbol{F}_{\text{virt}} \in \text{Int}(\mathcal{U}_{\text{res}})$. Then, the input constraints of the system can be made holonomic by redefining the control objective: Instead of controlling the position of the freeflyer itself, it is brought into a small-scale orbit, called micro-orbit, around a setpoint or trajectory. By these means, the system can be kept within a region around the desired point even though the point itself may not be recursively feasible.

Proof Start by showing that the virtual fault exists and the sets have the desired properties: As $\mathcal{U}_{\text{res}}$ is the set of resulting overall forces which includes both the controllable control input as well as the uncontrollable fault input $\boldsymbol{F}_{\text{fault}}$, the resulting force needs to be within $\mathcal{U}_{\text{res}}$. By Definition 9 and Eq. (3.11), $\boldsymbol{F}_{\text{virt}}$ is hence always feasible and within the bounds of the definition freely selectable. If $\boldsymbol{F}_{\text{comp}}$ is moreover chosen s.t. $\boldsymbol{F}_{\text{virt}} \in \text{Int}(\mathcal{U}_{\text{res}})$, then $\mathbf{0} \in \text{Int}(\mathcal{U})$ by Definition 9.

Assume the system is operated on an orbit that can be kept up by $\boldsymbol{F}_{\text{virt}}$. Then, as seen from Eq. (4.1), the accelerations and $\boldsymbol{F}_{\text{virt}}$ cancel out. The remaining controllable input therefore lies by Eq. (4.3) within $\mathcal{U}$. ∎

It is important to note that Definition 8 only requires the system to stay within a region around a point, i.e. the error does not need to be zero. If the system stays on the orbit, the error is therefore constant and equal to r and the control objective is fulfilled.

Potentially, if only staying in a region around the setpoint is required, any feasible closed trajectory could be used - from tracking "8"-shaped paths over squares with rounded corners to completely free (feasible) forms. A circle is just the easiest-to-describe trajectory as it is not time-dependent. Moreover, the faulty force is constant so the circle is viable without additional control effort and energy. And as a third advantage, micro-orbiting allows to avoid the nonholonomicity, therefore solving two problems at once.

4.3 Practical Considerations

The current section has shown that there are combinations of forces and orbit parameters that keep the robot on the orbit. So far this has been restrained to "suitable" or "feasible" orbits and not been specified further. Also it has not been discussed according to which criteria $\boldsymbol{F}_{\text{virt}}$ should be selected.

To start with, the latter question can be answered: As explained in the foregoing section, the introduction of the virtual faults was reasoned by its ability to bring $\mathbf{0}$ into the interior of $\mathcal{U}$. This increases the control authority because, as explained, the orbiting dynamics and the virtual fault cancel out and only $\mathcal{U}$ remains.

$\mathcal{U}$ is therefore the control that can be used for trajectory following. As such, it imposes also a feasibility constraint: Only trajectories that can be realized with the inputs in $\mathcal{U}$ are feasible.

Consequently, it is desirable to increase the control authority as much as possible. To put it into more concrete terms, $\boldsymbol{F}_{\text{virt}}$ should have the maximal distance to the boundary of $\mathcal{U}_{\text{res}}$[2] which would put it as a first guess at the Chebychev center of $\mathcal{U}_{\text{res}}$.

Additionally, however, it was pointed out that no external torque should act on the system in order to maintain the orbit as described. Therefore, the third value of $\boldsymbol{F}_{\text{virt}}$ needs to be fixed to 0 and only the other two values can be optimized. Finding $\boldsymbol{F}_{\text{virt}}$ becomes thus a convex optimization problem of the form

$$\begin{aligned} \underset{\boldsymbol{F}_{\text{virt}},\,\xi}{\text{maximize}} \quad & \xi \\ \text{subject to} \quad & \boldsymbol{F}_{\text{virt}} + \xi u \in \mathcal{U}_{\text{res}}, \forall \|u\| \leq 1, \\ & \boldsymbol{F}_{\text{virt}}[3] = 0 \end{aligned} \tag{4.4}$$

which is the problem of finding the largest ball contained in the polytope $\mathcal{U}_{\text{res}}$. The solution can be easily found using numerical solvers, for reference see also [42, p. 417 f.].

[2] Wherefore $\mathbf{0}$ has the maximal distance to the boundary of $\mathcal{U}$ by Eq. (4.3)

> **Remark 17:** One interpretation of the orbiting with the virtual fault would be as a method to compensate a "desired amount of fault" such that a maximal maneuverability is achieved.

For the orbit parameters, Section 4.1 implicitly already contains the answer: In the discussion, it has been shown that

$$\boldsymbol{F}_{\text{virt}} \stackrel{!}{=} -m\omega_0^2 r \boldsymbol{e}_r \tag{4.5}$$

respectively

$$\|\boldsymbol{F}_{\text{virt}}\| \stackrel{!}{=} m\omega_0^2 r. \tag{4.6}$$

This leaves two degrees of freedom: One of them is already fixed by the choice of $\boldsymbol{F}_{\text{virt}}$. Accordingly, ω_0 can be selected fixing r or the other way around.

The parameters ω_0 and r can be treated as tuning variables. Selecting these parameters involves a trade-off: A smaller orbital radius better satisfies the original control objective, but reducing the radius necessitates an increase in ω_0. Regular trajectories typically exhibit relatively low values of ω_0 wherefore increasing it means that the system starts increasingly distant from the desired setpoint. This can lead to a transient response that violates state constraints or the possible infeasibility of the problem.

4.4 Orbit Center Dynamics

Until now, the micro-orbiting was described by the position of the freeflyer that executes two motions in superposition, namely the movement along the trajectory and the orbit. Even though this is a good perspective for describing the idea, it turns out that a different model is more useful for the mathematical treatment: Instead of the position of the robot, the position of the orbit center can be described. In this case, the center means where the center should be with reference to the *robot*, not with reference to the trajectory.

Without loss of generality, the local coordinate system can be chosen at the center of mass of the robot such that the virtual fault is aligned with the positive local y-axis. The position of the desired center follows accordingly as $r\boldsymbol{e}_{y'}$. In this formulation, the inputs F_x and F_y are not necessarily aligned with the axes any more, but this can simply be solved by a change of variables through a rotation of the input. The rotation of all necessary input sets follows accordingly.

It is important to note that this change of variables introduces a distinction between the equations presented up to this point and those that follow. While the symbols remain the same, they now carry new definitions. The rest of this work will use the notation introduced here. A schematic of the model is shown in Figure 4.3.

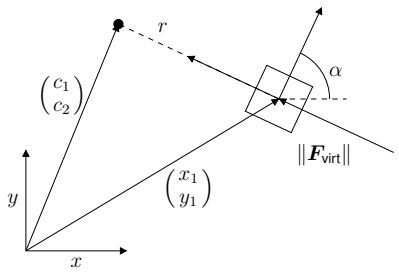


Figure 4.3: Schematic of the center point dynamics under micro-orbiting.

The dynamics of the robot can then be given as

$$f(x) = \begin{pmatrix} \dot{x}_1 \\ \dot{y}_1 \\ \dot{\alpha} \\ \dot{x}_2 \\ \dot{y}_2 \\ \dot{\omega} \end{pmatrix} = \begin{pmatrix} x_2 \\ y_2 \\ \omega \\ \boldsymbol{R}\begin{pmatrix} F_x \\ F_y + \|\boldsymbol{F}_{\text{virt}}\| \end{pmatrix} \frac{1}{m} \\ T\frac{1}{J} \end{pmatrix} \tag{4.7}$$

where

$$\boldsymbol{R} = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix}. \tag{4.8}$$

The dynamics of the center can be derived by forward-calculation using the relationship

$$\begin{pmatrix} c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} x_1 - \cos(\frac{\pi}{2} - \alpha)r \\ y_1 + \sin(\frac{\pi}{2} - \alpha)r \end{pmatrix} = \begin{pmatrix} x_1 - r\sin(\alpha) \\ y_1 + r\cos(\alpha) \end{pmatrix} \tag{4.9}$$

with its derivatives

$$\begin{pmatrix} \dot{c}_1 \\ \dot{c}_2 \end{pmatrix} = \begin{pmatrix} \dot{x}_1 - r[\cos(\alpha)\omega] \\ \dot{y}_1 - r[\sin(\alpha)\omega] \end{pmatrix} = \begin{pmatrix} x_2 - \omega r\cos\alpha \\ y_2 - \omega r\sin\alpha \end{pmatrix} =: \begin{pmatrix} c_3 \\ c_4 \end{pmatrix} \tag{4.10}$$

and

$$\begin{aligned} \begin{pmatrix} \ddot{c}_1 \\ \ddot{c}_2 \end{pmatrix} = \begin{pmatrix} \dot{c}_3 \\ \dot{c}_4 \end{pmatrix} &= \begin{pmatrix} \dot{x}_2 - \dot{\omega}r\cos\alpha + \omega^2 r\sin\alpha \\ \dot{y}_2 - \dot{\omega}r\sin\alpha - \omega^2 r\cos\alpha \end{pmatrix} \\ &= \boldsymbol{R}\begin{pmatrix} F_x \\ F_y + \|\boldsymbol{F}_{\text{virt}}\| \end{pmatrix}\frac{1}{m} + \begin{pmatrix} -\frac{T}{J}r\cos\alpha + \omega^2 r\sin\alpha \\ -\frac{T}{J}r\sin\alpha - \omega^2 r\cos\alpha \end{pmatrix}. \end{aligned} \tag{4.11}$$

The equations

$$\dot{\alpha} = \omega \tag{4.12}$$

$$\dot{\omega} = \frac{T}{J} \tag{4.13}$$

still hold. Due to the definition of $\boldsymbol{R}$ in Eq. (4.8), this can be rewritten as

$$\begin{pmatrix}\dot{c}_3\\ \dot{c}_4\end{pmatrix} = \boldsymbol{R}\begin{pmatrix}F_x\\ F_y + \|\boldsymbol{F}_{\text{virt}}\|\end{pmatrix}\frac{1}{m} + \begin{pmatrix}\cos\alpha & -\sin\alpha\\ \sin\alpha & \cos\alpha\end{pmatrix}\begin{pmatrix}-T\frac{r}{J}\\ 0\end{pmatrix} + \begin{pmatrix}\cos\alpha & -\sin\alpha\\ \sin\alpha & \cos\alpha\end{pmatrix}\begin{pmatrix}0\\ -\omega^2 r\end{pmatrix} \tag{4.14a}$$

$$= \begin{pmatrix}\cos\alpha & -\sin\alpha\\ \sin\alpha & \cos\alpha\end{pmatrix}\begin{pmatrix}\frac{F_x}{m} - T\frac{r}{J}\\ \frac{F_y+\|\boldsymbol{F}_{\text{virt}}\|}{m} - \omega^2 r\end{pmatrix} = \boldsymbol{R}\left(\begin{pmatrix}\frac{F_x}{m} - T\frac{r}{J}\\ \frac{F_y}{m}\end{pmatrix} + \begin{pmatrix}0\\ \frac{\|\boldsymbol{F}_{\text{virt}}\|}{m} - \omega^2 r\end{pmatrix}\right) \tag{4.14b}$$

$$= \boldsymbol{R}\left(\begin{pmatrix}\frac{F_x}{m} - T\frac{r}{J}\\ \frac{F_y}{m}\end{pmatrix} + \begin{pmatrix}0\\ (\omega_0^2 - \omega^2)r\end{pmatrix}\right) \tag{4.14c}$$

with a proper selection of $\|\boldsymbol{F}_{\text{virt}}\|$, r and ω_0 according to Eq. (4.6). Introduce a new input using the clearly invertible linear map

$$\begin{pmatrix}\breve{u}_1\\ \breve{u}_2\\ \breve{u}_3\end{pmatrix} = \underbrace{\begin{pmatrix}\frac{1}{m} & 0 & -\frac{r}{J}\\ 0 & \frac{1}{m} & 0\\ 0 & 0 & \frac{1}{J}\end{pmatrix}}_{=:\boldsymbol{M}}\underbrace{\begin{pmatrix}F_x\\ F_y\\ T\end{pmatrix}}_{=\boldsymbol{F}} \tag{4.15}$$

and the "de-turned" input with an as well invertible map

$$\underbrace{\begin{pmatrix}u_1\\ u_2\\ u_3\end{pmatrix}}_{=:\boldsymbol{u}} = \underbrace{\begin{pmatrix}\cos\alpha & -\sin\alpha & 0\\ \sin\alpha & \cos\alpha & 0\\ 0 & 0 & 1\end{pmatrix}}_{=\mathfrak{R}}\begin{pmatrix}\breve{u}_1\\ \breve{u}_2\\ \breve{u}_3\end{pmatrix}. \tag{4.16}$$

Then, after rearranging the rows, the new system is

$$\begin{pmatrix}\dot{c}_1\\ \dot{c}_2\\ \dot{c}_3\\ \dot{c}_4\\ \dot{\omega}\\ \dot{\alpha}\end{pmatrix} = \begin{pmatrix}c_3\\ c_4\\ \begin{pmatrix}-\sin\alpha\\ \cos\alpha\end{pmatrix}(\omega_0^2 - \omega^2)r + \begin{pmatrix}u_1\\ u_2\end{pmatrix}\\ u_3\\ \omega\end{pmatrix} = \begin{pmatrix}c_3\\ c_4\\ -\sin\alpha(\omega_0^2 - \omega^2)r\\ \cos\alpha(\omega_0^2 - \omega^2)r\\ 0\\ \omega\end{pmatrix} + \begin{pmatrix}0\\ 0\\ u_1\\ u_2\\ u_3\\ 0\end{pmatrix} \tag{4.17}$$

where, as follows from above,

$$u = \mathfrak{R}\boldsymbol{M}\boldsymbol{F} \tag{4.18}$$

$$\boldsymbol{F} = \boldsymbol{M}^{-1}\mathfrak{R}^{-1}u. \tag{4.19}$$

The error dynamics of the system, with a trajectory given as

$$\chi(t) = \begin{pmatrix}\chi_1(t) & \chi_2(t) & (\cdot)\end{pmatrix}^T \tag{4.20}$$

follow from $e = x - \chi$:

$$\begin{aligned}
\dot{e} &= \frac{d}{dt}\begin{pmatrix} c_1 - \chi_1 \\ c_2 - \chi_2 \\ c_3 - \dot{\chi}_1 \\ c_4 - \dot{\chi}_2 \\ \omega - \omega_0 \\ \alpha \end{pmatrix} = \begin{pmatrix} \dot{e}_1 \\ \dot{e}_2 \\ \dot{e}_3 \\ \dot{e}_4 \\ \dot{e}_5 \\ \dot{\alpha} \end{pmatrix} \\
&= \begin{pmatrix} e_3 \\ e_4 \\ -\sin\alpha(\omega_0^2 - (\omega_0 + e_5)^2)r - \ddot{\chi}_1 \\ \cos\alpha(\omega_0^2 - (\omega_0 + e_5)^2)r - \ddot{\chi}_2 \\ 0 \\ \omega \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ u_1 \\ u_2 \\ u_3 \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} e_3 \\ e_4 \\ \sin\alpha f(e_5) - \ddot{\chi}_1 \\ -\cos\alpha f(e_5) - \ddot{\chi}_2 \\ 0 \\ \omega \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ u_1 \\ u_2 \\ u_3 \\ 0 \end{pmatrix}
\end{aligned} \tag{4.21}$$

In the last step, the new definition

$$f(e_5) = (2\omega_0 + e_5)e_5 r \tag{4.22}$$

was used.

The obvious advantage of this formulation is that the trajectory itself does not need to be modified: Once an error has occurred, only the controller itself needs to be modified and not both controller and trajectory. Additionally, the formulation is rather compact.

The new state $\begin{pmatrix} c_1 & c_2 \end{pmatrix}^T$ is called the center point in this work, even though this could be seen as slightly imprecise as explained in the following important remark:

> **Remark 18:** A motion that actually resembles an orbit can only be observed if the error is small compared to the radius. Only once it is zero, the robot will actually orbit around the trajectory with the given r and ω_0. A more precise name for the point $\begin{pmatrix} c_1 & c_2 \end{pmatrix}^T$ would therefore be the *desired* center point. What this means in practice can be seen in Chapter 6.

Concerning the control of the systems, two more things can be noticed:

> **Remark 19:** Notice that the input transformation Eq. (4.19) implicitly depends on the state due to $\mathfrak{R}$. Due to the invertibility of the transformation, a control law can be found that uses $\boldsymbol{u}$. For deriving sets of valid control laws, either a transformation of the input set for $\boldsymbol{F}$ into an input set for $\boldsymbol{u}$ is necessary, or the inverse transformation needs to be taken into account.

> **Remark 20:** The system is orbiting around the point on the trajectory that it would track in a non-faulty case. The orientation, in turn, is also defined for the nominal trajectory but dropped when the orbiting starts. Therefore, the third element in Eq. (4.20) is not given explicitly. It is also never taken into account when calculating the error towards the trajectory.

4.5 Concluding Remarks

This chapter presented micro-orbiting, a strategy where a freeflyer under actuator failures is brought into a controlled orbit around a fixed or moving point. One of the main features of the proposed solution is that it makes very few assumptions on the type of actuator fault. It further solves the problem in a very general fashion without the need of treating a lot of edge cases which would appear if the position of the system itself was controlled instead.

One of the drawbacks of the method is certainly that the orientation of the controller is not controlled any more. In a normal case, this would be unacceptable as the system cannot serve its original purpose any more. When it comes to control under actuator failures, contrarily, the main concern is to maintain safety and to avoid crashes and further damage. This is ensured as micro-orbiting still allows the system to autonomously return to a place where help is available and the actuators can be repaired.

The concluding part of the chapter consists of a couple of observations and remarks that are not directly connected to each other. They provide further insight into the method, but are not necessary for the rest of the work.

Different Motivations for Micro-Orbiting Micro-orbiting was, in this work's line of thought, mainly motivated by the need to resolve the nonholonomicity that appears for both C2 and C3. The philosophy behind it what thus that instead of resolving all different cases of nonholonomicity, a new problem is formulated that does not suffer from this any more.

If only C3 is considered, however, another interpretation is possible: As the trajectory might not be feasibly any more, orbiting may be seen as a general description of a new path that is both feasible for all possible fault scenarios *and* does not diverge from the original path too much.

Reasons for the feasibility of the approach While it was explained how the orbiting is done, no thought has been put onto why it is possible so far. It is clear from Eq. (4.21) that the most important part is to keep $f(e_5)$ and therefore e_5 (the error in ω) as close to zero as possible: Only then, the nonlinear term will vanish (which, in physical terms, happens by the cancellation of the virtual fault and the inertial forces).

Thus, it is of crucial importance to have control authority over the angular velocity, which is in turn controlled by the torque $T = Ju_3$. This is always guaranteed with C2 and C3 by definition.

It has been shown that for up to three failing actuators, only the cases C1- C3 occur. But to understand the reasons for the feasibility of micro-orbiting on a fundamental level, it needs to be understood why only these cases appear:

The approach functions as the actuators of the system are placed off-centered, so each of them contributes to the overall torque. Hence, the torque remains controllable for a lot of combinations of actuator failures. To put it into other words, one could say that there is more redundancy for controlling the rotation than for straight movements.

Comparison to Fault-Tolerant Control for Quadrotors The already earlier cited works [24, 27] propose ways to control quadrotors under actuator failures. They also let go of the control of some states, particularly the yaw rate. The difference to this approach is that they do not use this actively: According to [72], the control of the yaw direction becomes infeasible under the considered actuator failures, wherefore they simply exclude it from the control scheme. It may also be noted, anticipating the next chapter, that [27] does give any guarantees for the proposed MPC.

Comparison to ε-tracking Micro-orbiting has a curious similarity to a work on the control of surface vessels, see [73, 74]. There, a feedback-controller is presented that globally stabilizes a surface vessel up to an arbitrary small error. This is achieved by not controlling the position of the surface vessel, but a point arbitrarily close to its position that lies either right or left of the vessel. By not considering the actual position and instead a point next to it, (in combination with some state transforms) the nonholonomicity is avoided there, similar to the center dynamics in this work.

The perspective from where the result was found was however a very different one: Both cited works do not consider actuator constraints. The nonholonomic constraints that they take into account stem from the system equations, even though they are similar in the way that they constrain "sideways" movement. This is the reason why their controller cannot be readily applied in the present work for all scenarios. It may be noted, nonetheless, that the "sliding unicycle" from Remark 9 can indeed be stabilized using it.

Generalization to 3D movements As a concluding remark to this section, statements similar to the 2D-orbiting are possible for 3D-rotations. A transfer of the presented control approach to 3D-systems is currently underway. It may be noted that it is not as straightforward as it may seem as Euler's rotation equations suggest a *coupled* nonlinear system of differential equations. On the other hand, the micro-orbit around the trajectory can be chosen to be any orbit on a sphere, so there are more degrees of freedom that can be exploited.

Chapter 5

Control Through Micro-Orbiting

In the previous chapter, micro-orbiting was presented as a strategy that allows the freeflyers to operate even under actuator failures. By keeping the virtual uncontrollable force directed at the center of a micro-orbit and orbiting at the angular speed ω_0, the orbit dynamics and the virtual fault cancel out. It then becomes possible to control the resulting system. In the following chapter, it is shown how to do that using three different controllers.

The first controller in Section 5.1 is a feedback controller based on feedback linearization. It is capable of controlling the robot with a low computational effort, but not able to avoid unsafe sets of states. The controller can only stabilize a setpoint and not follow a trajectory, but on the other hand it is shown that parameters exist that make the stabilization global. The second and the third controller are MPC controllers and therefore may satisfy additional constraints concerning their region of stability. They also allow for tracking.

Both MPC controllers rely on a terminal controller for which terminal costs and terminal sets are derived. The difference lies in the terminal controllers used: For the second controller in Section 5.2, the system is feedback linearized and locally controlled with linear feedback which leads to a small terminal set. The third controller in Section 5.3, in contrast, uses a terminal controller based on feedback linearization techniques together with explicit MPC. This has, as to the best knowledge of the author, not been proposed yet and leads to a much larger terminal set.

5.1 Feedback-linearizing Control

Feedback-linearization is a powerful technique and one of the most universally applicable approaches for the control of nonlinear systems. It represents the basis for the first controller that is shown in this chapter. Interested readers can get a good overview over the possibilities and drawbacks in [70, Chap. 5].

The feedback linearizing controller inverts the nonlinear dynamics in Eq. (4.21) which gives the linearized error dynamics of the orbit center

$$\dot{e} = \begin{pmatrix} e_3 \\ e_4 \\ 0 \\ 0 \\ 0 \\ \omega \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ u_1 \\ u_2 \\ u_3 \\ 0 \end{pmatrix} \tag{5.1}$$

and the control law

$$\boldsymbol{u} = \begin{pmatrix} -\sin\alpha f(e_5) + \ddot{\chi}_1 \\ \cos\alpha f(e_5) + \ddot{\chi}_2 \\ 0 \end{pmatrix} - Ke. \tag{5.2}$$

Choose K as

$$K = \begin{pmatrix} k_1 & 0 & k_2 & 0 & 0 \\ 0 & k_1 & 0 & k_2 & 0 \\ 0 & 0 & 0 & 0 & k_3 \end{pmatrix} \tag{5.3}$$

where the parameters k_1, k_2 and $k_3 > 0$ are yet unspecified. Define a saturation function as follows:

Definition 10 (Saturation through truncation): Take a control input $u \notin \mathcal{U}$ and introduce $\mathcal{U}$ as a convex input set satisfying $\mathbf{0} \in \text{Int}(\mathcal{U})$. The saturated control input $u_{\text{trunc}} \in \mathcal{U}$ is determined as the intersection between the vector u and $\partial\mathcal{U}$.

Remark 21: Note that the saturation is not achieved by projection on $\partial\mathcal{U}$ but rather by truncating at the intersection of $\boldsymbol{u}$ with $\partial\mathcal{U}$ (see Figure 5.1). The truncation preserves the signs of each variable of $\boldsymbol{u}$ which the projection does not do necessarily. The first statement is clear as the truncation can be written as $\boldsymbol{u}_{\text{trunc}} = \gamma\boldsymbol{u}$ with a suitable $\gamma \in (0, 1]$. The statement for the projection can be easily shown through contradiction with a counterexample.

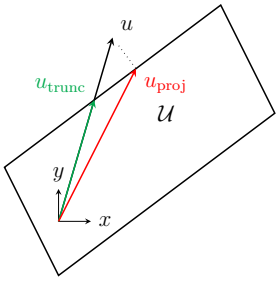


Figure 5.1: Different saturation strategies via truncation or projection.

The following theorem establishes stability for the controller:

Theorem 4: Consider the system with the dynamics given in Eq. (4.21) and input constraints $\boldsymbol{M}\mathfrak{R}\boldsymbol{u} \in \mathcal{U}$, with $\mathcal{U}$ from Definition 9 where $\mathbf{0} \in \mathrm{Int}(\mathcal{U})$. Let the controller be given by Eq. (5.2) with an additional saturation function as in Definition 10.

Then, for all starting states $x_0 \in \mathbb{R}^{n_x}$, suitable parameters k_1, k_2 and k_3 can be found that globally asymptotically stabilize errors e_1 to e_5 at zero.

The sixth state α is not stabilized, but this a direct consequence of micro-orbiting with a constant angular velocity.

Proof First consider the system without input constraints: In this case, solve the Lyapunov equation by direct calculation, where $C^TC = \boldsymbol{I}_{5\times5}$, and get for the closed-loop system the solution

$$P = \begin{pmatrix} \frac{k_1^2+k_2^2+k_1}{2k_1k_2} & 0 & \frac{1}{2k_1} & 0 & 0 \\ 0 & \frac{k_1^2+k_2^2+k_1}{2k_1k_2} & 0 & \frac{1}{2k_1} & 0 \\ \frac{1}{2k_1} & 0 & \frac{k_1+1}{2k_1k_2} & 0 & 0 \\ 0 & \frac{1}{2k_1} & 0 & \frac{k_1+1}{2k_1k_2} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2k_3} \end{pmatrix} \tag{5.4}$$

This matrix is always positive definite for any $k_1, k_2, k_3 > 0$ and the pair (C, A) with $C = \boldsymbol{I}_{5\times5}$ and

$$A = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ -k_1 & 0 & -k_2 & 0 & 0 \\ 0 & -k_1 & 0 & -k_2 & 0 \\ 0 & 0 & 0 & 0 & -k_3 \end{pmatrix} \tag{5.5}$$

is trivially observable. Note that the sixth state α has been left out in this statement as it is not a controlled variable. Therefore (ref. [43, Thm. 8.2]), the system without state α is stable.

Before the input-constrained case is treated explicitly, observe that the transformation $\boldsymbol{M}\mathfrak{R}\boldsymbol{u}$ allows to derive an expression for $\boldsymbol{u}$ that is independent of α

$$\boldsymbol{u} \in \mathcal{U}_u := \bigcap_{\alpha\in[0,2\pi)} \boldsymbol{M}\mathfrak{R}\mathcal{U} \tag{5.6}$$

where, for every fixed α, the transformation $\boldsymbol{M}\mathfrak{R}\mathcal{U}$ is applied for every point in $\mathcal{U}$. The set is the intersection of $\boldsymbol{M}\mathfrak{R}(\alpha)\mathcal{U}$ for every α, i.e. the intersection of all possible combinations of rotation and linear transformation of $\mathcal{U}$. Intuitively speaking, it will result in a body of revolution with rotation around the u_3- resp. T-axis. Notice that this set, equally to $\mathcal{U}$, contains zero in its interior, i.e. $\mathbf{0} \in \mathrm{Int}(\mathcal{U}_u)$. This follows directly as, with a fixed α, both transformations are linear maps.

In order to extend the stability result for the input-constrained case, first notice that $u_3 = \frac{T}{J}$ and $\dot{e}_5 = u_3$. Conditions for the stability of the e_5-subsystem can be derived using

the Lyapunov function

$$V(e_5) = \frac{1}{2}e_5^2 \tag{5.7}$$

$$\Rightarrow \dot{V}(e_5) = e_5 u_3 \overset{!}{<} 0 \ \forall e_5 \neq 0. \tag{5.8}$$

The last requirement can easily be satisfied by $\text{sgn}(u_3) \neq \text{sgn}(e_5)$. This is always satisfied due to $u_3 = -k_3 e_5$ following from the control law and as the saturation preserves the sign of the input. Accordingly, it follows stability in terms of Lyapunov for e_5 even under the input constraints.

In general, it is not always ensured that the nonlinear compensation is possible under satisfaction of $\boldsymbol{u} \in \mathcal{U}_u$. However, as of Eq. (4.22), $e_5 \to 0 \Rightarrow f(e_5) \to 0$ for $t \to \infty$ and therefore the compensation will always be possible for a finite $t \geq \tilde{t}$ with a suitable bounded $\tilde{t} \geq t_0$.

From this follow two things: First, the e_5-subsystem is globally stable and second, the e_1-e_4-subsystem always becomes feedback linearizable. Additionally, the parameters for K can be chosen freely and will always stabilize the system.

Now introduce a Lyapunov function $V(e) = e^T P e$ with P from Eq. (5.4). Choose the parameters such that the control output satisfies the constraints for $\partial\tilde{\mathcal{E}}$ with

$$\tilde{\mathcal{E}} = \{e | V(e) \leq V(\tilde{e})\} \tag{5.9}$$

where $\tilde{e}$ is the error at time $\tilde{t}$ when the feedback linearization becomes possible. Then, the control law is feasible under the input constraints for all $e \in \tilde{\mathcal{E}}$. Moreover, $\tilde{\mathcal{E}}$ is a level set of the Lyapunov function and thus forms a natural control invariant set. Accordingly, $e \in \tilde{\mathcal{E}} \ \forall t \geq \tilde{t}$ and therefore $\boldsymbol{u} \in \mathcal{U}_u \ \forall t \geq \tilde{t}$. The proposed controller therefore both globally asymptotically stabilizes the system and satisfies the input constraints. ∎

> **Remark 22:** For e close to the boundary, i.e. $e \in \tilde{\mathcal{E}} \to \partial\tilde{\mathcal{E}}$, the parameters for K are close to zero. As $e_5 \to 0$ for t increasing, e moves away from the boundary and it becomes possible to choose larger values.

With the result from Theorem 4, the global stability of the proposed controller has been shown. This shows the capability of micro-orbiting as a strategy to overcome input constraints posed though failing actuators.

One thing the controller lacks, on the other side, is the ability to avoid unsafe states which often mark states that would result in crashes. This can be solved using one of the controllers that are presented in the following.

5.2 Model Predictive Control Using Feedback-Linearization

The second controller presented in this work is an MPC controller which addresses some limitations of the controller described in Section 5.1. This MPC controller employs a terminal controller based on feedback linearization of the system's nonlinearities. The

approach draws inspiration from [55], but differs in its linearization method. Instead of using a first-order Taylor approximation, this terminal controller applies feedback linearization. For a detailed comparison between these approaches, see Remark 28.

As MPC is a discrete-time method, the model is discretized using the Euler-forward method. For the MPC controller itself, the error dynamics of the orbit center (Eq. (4.21)) are discretized, but the non-transformed input $\boldsymbol{F}$ is used instead of $\boldsymbol{u}$ (ref. Eq. (4.19)).

$$\begin{aligned} e_{t+1} &= e_t + \delta \dot{e}_t \\ &= e_t + \delta \left[\begin{pmatrix} e_3 \\ e_4 \\ \sin\alpha f(e_5) - \ddot{\chi}_1 \\ -\cos\alpha f(e_5) - \ddot{\chi}_2 \\ 0 \\ \omega_0 + e_5 \end{pmatrix} + \begin{pmatrix} \boldsymbol{I}_{2\times 3} \\ \mathfrak{R}\boldsymbol{M} \\ \boldsymbol{I}_{1\times 3} \end{pmatrix} \boldsymbol{F} \right] \\ &=: \Phi(e_t, \boldsymbol{F}_t) \end{aligned} \tag{5.10}$$

The trajectory tracking MPC problem is formulated in agreement with Section 2.3.2 using the input

$$\tilde{u}_{t+k|t} = m\begin{pmatrix} \ddot{\chi}_1(t+k) & \ddot{\chi}_2(t+k) & 0 \end{pmatrix}^T \tag{5.11}$$

that perfectly tracks the trajectory and $\boldsymbol{F}_{t+k|t} = \bar{u}_{t+k|t} + \mathfrak{R}^{-1}_{t+k|t}\tilde{u}_{t+k|t}$.

> **Remark 23:** As the error dynamics of the orbit center are used, the additional constant input $\boldsymbol{F}_{\text{comp}}$ is applied implicitly.

> **Remark 24:** The nominal inputs $\tilde{u}_{t+k|t}$ are in global coordinates. As the orientation $\alpha_{t+k|t}$ is not known a priori, the transformation $\mathfrak{R}^{-1}_{t+k|t}$ becomes part of the MPC problem.

The MPC problem can then be formulated as

$$\begin{aligned} \underset{\bar{U}}{\text{minimize}} \quad & V(e_0) = \sum_{k=0}^{N-1} \left[e_{t+k|t}^T Q_e e_{t+k|t} + \bar{u}_{t+k|t}^T Q_u \bar{u}_{t+k|t} \right] + l_N(e_{t+N|t}) \\ \text{subject to} \quad & e_{t+k+1|t} = \Phi(e_{t+k|t}, \bar{u}_{t+k|t} + \mathfrak{R}^{-1}_{t+k|t}\tilde{u}_{t+k|t}) \quad k = 0, \dots, N-1, \\ & e_{t+k|t} \in \mathcal{X} \qquad\qquad k = 0, \dots, N-1, \\ & \bar{u}_{t+k|t} + \mathfrak{R}^{-1}_{t+k|t}\tilde{u}_{t+k|t} \in \mathcal{U} \qquad\qquad k = 0, \dots, N-1, \\ & e_{t+N|t} \in \mathcal{X}_{\text{term}}, \\ & e_{t|t} = e_0 \end{aligned} \tag{5.12}$$

The set $\mathcal{X}$ is the set of safe states and the set $\mathcal{U}$ is given by Definition 9. The expressions for the terminal set $\mathcal{X}_{\text{term}}$ and the terminal cost $l_N(e)$ are not yet specified. The rest of this section is dedicated to designing them such that they stabilize the system.

5.2.1 Terminal Controller

Stability is ensured by employing a dual-mode principle with an MPC controller until horizon N and a terminal controller for $k \geq N$. The terminal controller is, due to the receding horizon, never actually used, but serves to ensure that at least one stabilizing control law exists at all time. In the MPC problem (5.12), the existence of the terminal controller is reflected implicitly through the terminal cost and set.

In the following, the terminal controller is proposed. It is derived similarly to the controller for Section 5.1 and a schematic block diagram is shown in Figure 5.2. Ignoring the difference between the continuous-time and discrete-time models, the inner system shows the block diagram of the orbit dynamics as in Eq. (4.21) with input $\boldsymbol{u}$, while the outer system shows them as in Eq. (5.10) with the input $\boldsymbol{F}$.

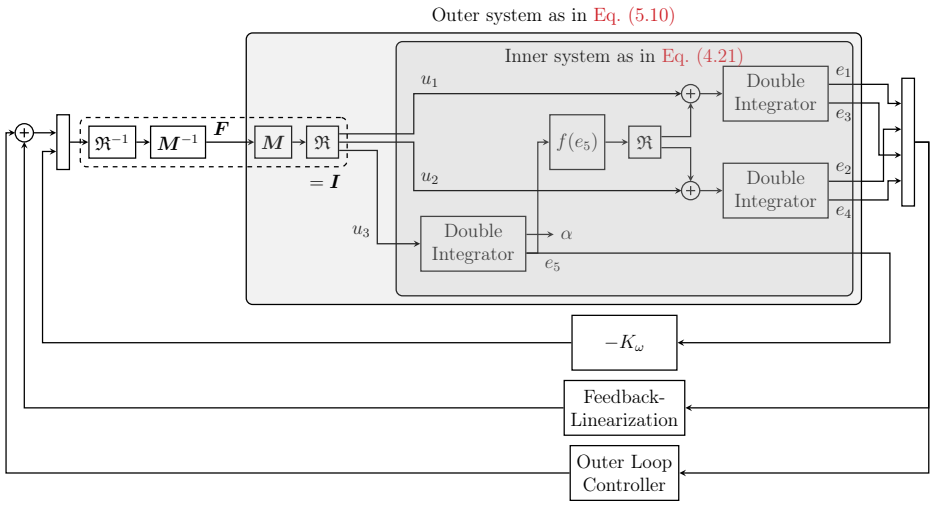


Figure 5.2: The terminal controller for the MPC controllers based on feedback linearization, a linear controller for e_5 and an additional controller for the linearized system. The mux and demux blocks have been borrowed from Simulinks notation.

As said in the foregoing section, the input of the MPC controller is $\boldsymbol{F}$. The terminal controller, in contrast, compensates for $\boldsymbol{M}$ and $\mathfrak{R}$, which needs to be taken into account once the terminal cost is calculated. Using the compensation, the part in the dashed box cancels out and the terminal controller can also be displayed as in Figure 5.3.

> **Remark 25:** From Figure 5.3, an important aspect of the system structure becomes visible, as also discussed in Section 4.5: The integrators of e_5 and α form a closed loop with the linear feedback law $-K_\omega e_5$. The nonlinearity only depends on e_5, which is why it can be cancelled out completely.

Figures 5.2 and 5.3 are the schematics for both the current terminal controller as well as the one of the following section. The difference lies in the block named "Outer Loop Controller", which is a different controller for both cases. In this section, it is a simple linear feedback controller.

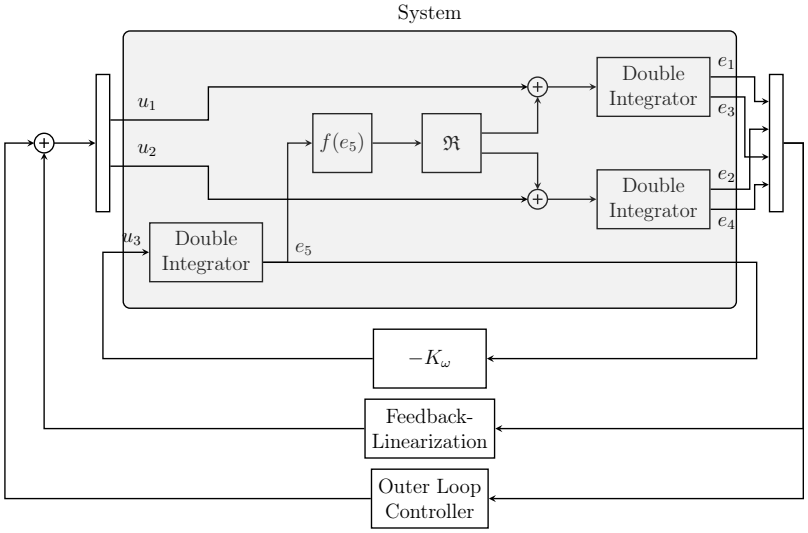


Figure 5.3: The terminal controller after compensation of $\mathfrak{R}\boldsymbol{M}$.

For this outer loop control law, the full (compensating) terminal control law can be given as

$$\begin{aligned}\bar{u} &= \boldsymbol{M}^{-1}\mathfrak{R}^{-1}\left[\begin{pmatrix}-\sin\alpha f(e_5)\\ \cos\alpha f(e_5)\\ 0\end{pmatrix} - Ke_k\right] \\ &= \boldsymbol{M}^{-1}\mathfrak{R}^{-1}[\mathfrak{R}\boldsymbol{f}(e_5) - Ke_k]\end{aligned} \tag{5.13}$$

with

$$\boldsymbol{f}(e_5) = \begin{pmatrix}0\\ f(e_5)\\ 0\end{pmatrix}. \tag{5.14}$$

The state feedback matrix K is chosen such that the feedback-linearized system of the form

$$e_{k+1} = \begin{pmatrix}\mathbf{0}_{2\times3} & \boldsymbol{I}_{2\times2}\\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times2}\end{pmatrix}e_k + \begin{pmatrix}\mathbf{0}_{2\times3}\\ \boldsymbol{I}_{3\times3}\end{pmatrix}(-K)e_k \tag{5.15}$$

is stable. The proof of stability is postponed to Section 5.2.3 where a set is derived for which the controller is guaranteed feasible.

5.2.2 Terminal Cost

The real infinite-horizon cost $\tilde{l}_N(e_0)$ of the terminal controller can be calculated as

$$\tilde{l}_N(e_0) = \sum_{k=0}^{\infty}\left[e_k^T Q_e e_k + \bar{u}_k^T Q_u \bar{u}_k\right]. \tag{5.16}$$

where the index starting from N is already shifted to zero for simplicity of notation. Calculating this function is, for general nonlinear systems, a tedious endeavour: First an explicit formulation for the trajectory and then another explicit form for the quadratic

cost needs to be found. In practice, this is often infeasible. Thus, in the following an upper bound $l_N(e) \geq \tilde{l}_N(e)$ is derived.

For linear systems, explicit solutions for the state evolution are known and due to the feedback-linearization, the state evolution of the given nonlinear system behaves as a linear one. With a feedback matrix A_K of the linearized and controlled closed-loop system, the state trajectory can directly be calculated from

$$e_{k+1} = A_K e_k \tag{5.17}$$

as

$$e_k = A_K^k e_0. \tag{5.18}$$

The value of the infinite sum $\sum_{k=0}^{\infty} e_k^T Q_e e_k$, the left term of Eq. (5.16), can be calculated as $e_0^T P_Q e_0$. P_Q is here the solution of the corresponding Lyapunov function $A_K^T P_Q A_K - P_Q = -Q_e$ which always has a solution as the controlled system is stable.

For the second term, however, the explicit calculation is not as straightforward: The explicit formulation of the state evolution still holds, but the controller has nonlinearities, on the one hand through the rotation matrix $\mathfrak{R}$ and on the other hand through $f(e_5)$. Instead, an upper bound for the control cost is derived in the following.

In order to achieve this, the following inequality is presented an intermediate result:

$$\begin{aligned}\left[\boldsymbol{f}(e_5) + \mathfrak{R}^{-1}(-Ke_k)\right]^T \tilde{Q}_u \left[\boldsymbol{f}(e_5) + \mathfrak{R}^{-1}(-Ke_k)\right] \\ \leq 2\|\tilde{Q}_u\|_2 \left(\|\boldsymbol{f}(e_5)\|_2^2 + \| - Ke_k\|_2^2\right)\end{aligned} \tag{5.19}$$

Proof It is derived by

$$\begin{aligned}&\left[\boldsymbol{f}(e_5) + \mathfrak{R}^{-1}(-Ke_k)\right]^T \tilde{Q}_u \left[\boldsymbol{f}(e_5) + \mathfrak{R}^{-1}(-Ke_k)\right] \\ =&\left\|\boldsymbol{L}^T \left(\boldsymbol{f}(e_5) + \mathfrak{R}^{-1}(-Ke_k)\right)\right\|_2^2\end{aligned}$$

Here, $\tilde{Q}_u$ is a positive definite, symmetric matrix and therefore, the Cholesky decomposition $\tilde{Q}_u = \boldsymbol{L}\boldsymbol{L}^T$ exists. Continue with

$$\begin{aligned}\leq&\|\boldsymbol{L}\|_2^2 \left\|\left(\boldsymbol{f}(e_5) + \mathfrak{R}^{-1}(-Ke_k)\right)\right\|_2^2 \\ \leq&\|\tilde{Q}_u\|_2 \left(\|\boldsymbol{f}(e_5)\|_2 + \|\mathfrak{R}^{-1}(-Ke_k)\|_2\right)^2\end{aligned}$$

Note that $\|\mathfrak{R}\|_2 = 1$ as $\mathfrak{R}$ is a rotation matrix and thus

$$\begin{aligned}=&\|\tilde{Q}_u\|_2 \left(\|\boldsymbol{f}(e_5)\|_2 + \| - Ke_k\|_2\right)^2 \\ \leq&2\|\tilde{Q}_u\|_2 \left(\|\boldsymbol{f}(e_5)\|_2^2 + \| - Ke_k\|_2^2\right)\end{aligned}$$

In the last step, the Arithmetic-Mean-Quadratic-Mean Inequality [75], i.e.

$$(a+b)^2 \leq 2(a^2+b^2) \text{ for } a,b \geq 0$$

has been used which closes the derivation. ∎

Now bound the infinite-horizon control cost of the controller (5.13) from above:

$$\sum_{k=0}^{\infty} \bar{u}_k^T Q_u \bar{u}_k \tag{5.20}$$

$$= \sum_{k=0}^{\infty} \left[\boldsymbol{M}^{-1}\mathfrak{R}^{-1}(\mathfrak{R}\boldsymbol{f}(e_5) - Ke_k)\right]^T Q_u \left[\boldsymbol{M}^{-1}\mathfrak{R}^{-1}(\mathfrak{R}\boldsymbol{f}(e_5) - Ke_k)\right] \tag{5.21}$$

$$\leq \sum_{k=0}^{\infty} [-\mathfrak{R}^{-1}Ke_k + \boldsymbol{f}(e_5)]^T \underbrace{(\boldsymbol{M}^{-1})^T Q_u \boldsymbol{M}^{-1}}_{:=\tilde{Q}_u} [-\mathfrak{R}^{-1}Ke_k + \boldsymbol{f}(e_5)] \tag{5.22}$$

where $\boldsymbol{M}$ is invertible and Q_u positive definite, wherefore also $\tilde{Q}_u$ is positive definite and Eq. (5.19) is applicable. The full infinite-horizon cost of the controller (5.16) follows then as

$$\begin{aligned}
\tilde{l}_N(e_0) &= \sum_{k=0}^{\infty} e_k^T Q_e e_k + \bar{u}_k^T Q_u \bar{u}_k \\
&\leq \sum_{k=0}^{\infty} \left[e_k^T Q_e e_k + 2\|\tilde{Q}_u\|_2 \left(\|\boldsymbol{f}(e_5)\|_2^2 + \|(-Ke_k)\|_2^2 \right) \right] \\
&= e_0^T P_Q e_0 + 2\|\tilde{Q}_u\|_2 \left(e_0^T P_K e_0 + \frac{(2\omega_0 r)^2 e_5^2[0]}{1-(1-K_\omega)^2} + \frac{4\omega_0 r e_5^3[0]}{1-(1-K_\omega)^3} + \frac{r^2 e_5^4[0]}{1-(1-K_\omega)^4} \right) \\
&:= l_N(e_0)
\end{aligned} \tag{5.23}$$

The last expression follows as the infinite sum can be calculated explicitly: The parts $\sum_{k=0}^{\infty} e_k^T Q_e e_k$ and $\sum_{k=0}^{\infty} \|(-Ke_k)\|_2^2 = \sum_{k=0}^{\infty} e_k^T K^T K e_k$ can directly be evaluated by finding a solution P_Q resp. P_K to the corresponding Lyapunov function. The solvability follows as stated earlier from the stability of the linearized system and its system matrix A_K.

For the middle term, the linear dynamics $e_5[k+1] = (1-K_\omega)e_5[k] \Rightarrow e_5[k] = (1-K_\omega)^k e_5[0]$ and the formula for the infinite geometric series can be used:

$$\sum_{k=0}^{\infty} \|\boldsymbol{f}(e_5)\|_2^2 = \sum_{k=0}^{\infty} \|f(e_5)\|_2^2 \tag{5.24}$$

$$= \sum_{k=0}^{\infty} (2\omega_0 r)^2 e_5^2 + 4\omega_0 r e_5^3 + r^2 e_5^4 \tag{5.25}$$

$$= \sum_{k=0}^{\infty} (2\omega_0 r)^2 ((1-K_\omega)^2)^k e_5^2[0] + 4\omega_0 r ((1-K_\omega)^3)^k e_5^3[0] + r^2 ((1-K_\omega)^4)^k e_5^4[0] \tag{5.26}$$

$$= \frac{(2\omega_0 r)^2 e_5^2[0]}{1-(1-K_\omega)^2} + \frac{4\omega_0 r e_5^3[0]}{1-(1-K_\omega)^3} + \frac{r^2 e_5^4[0]}{1-(1-K_\omega)^4} \tag{5.27}$$

The convergence of the expression follows from the stability of the linearized system which implies $\|1 - K_\omega\| < 1$.

5.2.3 Terminal Set

The inequality that is derived in the previous section is a bound on the infinite horizon cost if the terminal control law (5.13) is applied. It is based on the fact that the state evolution of the feedback-linearized system can be calculated explicitly. Therefore, the bound only holds in the region where the control law is admissible under the input constraints. Thus, the terminal set is determined in the present section as a control invariant set that satisfies the input constraints.

In order to find a feasible terminal set, take the set

$$\begin{aligned} \mathcal{U}_\mathrm{f} &= \{\boldsymbol{F} = \boldsymbol{M}^{-1}\mathfrak{R}^{-1}u \in \mathcal{U} | u^T u \leq r_\Phi\} \\ &= \{\boldsymbol{MF} = \mathfrak{R}^{-1}u \in \boldsymbol{M}\mathcal{U} | u^T u \leq r_\Phi\} \\ &= \{\boldsymbol{MF} = u \in \boldsymbol{M}\mathcal{U} | u^T u \leq r_\Phi\} \end{aligned} \tag{5.28}$$

as the largest ball of compensated inputs u satisfying the input constraints that is centered at the origin. Notice that $\mathfrak{R}$ does not need to be taken into account explicitly as the set is a ball and therefore rotationally invariant, i.e. $u \in \mathcal{U}_\mathrm{f} \Rightarrow \mathfrak{R}^{-1}u \in \mathcal{U}_\mathrm{f}\ \forall\alpha$. The set $\boldsymbol{M}\mathcal{U}$ is the linear projection of $\mathcal{U}$ by $\boldsymbol{M}$.

The parameter r_Φ can be found by solving the optimization problem (ref. [42, ch. 8.4.2] for how to formulate this problem as a convex optimization problem)

$$\begin{aligned} \underset{r_\Phi}{\text{maximize}} \quad & r_\Phi \\ \text{subject to} \quad & u \in \boldsymbol{M}\mathcal{U}\ \forall u^T u \leq r_\Phi \end{aligned} \tag{5.29}$$

using a suitable solver. Then,

$$u^T u \leq r_\Phi \tag{5.30}$$

$$\Leftrightarrow \sqrt{r_\Phi} \geq \left\| \begin{pmatrix} -\sin\alpha \\ \cos\alpha \\ 0 \end{pmatrix} f(e_5) + \begin{pmatrix} \ddot{\chi}_1 \\ \ddot{\chi}_2 \\ 0 \end{pmatrix} - Ke \right\|_2 \tag{5.31}$$

$$\geq \left\| \begin{pmatrix} -\sin\alpha \\ \cos\alpha \\ 0 \end{pmatrix} r e_5^2 \right\|_2 + \left\| 2\omega_0 r e_5 \begin{pmatrix} -\sin\alpha \\ \cos\alpha \\ 0 \end{pmatrix} - Ke \right\|_2 + \left\| \begin{pmatrix} \ddot{\chi}_1 \\ \ddot{\chi}_2 \\ 0 \end{pmatrix} \right\|_2 \tag{5.32}$$

$$\Leftrightarrow \underbrace{\sqrt{r_\Phi} - \left\| \begin{pmatrix} \ddot{\chi}_1 \\ \ddot{\chi}_2 \\ 0 \end{pmatrix} \right\|_2}_{=:c_1} \geq \underbrace{r}_{=:c_2} e_5^2 + \underbrace{\max_\alpha \left\| \tilde{K} \right\|_2}_{=:c_3} \|e\|_2 \tag{5.33}$$

where $c_1 - c_3$ are introduced for the ease of notation and therefore

$$\begin{aligned} c_2 e_5^2 + c_3\sqrt{e^T e} &\leq c_1 \\ \Leftrightarrow c_3^2 e^T e &\leq c_1^2 - 2c_1 c_2 e_5^2 + c_2^2 e_5^4 \\ e^T\tilde{C}e \underbrace{-c_2^2 e_5^4}_{\leq 0} \leq e^T\tilde{C}e &\leq c_1^2 \end{aligned} \tag{5.34}$$

where the variables $\tilde{K}$ and $\tilde{C}$ follow by direct calculation.

On the one hand, this poses a constraint on the maximal acceleration of the trajectory under actuator failures. On the other hand, the last inequality marks a set of states within which the control law is feasible.

In order to get an invariant set from Eq. (5.34), another optimization problem can be solved: The Lyapunov function $e^T P e$ of the linearized system naturally gives invariant level sets. The question is which of those is the largest invariant set, i.e.

$$\begin{aligned} &\underset{\beta}{\text{maximize}} \quad \beta \\ &\text{subject to} \quad e^T\tilde{C}e \leq c_1^2 \; \forall e^T P e \leq \beta \end{aligned} \tag{5.35}$$

where β is the upper limit for the level set. Due to the constraint this cannot be implemented directly, but the problem can be reformulated using the s-procedure [42, appendix B.2]. This approach reformulates the present constraint as a definiteness constraint and the problem becomes

$$\begin{aligned} &\underset{\beta}{\text{maximize}} \quad \beta \\ &\text{subject to} \quad \begin{pmatrix} P & \mathbf{0} \\ \mathbf{0} & -\beta \end{pmatrix} - \lambda \begin{pmatrix} \tilde{C} & \mathbf{0} \\ \mathbf{0} & -c_1^2 \end{pmatrix} \succeq 0, \\ &\qquad\qquad \lambda > 0 \end{aligned} \tag{5.36}$$

Once this optimization problem is solved, the terminal set has been found as

$$\mathcal{X}_{\text{term}} = \{e \in \mathbb{R}^5 | e^T P e \leq \beta\}. \tag{5.37}$$

By the derivation, controller (5.13) satisfies the input constraints for all $e \in \mathcal{X}_{\text{term}}$ and dynamically feasible trajectories.

Remark 26: The formulation in Eq. (5.36) is a slight alteration of the classical s-procedure as described in [42]: In its standard formulation, the optimization problem would result as

$$\begin{aligned} &\underset{\beta}{\text{maximize}} \quad \beta \\ &\text{subject to} \quad \tilde{\lambda} \begin{pmatrix} P & \mathbf{0} \\ \mathbf{0} & -\beta \end{pmatrix} - \begin{pmatrix} \tilde{C} & \mathbf{0} \\ \mathbf{0} & -c_1^2 \end{pmatrix} \succeq 0, \\ &\qquad\qquad \tilde{\lambda} \geq 0 \end{aligned} \tag{5.38}$$

> The multiplication of β with $\tilde{\lambda}$ makes this problem non-convex. This can be easily relaxed by introducing a change of variables $\tilde{\lambda} = \frac{1}{\lambda}$ and multiplying by λ, leading to the formulation in Eq. (5.35). This is possible when requiring $\tilde{\lambda} > 0$ instead of $\tilde{\lambda} \geq 0$ and in this case, simply the inverse of $\tilde{\lambda}$ is optimized for.

After $\mathcal{X}_{\text{term}}$ has been derived, the stability properties of the terminal controller can be stated.

> **Lemma 2:** The terminal controller (5.13) stabilizes the discrete-time orbit dynamics (5.10) asymptotically for all states $e \in \mathcal{X}_{\text{term}}$.

Proof Due to the feedback-linearization, once again a quadratic Lyapunov function can be derived similar to the previous controller. A suitable K can be derived for the feedback-linearized system (5.15) by solving the Riccati equation. The solution exists as the system (5.15) is controllable and observable. ∎

Lemma 2 shows the stability of the terminal controller. What is missing is however the stability of the MPC controller (5.12) which is established in the next section.

5.2.4 Proof of Stability

With the terminal cost and set determined, the full result can be given.

> **Proposition 3:** Consider the system with dynamics (5.10), state constraints $e \in \mathcal{X}$ and input constraints $\boldsymbol{F} \in \mathcal{U}$ with $\mathcal{U}$ from Definition 9. Assume the trajectory χ is continuously dynamically feasible in the sense of Definition 2. If the system is controlled with the MPC controller (5.12) with l_N given by Eq. (5.23) and $\mathcal{X}_{\text{term}}$ by Eq. (5.37), the system is asymptotically stable for all states for which problem (5.12) is feasible.

Proof As the stage cost l is a quadratic function, it is continuous, zero at the origin and lower-bounded by a class κ_∞ function for all e_k and u_k.

From Eq. (5.23) one can directly see that the terminal cost l_N satisfies $l_N(0) = 0$, $l_N(e) > 0 \ \forall e \neq 0$ and the continuity of l_N not only at the origin, but globally. Furthermore, the definitions of $\mathcal{X}$, $\mathcal{U}$ and $\mathcal{X}_{\text{term}}$ ensure that they are all closed and contain zero in their interior.

As $\mathcal{X}_{\text{term}}$ was constructed such that a controller exists that stabilizes the system for all $e \in \mathcal{X}_{\text{term}}$ (ref. Lemma 2), the set is control invariant. For the same control law, the terminal cost l_N is constructed in Eq. (5.23) such it bounds the infinite-horizon cost of the controller from above. This directly ensures

$$l_N(\Phi(e_0, \bar{u}_0 + \tilde{u}_0)) - l_N(e_0) + e_0^T Q_e e_0 + \bar{u}_0^T Q_u \bar{u}_0 \leq 0 \tag{5.39}$$

as the inequality holds for every single time step. Therefore, the proof follows immediately from Theorem 2. ∎

Remark 27: An almost identical approach could be used for the continuous-time model of the orbit dynamics. In this case, the stability of the freeflyer under actuator failures can be shown easily and directly. In the following section, however, the last controller is derived which makes use of the inherently discrete eMPC. Therefore, a discrete-time formulation has also been used in this section to unify the stability proofs and to better show the similarities and differences between the controllers.

Remark 28 (Comparison to standard nonlinear MPC methods): One of the classical results for nonlinear MPC is [55]. This work uses a terminal cost and terminal set approach, where the system is linearized around the desired setpoint. The approach goes as follows:

1. Find a locally stabilizing controller K based on the Jacobian linearization of the system
2. Solve a special Lyapunov function whose solution can be used as a terminal cost function. The solution will be ensured to match or overapproximate the actual cost from step N.
3. Find the largest set in which $u = Kx$ satisfies the input constraints $\mathcal{U}$.
4. Find the largest subset in which also the overapproximation holds.

The issue of this approach for the problem at hand is that α is not a controlled variable. Therefore, the Jacobian linearization becomes state-dependent. There are multiple solutions for this:

The obvious, but undesirable, solution would be to have a trajectory that includes the orientation α. Then linearization around the error $\mathbf{0}$ would become possible and the results be applicable. This however removes the advantages of the additional degrees of freedom that the controller has: Not only the orientation, but also the position of the freeflyer on the orbit would be fixed.

Solutions with time-dependent terminal sets would also be possible. They may be, albeit viable, undesirable due to the additional computational cost that they entail.

The final solution is the one used here, where step 1 is replaced by a feedback linearization and the overapproximation of the terminal cost in step 2 is directly calculated instead. The equivalents to steps 3 and 4 is performed in Section 5.2.3 where a terminal set is sought that satisfies all constraints.

It may be noted, however, that key ideas such as (1) finding a terminal controller, (2) deriving a terminal cost and set and (3) overapproximating the actual infinite horizon cost are similar.

The advantage of this approach is that a constant terminal set can be calculated while keeping as many degrees of freedom as possible.

5.3 Model Predictive Control Using Explicit MPC

In anticipation of the discussion, the terminal set for the controller in the previous section is rather small. This makes intuitively sense as the terminal controller needs control action to both feedback-linearize the system and drive the linearized system to zero. The linearized system uses a linear controller with limited available control, resulting in a small set where the control law is valid. This aligns with experience from similar control schemes based on linearization [55] (ref. Remark 28). In the present case, the control authority is further reduced in comparison to the nominal case due to the actuator failures.

With a closer look at the dynamics of the feedback-linearized system dynamics (5.15), one can however see that the linearized system simply consists of multiple integrators. It is therefore intuitively reasonable to assume that the linearized system can even be stabilized using a control law that is saturated once the input constraints are reached. A controller for this is rather easy to derive using techniques for linear systems under input constraints. In order to use it in a terminal controller, the limiting problem is then still to derive a cost function that can be used to show closed-loop stability of the MPC controller.

Besides of traditional anti-windup schemes and similar concepts, a different approach has been proposed around 20 years ago: Explicit MPC (eMPC) allows to formulate (not only input) constrained control problems and to derive a parametric, i.e. explicit state-feedback solution for them. Its advantage in this context is that it also allows to easily derive a cost function that can be used as terminal cost. This comes from its foundation in optimization, which enables it to seamlessly integrate within the existing framework.

This idea will be followed for the last controller that is proposed in this chapter. In particular, the complete terminal controller will be analyzed regarding its stability properties and the resulting terminal cost and set.

Note that for the resulting MPC controller only the terminal cost and set change: Just the terminal controllers are different that are used to derive the aforementioned ingredients. The controller can therefore still be written as (5.12), only with adapted $\mathcal{X}_{\text{term}}$ and $l_N(e)$. The symbols $\mathcal{X}_{\text{term}}$ and $l_N(e)$ themselves are not changed between Sections 5.2 and 5.3 and within each section, the symbol always belongs to the respective controller. In the following, every variable that belongs to the eMPC-part of the terminal controller is denoted with a hat while for the actual MPC controller, the previous notation is kept.

5.3.1 Terminal Controller

As already introduced in the introduction to this section, this terminal controller uses eMPC together with feedback-linearization. Thus, the controller can be displayed as in Figures 5.2 and 5.3, with and without the compensation of $\mathfrak{R}\boldsymbol{M}$, respectively. The "Outer Loop Controller" is in this case accordingly an eMPC controller.

The eMPC controller is defined as follows: Consider the feedback-linearized system

$$\hat{e}_{k+1} = \hat{\phi}(\hat{e}_k, \hat{u}_k) = \begin{pmatrix} 1 & 0 & \delta & 0 \\ 0 & 1 & 0 & \delta \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \hat{e}_k + \begin{pmatrix} 0 \\ 0 \\ \delta \\ \delta \end{pmatrix} \hat{u}_k \tag{5.40}$$

under the yet-to-be-defined input constraints $\hat{u}_k \in \mathcal{U}_{\text{eMPC}}$ and with $\hat{e} \in \mathbb{R}^4$ such that

$$e = \begin{pmatrix} \hat{e} \\ e_5 \\ \alpha \end{pmatrix}. \tag{5.41}$$

> **Remark 29:** This is the Euler-forward discretization of a system consisting of two double integrators. For the linearized system, it would also be easy to calculate the exact discretization at the sampling times. This variant is however consistent with the formulation of the MPC controller as it is the feedback-linearized e_1-e_4-subsystem of Eq. (5.10).

The control $\mu_k = \mu(e_k)$ is the optimal piecewise linear and continuous solution to the optimization problem (i.e. the eMPC controller)

$$\begin{aligned} \underset{\hat{U}}{\text{minimize}} \quad & \sum_{k=0}^{\hat{N}} \left[\hat{e}_{t+k|t}^T \hat{Q}_{\text{e}} \hat{e}_{t+k|t} + \hat{u}_{t+k|t}^T \hat{Q}_{\text{u}} \hat{u}_{t+k|t} \right] + \hat{e}_{t+\hat{N}|t}^T P_{\text{eMPC}} \hat{e}_{t+\hat{N}|t} \\ \text{subject to} \quad & \hat{e}_{t+k+1|t} = \hat{\phi}(\hat{e}_{t+k|t}, \hat{u}_{t+k|t}), \\ & \hat{e}_{t+k|t} \in \hat{\mathcal{X}} \qquad k = 0, \dots, \hat{N}, \\ & \hat{u}_{t+k|t} \in \mathcal{U}_{\text{eMPC}} \; k = 0, \dots, \hat{N}, \\ & \hat{e}_{t+\hat{N}|t} \in \hat{\mathcal{X}}_t, \\ & \hat{e}_{t|t} = \hat{e}_t \end{aligned} \tag{5.42}$$

where the terminal set $\hat{\mathcal{X}}_t$ is chosen as described in Section 2.4 and $\hat{\mathcal{X}}$ is the set of allowed states.

With this, the terminal controller for the second MPC controller in this work can be given as

$$\begin{aligned} \boldsymbol{F} &= \boldsymbol{M}^{-1} \mathfrak{R}^{-1} \left(\mu_k + \mathfrak{R} \boldsymbol{f}(e_5) + \begin{pmatrix} 0 \\ 0 \\ -k_\omega e_5 \end{pmatrix} \right) \\ &= \boldsymbol{M}^{-1} \mathfrak{R}^{-1} \left(\mu_k + u_k^{\text{fb}} \right). \end{aligned} \tag{5.43}$$

In difference to the terminal controller for the previous section, the complete controller (5.43) completely separates the control of the rotational and the translational subsystems. The error in the angular velocity $e_5 \in \mathbb{R}$ is controlled by the linear feedback while the position and velocity in $\mathbb{R}^4$ are controlled by eMPC. For the simplicity of the notation,

in some cases μ_k is written where actually $\begin{pmatrix}\mu_k^T & 0\end{pmatrix}^T$ would be correct, i.e. where a zero would be appended to match the correct dimensions.

> **Remark 30:** The presented approach is not the first time when the combination of feedback linearization and MPC is proposed: In [76] (see also [77]), the system is feedback-linearized and the linearized system is controlled via MPC. This is different from the presented approach as in the referenced work, the system itself is controlled with linearization and MPC while in this work, only the terminal controller is feedback-linearized - the system itself and the actual MPC controller are treated as nonlinear. This way, potentially beneficial nonlinear dynamics can be exploited by the controller.

5.3.2 Terminal Cost

The stage cost for the eMPC is a sum of quadratic terms of inputs e and μ_k (notice that the error for the eMPC is the same as for the whole translational system) and the infinite-time cost is given as

$$l_{\text{eMPC}}(e_0) = \sum_{k=0}^{\hat{N}} \left[\hat{e}_k^T \hat{Q}_{\text{e}} \hat{e}_k + \mu_k^T \hat{Q}_{\text{u}} \mu_k\right] + \hat{e}_{\hat{N}}^T P_{\text{eMPC}} \hat{e}_N \geq \sum_{k=0}^{\infty} \hat{e}_k^T \hat{Q}_{\text{e}} \hat{e}_k + \mu_k^T \hat{Q}_{\text{u}} \mu_k. \tag{5.44}$$

The terminal cost matrix P_{eMPC} is chosen such that, for the chosen $\hat{Q}_{\text{e}}$ and $\hat{Q}_{\text{u}}$, it leads to a stabilizing terminal cost and consequentially the inequality holds. A simple selection is the cost of the infinite horizon optimal control problem with a suitable eMPC terminal set.

> **Remark 31:** Be careful with the notion of a terminal set here: As an MPC scheme (eMPC) is used "within" another MPC scheme, there are two terminal sets. In general, the one of the actual controller (5.12) is meant and otherwise it is specified explicitly that the one of the eMPC controller is meant as done here.

To make the following derivations, two assumptions are necessary: First of all, the matrix Q_u with which the control cost is calculated is diagonal and the first two indices are identical

$$Q_u = \begin{pmatrix} q_{u1} & 0 & 0 \\ 0 & q_{u1} & 0 \\ 0 & 0 & q_{u3} \end{pmatrix}. \tag{5.45}$$

Additionally, Q_e with which the state cost is calculated has no coupling between the position and the angular velocity, i.e.

$$Q_e = \begin{pmatrix} \hat{Q}_{\text{e}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & q_{e5} & 0 \\ \mathbf{0} & 0 & 0 \end{pmatrix} \tag{5.46}$$

where the cost of α is zero due to the micro-orbiting.

Before the full calculation, use $\mu_k = \begin{pmatrix}\mu_{1,k} & \mu_{2,k}\end{pmatrix}^T$ and observe that

$$\begin{aligned}
&\mu_k^T(\mathfrak{R}^{-1})^T\tilde{Q}_u\mathfrak{R}^{-1}u_k^{\text{fb}}\\
=&\mu_k^T(\mathfrak{R}^{-1})^T\tilde{Q}_u\left(\boldsymbol{f}(e_5)+\mathfrak{R}^{-1}\begin{pmatrix}0\\0\\-k_\omega e_5\end{pmatrix}\right)\\
=&[(\mu_{1,k}\cos\alpha+\mu_{2,k}\sin\alpha)r(-k_\omega e_5)+(-\mu_{1,k}\sin\alpha+\mu_{2,k}\cos\alpha)f(e_5)]m^2q_{u1}\\
\leq&\sqrt{\mu_{1,k}^2+\mu_{2,k}^2}(\|-rk_\omega e_5\|+\|f(e_5)\|)m^2q_{u1}\\
\leq&\sqrt{\mu_{1,max}^2+\mu_{2,max}^2}m^2q_{u1}(\|-rk_\omega e_5\|+\|(2\omega_0+e_5)e_5r\|)
\end{aligned}$$

where the relationship $a\cos x+b\sin x=\text{sgn}(a)\sqrt{a^2+b^2}\cos(x+\arctan\frac{-b}{a})$ has been used [78]. The variables $\mu_{1,max}$ and $\mu_{2,max}$ are introduced as the values for $\mu_{1,k}$ and $\mu_{2,k}$ that independently maximize the norm in their allowed set.

Keeping in mind that e_5 is time dependent with $e_5[k]=(1-k_\omega)^ke_5[0]$ follows further

$$\begin{aligned}
&\sum_{k=0}^{\infty}\mu_k^T(\mathfrak{R}^{-1})^T\tilde{Q}_u\mathfrak{R}^{-1}u_k^{\text{fb}}\\
\leq&\sqrt{\mu_{1,max}^2+\mu_{2,max}^2}rm^2q_{u1}\sum_{k=0}^{\infty}\left[(1-k_\omega)^{2k}e_5[0]^2+(k_\omega+2\|\omega_0\|)(1-k_\omega)^k\|e_5[0]\|\right] \qquad (5.47)\\
=&\sqrt{\mu_{1,max}^2+\mu_{2,max}^2}rm^2q_{u1}\left(\frac{e_5[0]^2}{1-(1-k_\omega)^2}+\frac{k_\omega+2\|\omega_0\|}{1-(1-k_\omega)}\|e_5[0]\|\right).
\end{aligned}$$

The term $(1-k_\omega)^k$ is always positive if $k_\omega<1$ which gives together with the requirement for stabilization possible values for k_ω as $k_\omega\in(0,1)$.

> **Remark 32:** It would be possible to choose $k_\omega\in(0,2)$, but in this case the series expansion is slightly more complicated. Also, the terminal set for the third input would become small which will become clearer in the next section.

Now calculate the cost as

$$\begin{aligned}
&\sum_{k=0}^{\infty}e_k^TQ_ee_k+\boldsymbol{F}^TQ_u\boldsymbol{F}\\
=&\sum_{k=0}^{\infty}e_k^TQ_ee_k+[\boldsymbol{M}^{-1}\mathfrak{R}^{-1}(\mu_k+u_k^{\text{fb}})]^TQ_u[\boldsymbol{M}^{-1}\mathfrak{R}^{-1}(\mu_k+u_k^{\text{fb}})]\\
=&\sum_{k=0}^{\infty}e_k^TQ_ee_k+[\mathfrak{R}^{-1}(\mu_k+u_k^{\text{fb}})]^T\underbrace{(\boldsymbol{M}^{-1})^TQ_u\boldsymbol{M}^{-1}}_{=:\bar{Q}_u}[\mathfrak{R}^{-1}(\mu_k+u_k^{\text{fb}})]\\
=&\sum_{k=0}^{\infty}e_k^TQ_ee_k+\mu_k^T(\mathfrak{R}^{-1})^T\tilde{Q}_u\mathfrak{R}^{-1}\mu_k\\
&\qquad+\sum_{k=0}^{\infty}2\mu_k^T(\mathfrak{R}^{-1})^T\tilde{Q}_u\mathfrak{R}^{-1}u_k^{\text{fb}}+(u_k^{\text{fb}})^T(\mathfrak{R}^{-1})^T\tilde{Q}_u\mathfrak{R}^{-1}u_k^{\text{fb}}
\end{aligned}$$

with Eq. (5.46) and Eq. (5.41)

$$= \sum_{k=0}^{\infty} \hat{e}^T \hat{Q}_\mathrm{e} \hat{e} + \mu_k^T (\mathfrak{R}^{-1})^T \tilde{Q}_u \mathfrak{R}^{-1} \mu_k + q_{e5} \sum_{k=0}^{\infty} e_5^2 + \sum_{k=0}^{\infty} 2\mu_k^T (\mathfrak{R}^{-1})^T \tilde{Q}_u \mathfrak{R}^{-1} u_k^\mathrm{fb} + (u_k^\mathrm{fb})^T (\mathfrak{R}^{-1})^T \tilde{Q}_u \mathfrak{R}^{-1} u_k^\mathrm{fb}$$

Choose $\hat{Q}_\mathrm{u} - [(\mathfrak{R}^{-1})^T \tilde{Q}_u \mathfrak{R}^{-1}] \succeq 0 \ \forall \alpha$ and it follows

$$\leq \sum_{k=0}^{\infty} \hat{e}^T \hat{Q}_\mathrm{e} \hat{e} + \mu_k^T \hat{Q}_\mathrm{u} \mu_k + q_{e5} \sum_{k=0}^{\infty} e_5^2 + 2 \sum_{k=0}^{\infty} \mu_k^T (\mathfrak{R}^{-1})^T \tilde{Q}_u \mathfrak{R}^{-1} u_k^\mathrm{fb} + \sum_{k=0}^{\infty} (u_k^\mathrm{fb})^T (\mathfrak{R}^{-1})^T \tilde{Q}_u \mathfrak{R}^{-1} u_k^\mathrm{fb}$$

and using Eq. (5.44),

$$\leq l_\mathrm{eMPC}(e_0) + q_{e5} \sum_{k=0}^{\infty} e_5^2 + 2 \sum_{k=0}^{\infty} \mu_k^T (\mathfrak{R}^{-1})^T \tilde{Q}_u \mathfrak{R}^{-1} u_k^\mathrm{fb} + \sum_{k=0}^{\infty} (u_k^\mathrm{fb})^T (\mathfrak{R}^{-1})^T \tilde{Q}_u \mathfrak{R}^{-1} u_k^\mathrm{fb}$$

which yields with Eq. (5.47) and Eq. (5.43)

$$\begin{aligned} \leq & l_\mathrm{eMPC}(e_0) + q_{e5} \sum_{k=0}^{\infty} e_5^2 \\ & + 2\sqrt{\mu_{1,max}^2 + \mu_{2,max}^2} r m^2 q_{u1} \left(\frac{e_5[0]^2}{1-(1-k_\omega)^2} + \frac{k_\omega + 2\|\omega_0\|}{k_\omega} \|e_5[0]\| \right) \\ & + \sum_{k=0}^{\infty} (f(e_5) + \mathfrak{R}^{-1}(-k_\omega e_5))^T \tilde{Q}_u (f(e_5) + \mathfrak{R}^{-1}(-k_\omega e_5)). \end{aligned}$$

Finally applying the previous result Eq. (5.19) and using the series expansion equivalently to Eq. (5.23) gives

$$\begin{aligned} \leq & l_\mathrm{eMPC}(e_0) + q_{e5} \sum_{k=0}^{\infty} e_5^2 \\ & + 2\sqrt{\mu_{1,max}^2 + \mu_{2,max}^2} r m^2 q_{u1} \left(\frac{e_5[0]^2}{1-(1-k_\omega)^2} + \frac{k_\omega + 2\|\omega_0\|}{k_\omega} \|e_5[0]\| \right) \\ & + 2\|\tilde{Q}_u\|_2 \left(e_0^T P_K e_0 + \frac{(2\omega_0 r)^2 e_5^2[0]}{1-(1-k_\omega)^2} + \frac{4\omega_0 r e_5^3[0]}{1-(1-k_\omega)^3} + \frac{r^2 e_5^4[0]}{1-(1-k_\omega)^4} \right) \\ =: & \ l_N(e_0) \end{aligned} \tag{5.48}$$

which concludes the calculation.

The derived terminal cost $l_N(e_0)$ is then piecewise defined due to the eMPC scheme involved. As it is piecewise quadratic and continuous, it anyhow forms a valid cost function. The terminal cost could be used directly, but then likely a common and well-

known problem of eMPC arises in practice: Checking the current state for the set in which it lies has proven to be computationally expensive and this operation would potentially be performed frequently by the solver.

A better idea is to bound the eMPC cost with a quadratic or higher-order polynomial from above. This is possible as the origin lies within the interior of one of the sets[1]. In combination with the piecewise quadratic nature and continuity of the eMPC cost, the bound thus becomes possible. This way, the solver for the actual MPC controller can avoid unnecessary computational overhead.

5.3.3 Terminal Set

As for the first MPC controller, the terminal cost is only valid in the region where the input constraints are not violated by the terminal controller - only there the system will evolve as predicted. The difference lies within the ability of the eMPC part of the terminal controller to handle the constraints better which results in a bigger terminal set.

The region of recursive feasibility and stability of the controller is now derived as follows: First of all the inputs necessary for nominally following the trajectory are taken into account. In the second step, u_k^{fb} is considered, i.e. the control of the angular velocity and the feedback linearization. This is done parametrically first and when the parameter is fixed, it leads to input constraints for the eMPC. By calculating the eMPC controller in the final step, on the one hand the value of $l_{\text{eMPC}}(e_0)$ becomes explicit and on the other hand the state constraints for e_1 to e_4 are determined.

Feedback-linearizing control Assuming dynamic feasibility of the trajectory in the sense of Definition 2, the necessary inputs for nominally following the trajectory satisfy $\tilde{u} \in \tilde{\mathcal{U}}$ and then the remaining inputs to control the error dynamics becomes $\bar{\mathcal{U}} = \boldsymbol{M}^{-1}\mathcal{U} \ominus \boldsymbol{M}^{-1}\tilde{\mathcal{U}}$.

> **Remark 33:** The factor $\boldsymbol{M}^{-1}$ is applied so that the calculation can be performed in the "half-compensated" space with $\breve{u}$, ref. Eq. (4.15).

In order to include u_k^{fb} into the considerations, first u_3 is taken into account. As follows from the system dynamics (Eq. (5.10)) and the control law (Eq. (5.43)) (ref. also Figure 5.3), u_3 forms a closed, linear control loop with a single integrator to e_5. With this subsystem

$$e_5[k+1] = e_5[k] + u_3, \tag{5.49}$$

the linear feedback law

$$u_3 = -k_\omega e_5 \tag{5.50}$$

[1] Within the region around the origin where the infinite-horizon optimal controller satisfies the constraints, the infinite-horizon optimal controller is also trivially optimal. Accordingly, the solution of the eMPC controller is equal to the infinite-horizon controller in said region. Due to the assumption of $\mathbf{0} \in \text{Int}(\mathcal{X})$, $\mathbf{0} \in \text{Int}(\mathcal{U})$ and the linearity of the infinite-horizon optimal controller, such a set always exists.

and the yet parametric, soon explicit bounds

$$-u_{3,max} \leq u_3 \leq u_{3,max} \tag{5.51}$$

the region of feasibility of the control law can be given as

$$\mathcal{X}_{t,2} = \left\{ e_5 \in \mathbb{R} \middle| -\frac{1}{k_\omega} u_{3,max} \leq e_5 \leq \frac{1}{k_\omega} u_{3,max} \right\}. \tag{5.52}$$

At the same time, this is also the region of attraction and a control invariant set which can be easily shown using a quadratic Lyapunov function.

Determination of $u_{3,max}$ and the input set for eMPC As the feedback linearization term only depends on e_5, the previous result allows to draw conclusions also on the control action necessary for the feedback-linearizing part of the controller. With Equations (4.22), (5.14) and (5.52), the set of inputs necessary for feedback-linearization can be calculated as

$$\mathcal{U}_{fblin} = \left\{ u \in \mathbb{R}^3 \middle| u_3 = 0, \|u\| \leq \left(\frac{1}{k_\omega} u_{3,max} + 2\omega_0 \right) \frac{1}{k_\omega} u_{3,max} r \right\} \tag{5.53}$$

and the set of remaining controls for the linear system becomes

$$\mathcal{U}_{linear} = \bar{\mathcal{U}} \ominus \mathcal{U}_{fblin}. \tag{5.54}$$

As established by Section 2.1, the set $\mathcal{U}_{linear}$ can be written as a direct parametrization of $u_{3,max}$ and becomes smaller with an increasing $u_{3,max}$.

A full sketch of the different sets is shown in Figure 5.4: First of all, a part of the constraining set $\boldsymbol{M}^{-1}\mathcal{U}$ is removed by taking the dynamic feasibility into account. From this the term $\mathcal{U}_{fblin}$ is subtracted that depends on $u_{3,max}$.

The set $\mathcal{U}_{linear}$ remains available for the linear controller (5.50) and the eMPC controller. However, two more criteria need to be met by the final input bounds: First of all, the linear controller and the eMPC act independently from each other while using a shared input space. Constraint satisfaction can be ensured by defining two subspaces $\mathcal{U}_\omega$ and $\mathcal{U}_{\text{eMPC}}$ and ensuring $\mathcal{U}_\omega \times \mathcal{U}_{\text{eMPC}} \subseteq \mathcal{U}_{linear}$. The interval constraint $\mathcal{U}_\omega$ is already given by Eq. (5.51).

The second additional constraint emerges as all calculations have so far been performed in the projection $\boldsymbol{M}^{-1}\mathcal{U}$. The actual input u instead needs to satisfy the constraints $u \in \mathfrak{R}^{-1}\boldsymbol{M}^{-1}\mathcal{U}$, i.e. the compensation of $\mathfrak{R}$ has so far been left out.

As stated in Section 2.1, however, if $\mathcal{U}_{\text{eMPC}}$ is a circular cross-section of $\mathbb{R}^3$ with

$$\mathcal{U}_{\text{eMPC}} = \left\{ u \in \mathbb{R}^3 \middle| u^T \begin{pmatrix} \boldsymbol{I}_{2\times 2} & \boldsymbol{0} \\ \boldsymbol{0} & 0 \end{pmatrix} u \leq r_{\text{eMPC}} \right\},$$

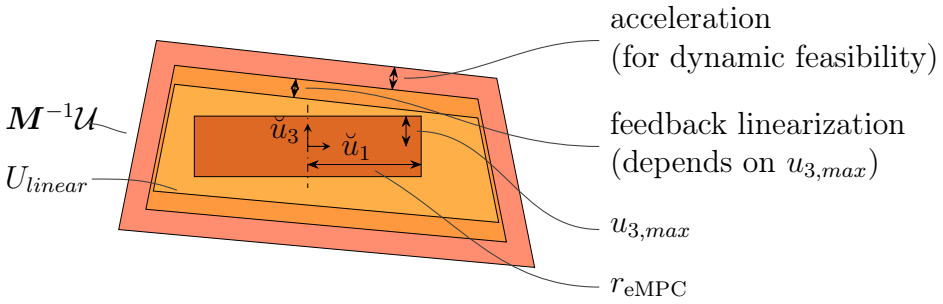


Figure 5.4: Sketch of the different sets in the input domain used to derive the terminal set. Shown is a projection on the $\tilde{u}_1$-$\tilde{u}_3$-axis, with $\tilde{u}$ from Eq. (4.16).

it fulfills

$$\hat{\mathcal{U}} \oplus \mathcal{U}_{fblin} \oplus (\mathcal{U}_\omega \times \mathcal{U}_{\text{eMPC}}) \in \boldsymbol{M}^{-1}\mathcal{U}$$

for every $\mathcal{U}_{\text{eMPC}}$ with

$$\mathcal{U}_\omega \times \mathcal{U}_{\text{eMPC}} \subseteq \mathcal{U}_{linear}.$$

Additionally, the set is rotationally invariant and therefore $v \in \mathcal{U}_{\text{eMPC}} \Rightarrow \mathfrak{R}v \in \mathcal{U}_{\text{eMPC}}$, i.e. the compensation of $\mathfrak{R}$ is always ensured. In other words, in this case the satisfaction of the input constraints is ensured.

The remaining task is now to fix a "good" value for $u_{3,max}$. Based on this, a "good" set $\mathcal{U}_\omega \times \mathcal{U}_{\text{eMPC}}$ can be found and with this, the terminal set $\mathcal{X}_{\text{term}}$ can be calculated. The most direct answer to what makes good selections is that the volume of $\mathcal{X}_{\text{term}}$ should be as big as possible. This is however not a trivial task as there is no direct method of computation for recursively feasible sets which is parametrizable by $u_{3,max}$ and r_{eMPC}. Therefore, the parameters are calculated that maximize the volume of $\mathcal{U}_\omega \times \mathcal{U}_{\text{eMPC}}$ which gives at least some sort of optimality. This is achieved by solving an optimization problem in adaption of the problems discussed in [42, 8.4.2]: Write the individual constraints of $\mathcal{U}_{linear}$ as $a_i x \leq b_i(u_{3,max})$. The largest volume inscribed cylinder is

$$\begin{aligned} &\underset{u_{3,max},\, r_{\text{eMPC}}}{\text{maximize}} && 2\log r_{\text{eMPC}} + \log 2u_{3,max} \\ &\text{subject to} && \left\| a_i \begin{pmatrix} r_{\text{eMPC}} & 0 \\ 0 & r_{\text{eMPC}} \\ 0 & 0 \end{pmatrix} \right\|_2 + a_i \begin{pmatrix} 0 \\ 0 \\ \pm u_{3,max} \end{pmatrix} \leq b_i(u_{3,max}) \end{aligned} \tag{5.55}$$

It can be checked easily by direct calculation that the objective function is the logarithm of the volume, leaving out the unnecessary constant $\log \pi$.

The formulation would be entirely convex if the bounds of $\mathcal{U}_{linear}$ were constant. As this is not the case, a nonlinear solver needs to be utilized.

In a last step, to be allowed as a constraint for eMPC, the constraint must be a polytope. Therefore, $\mathcal{U}_{\text{eMPC}}$ needs to be approximated from inside as a (potentially regular) polytope.

Calculation of $\mathcal{X}_{\text{term}}$ Once the input sets for the different controllers have been fixed, the eMPC can finally be calculated. By solving problem (5.42) as described in [59], two results can be computed: First of all, the control law μ_k, but also the cost function $l_{\text{eMPC}}(\hat{e})$ that was not explicitly known so far. The control law is a piecewise defined linear law which is defined only within the union of its partition regions and undefined outside, i.e.

$$\mathcal{X}_{t,1} = \{\hat{e} \in \mathbb{R}^4 | \mu(e_k) \neq \varnothing\}. \tag{5.56}$$

For $\mathcal{X}_{t,1}$ it is however guaranteed to be stabilizing. Combined with $\mathcal{X}_{t,2}$ from Eq. (5.52), a recursively feasible region of attraction can be calculated and chosen as the terminal set

$$\mathcal{X}_{\text{term}} = \mathcal{X}_{t,1} \times \mathcal{X}_{t,2}. \tag{5.57}$$

The main feature of this terminal controller and thus terminal set is that its size depends on the eMPC horizon $\hat{N}$: With an increasing $\hat{N}$, the set becomes bigger. But as the eMPC calculations are preformed offline, this does not reflect in a higher computational effort of the resulting MPC scheme.

For the controller in Section 5.2, the feedback-linearized system could only be stabilized within the feasible range of a linear feedback control law. The eMPC controller in turn gets rid of this and extends its (recursively) feasible set beyond the limitations of a linear control.

> **Remark 34:** In fact, the eMPC uses a linear control law for the derivation of its own terminal set. The expansion of this region stems from the utilization of a nonlinear control law.

> **Remark 35:** The terminal set can not be increased to infinity, or in most cases not even the complete feasible set: On the one side, the achievable size is limited by a practical constraint of eMPC. The computational time increases exponentially for large dimensions and horizons. Therefore, only the choice of a relatively limited $\hat{N}$ is sensible.
>
> On the other side, the state constraints in eMPC are formulated as convex sets. Maps with obstacles in between are therefore not permissible.
>
> After all, however, the main point still stands that this terminal controller can extend the terminal set into regions that input saturations previously prohibited. A comparison how this looks like in practice can be found in Chapter 6.

This closes the derivation of the MPC controller. The only thing left is to prove its stability which takes place in the following section. As an intermediate result, however, the stability of the terminal controller is shown before.

> **Lemma 3:** The controller (5.43) stabilizes the discrete-time orbit dynamics (5.10) asymptotically for all states $e \in \mathcal{X}_{\text{term}}$.

Proof The terminal set $\mathcal{X}_{t,2}$ has been derived such that the linear control law is possible for all $e_5 \in \mathcal{X}_{t,2}$ and its asymptotic stability follows from the requirement $k_\omega \in (0,1)$. The latter can be easily shown by a quadratic Lyapunov function.

As the feedback-linearizing term only depends on e_5 by Eq. (4.22), on the one side its convergence to zero is then also guaranteed. But more importantly its feasibility is guaranteed by the inclusion of $\mathcal{U}_{fblin}$ into the calculations of this section.

Therefore, e_5 is already stable and the evolution of the states e_1 to e_4 is described by Eq. (5.40), i.e. a linear system. For this system, $\mathcal{U}_{\text{eMPC}}$ is a valid set of inputs with which the overall input constraints are satisfied irrespective of how the other parts of the controller act.

Consequently, the eMPC controller can be formulated in a stabilizing way and its stability can be shown following standard results, ref. [53, Thm. 5.3.2]. The other part of the terminal set, $\mathcal{X}_{t,1}$, is hence asymptotically stable which follows immediately. The feasibility of the controller as a whole within $\mathcal{X}_{\text{term}}$ has been shown in the derivations of this section.■

Before the section ends, two remarks shall be given that comment on the derivation of the terminal set and why it is possible. After that, the next section follows with the proof of stability of the full MPC controller.

> **Remark 36:** The system structure helps a lot to find a suitable terminal set. First of all, the error e_5 of the angular velocity can be controlled by a one-dimensional control law and also the closed loop is linear. Even more, the feedback linearizing part only depends on e_5. Therefore a simple parameterization of the set $\mathcal{U}_{linear}$ becomes possible.

> **Remark 37:** For this system it is possible to break the nonlinearities apart: On the one hand, there is the nonlinear feedback-linearized term $f(e_5)$, but also the rotation $\mathfrak{R}$ is nonlinear and compensated. As $\mathfrak{R}$ is a rotation matrix and therefore has a lot of known and desirable properties (s.a. $\|\mathfrak{R}\| = 1$ used in the derivation of the cost function), the derivations simplify a lot. It is not quite clear if or how this generalizes on systems with different structures and nonlinearities.

5.3.4 Proof of Stability

After the derivation of the terminal ingredients for the MPC controller, its stability properties can be given.

> **Proposition 4:** Consider the system with dynamics (5.10), state constraints $e \in \mathcal{X}$ and input constraints $\boldsymbol{F} \in \mathcal{U}$ where $\mathcal{U}$ is given in Definition 9. Assume the trajectory χ is dynamically feasible as of Definition 2. Let the system be controlled by the MPC controller (5.12) where Q_e and Q_u satisfy Equations (5.45) and (5.46), l_N is given by Eq. (5.48) and $\mathcal{X}_{\text{term}}$ by Section 5.3.3. Then, the system is asymptotically stable for all states that allow to solve the optimization problem (5.12).

Proof The proof for this controller is performed analogous to the one for the previous one:

Once again, as the stage cost l is a quadratic function, it is continuous, zero at the origin and lower-bounded by a class κ_∞ function for all e_k and u_k.

The terminal cost l_N satisfies the properties $l_N(0) = 0$, $l_N(e) > 0 \; \forall e \neq 0$ as clear from Eq. (5.48) and is continuous. The definitions of $\mathcal{X}$, $\mathcal{U}$ and $\mathcal{X}_{\text{term}}$ ensure that they are all closed and contain zero in their interior.

As $\mathcal{X}_{\text{term}}$ was constructed also in this case such that a controller exists that stabilizes the system for all $e \in \mathcal{X}_{\text{term}}$ (ref. Lemma 3), the set is control invariant. For the same control law, the terminal cost l_N is constructed in Eq. (5.48) such it bounds the infinite-horizon cost of the controller from above. This directly ensures also for this controller that

$$l_N(\Phi(e_0, \bar{u}_0 + \hat{u}_0)) - l_N(e_0) + e_0^T Q_e e_0 + \bar{u}_0^T Q_u \bar{u}_0 \leq 0 \tag{5.58}$$

as the inequality holds for every single time step. Therefore, the proof follows immediately from Theorem 2. ∎

This closes the derivation of the controllers. Three different controllers have been shown that all use micro-orbiting as a strategy to overcome the nonholonomic constraints which follow from the actuator failures. One controller is a feedback controller and based on feedback-linearization, and two are based on MPC. The stability properties of the first MPC controller were derived using a terminal controller that feedback-linearized the system locally and then stabilized the linearized system with a linear feedback controller. The stability properties of the second MPC controller were derived using a terminal controller that also feedback-linearized the system locally, but then used an eMPC controller for the linearized system. As will be shown next, this is advantageous compared to the first MPC controller as it leads to much larger terminal sets.

Chapter 6

Evaluation

In the previous chapter, three different control laws have been derived of which one is a feedback-based controller that is based on feedback linearization (ref. Theorem 4). This control law will be denoted as κ_{fb} in the following.

The other two control laws are implicitly defined as optimization problems (ref. Eq. (5.12)) and their optimal solutions for control outputs that are called κ_1 and κ_2, respectively. The stability guarantees for κ_1 were derived using a terminal controller that feedback-linearizes the system and controls the linearized system with a linear controller in Section 5.2. For κ_2, the terminal controller was extended by an eMPC scheme for the linearized system in Section 5.3.

All three controllers are using micro-orbiting as strategy to overcome the nonholonomic constraints that appear for the error cases C2 and C3.

The remaining part of this work consists of a simulation of the control laws and the subsequent discussion of the results. Also the advantages and disadvantages of micro-orbiting will be considered.

The controllers were simulated in Python and implemented using `casadi` [79]. The eMPC controller for κ_2 was calculated using the `pympc` library [80]. The sampling time of the environment and the controllers were chosen differently by a factor of 10 and additional noise was added. The parameters used in the simulation are given in Appendix B and the code will soon be made available on GitHub under `https://github.com/raphael-jms/micro-orbiting`.

6.1 Terminal Sets of the MPC Controllers

For the simulation of the controllers, numerical values for the terminal sets of the MPC controllers are necessary to calculate. For κ_1, the derivation is as described in Eq. (5.12). κ_2 needs additional parameters, namely the eMPC state constraint that was chosen to be

a box constraint $\begin{pmatrix}-5.0 & -5.0 & -1.5 & -1.5\end{pmatrix}^T \leq \hat{e} \leq \begin{pmatrix}5.0 & 5.0 & 1.5 & 1.5\end{pmatrix}^T$[1]. The eMPC horizon was chosen to $\hat{N} = 50$. Based on the eMPC control law, the terminal set of the control law κ_2 was calculated.

The terminal sets for both MPC controllers can be seen in Figure 6.1. It is clear from previous sections that the terminal sets are multidimensional and therefore only projections or slices through the sets can be plotted. As the x- and y-direction are independent from each other, it is sensible to regard position and velocity with e_1 and e_3 together and equivalent results follow for e_2 and e_4.

As to be expected, the terminal set for κ_2 is much larger than the one for κ_1. In fact, the terminal set for κ_1 is only marginally bigger than the set $\{0\}$ while the set for κ_2 stretches over 5 m in both directions along the e_1-axis. The latter stems from the design with the eMPC in the terminal controller. Also when it comes to e_5, κ_2 is advantageous with e_5 within ± 0.129 rad for all $e \in \mathcal{X}_{\text{term}}$ (as it is an interval constraint). For κ_1 in turn, the maximal and minimal values (as it is an ellipsoidal set) is only ± 0.006 rad. For e_1 to e_4, the difference in the terminal controllers could be made responsible for the results, but e_5 is controlled in both terminal controllers simply with a linear controller. This allows for the conclusion that additionally the derivation of polytopic constraints is advantageous over ellipsoidal ones for this system.

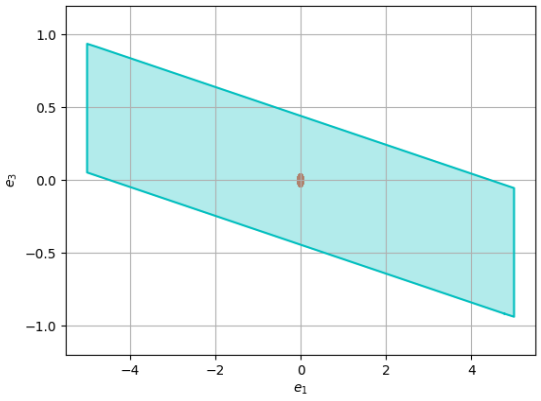


Figure 6.1: Terminal sets for κ_1 in red and κ_2 in cyan. Printed is a slice through the sets where $e_2 = e_4 = e_5 = 0$.

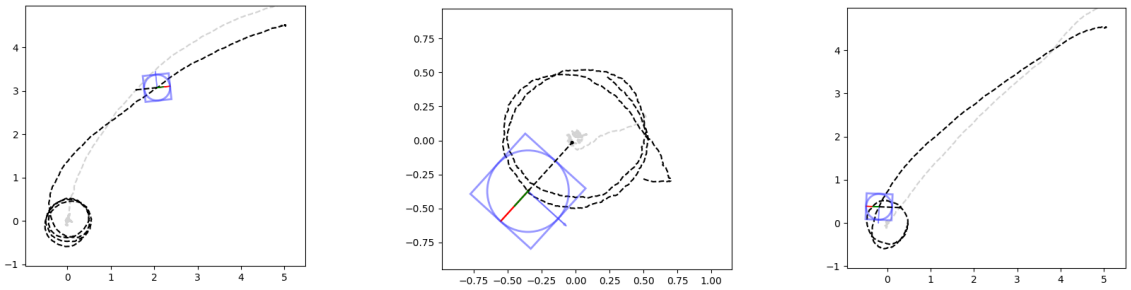
Concerning the computational performance of the controllers, there are little differences. Both controllers κ_1 and κ_2 solve the optimization problem by average in 0.047 1 s and 0.030 4 s, respectively. The numbers come from the point stabilization problem that is described next, but there is little difference to the solution of a tracking problem. The solvers are therefore faster than the controllers' sampling time of 0.1 s. The simulations were made on a HP laptop with an AMD Ryzen™ 7 5700U processor.

[1] The velocity constraints on $\hat{e}_3$ and $\hat{e}_4$ do not really have an effect for the resulting terminal constraint of κ_2 as the input constraints in combination with the constraints on the position error $\hat{e}_1$ and $\hat{e}_2$ forbids these velocities anyway.

6.2 Point Stabilization

Next, some simulations are presented, one example for point stabilization and one for trajectory tracking. They do not explicitly show the transition from normal operation to post-failure recovery but instead start directly at a state in the recovery mode: On a general level, the presented controllers are designed to act in the "recovery" phase after (1) the failure occurred, (2) it was noticed and (3) the new controller was prepared for action. Predicting the trajectory before the recovery phase during the three mentioned stadiums is hard as it depends on a lot of parameters, some of them implementation dependent. Therefore it is difficult to predict with which state the recovery phase will start and the controllers should stabilize the system for potentially large errors. If however, as attempted in this work, the controller in the recovery phase stabilizes the system for all possible states, safety is ensured no matter what happens before the recovery phase. Consequently, only simulations for the recovery phase are necessary.

The first scenario is a point stabilization at the origin. The failed actuators are $F_{3,1} = F_{4,1} = F_{\max}$, i.e. the robot experiences a thrust into its local y-axis with the maximal amount possible. The path that the robot takes with the different controllers is shown in Figure 6.2 and the corresponding states and inputs in Figure 6.3.



(a) Using κ_{fb} (b) Using κ_1 (c) Using κ_2

Figure 6.2: Path of the different controllers for point stabilization in the x-y-plane. The axis units are in m. κ_{fb} and κ_2 start from $c = \begin{pmatrix} 5 & 5 & 0.1 & 0 & 0 & 0 \end{pmatrix}^T$ and κ_1 starts from $c = \begin{pmatrix} 0.5 & 0.2 & 0.1 & -0.1 & 0.5 & 0 \end{pmatrix}^T$ as the problem is infeasible with the other starting point. Note that the path is jittery due to the noise, not because of performance issues.

The first plot shows the robots in blue and the two dashed paths are in black the path of the robot and in grey the path of the center of the orbit. The center of the orbit is marked with a black dot and a dashed line connects the center of the robot with the center of the orbit.

When comparing the paths between each other, first of all it can be seen that all controllers stabilize the system. κ_{fb} and κ_2 start from the same position far away from the desired setpoint while κ_1 starts with fewer error. This is because of the smaller feasible region, κ_1 would simply not converge at the starting point of the other controllers. κ_{fb} and κ_2 in turn have a large feasible region.

Notice that the path is jittery, which becomes - due to the zoom level - mostly visible for Figure 6.2b, but actually holds for all controllers. This is not due to bad controller

performance, but because the strong uniform noise in a maximal range over a second[2] of $\pm 0.25\,\mathrm{m}$. Simulations with less aggressive noise parameters resulted in smooth lines and the convergence along a perfect circle.

Looking at the paths instead of the state trajectories gives insights on how the robot behaves under micro-orbiting: The idea of micro-orbiting is to keep the system in a region around the trajectory or a stationary point and thus preserving safety and a certain amount of control. In fact, the robot stays within a r-radius around the orbit center point. And once the center point has converged, it stays in the same radius around the setpoint. Also, as previously mentioned in Remark 18, the robot's path only visually resembles an orbital pattern once the error has been sufficiently minimized.

Regarding the state evolutions and inputs in Figure 6.3 instead of the path gives similar results for the convergence and performance. The blue line shows the evolution of the orbit center as this is the controlled value. For the inputs of the MPC controllers the full input is shown in blue, the input to nominally follow the trajectory in black, the input calculated by the MPC in red and the compensation to set the virtual fault in grey. The last three do not all individually satisfy the input constraints of $[-3.5\,\mathrm{N}, 0\,\mathrm{N}]$ for F_x and F_y, but their sum in blue does. One can also see that the input constraints are satisfied for κ_{fb}, i.e. all controllers satisfy the input constraints.

6.3 Trajectory Tracking

As only the two MPC controllers are suitable for tracking, Figure 6.4 only shows two controllers that track a circular trajectory of radius 2 m. The parameters for the simulation remained identical to the first one regarding noise and failed actuators. The trajectory is shown in dark grey and the path of the center trajectory in light grey.

Here once again the advantage of κ_2 over κ_1 is clear as κ_2 has a much larger feasible set. Besides of this, both controllers let the center of the orbit converge to the trajectory. The robot itself is orbiting around the trajectory which, together with the motion of the trajectory, leads to the trajectories shown in Figures 6.4a and 6.4b. At first glance, they may seem unexpected, but it is just the overlay of two steady rotations, i.e. the rotation of the trajectory and the rotation of the micro-orbit. If the trajectory was e.g. instead a linear trajectory with constant velocity, the path of the robot would be come a spiral.

From Figures 6.4c and 6.4d, a similar statement is possible. One can see that the last state α is unbounded and increasing with a constant rate as for the point stabilization. This is not a problem and was to be expected as the idea of micro-orbiting is to keep the system at a constant angular velocity. This directly necessitates an increasing α. The controlled states are quickly converging towards the desired value and satisfy the input constraints.

[2]The noise was added in every time step sampled from a uniform distribution of $\delta_{\mathrm{env}}[-0.25\,\mathrm{m}, 0.25\,\mathrm{m}]$. The multiplication by δ_{env} ensures that the cumulative received noise over time remains independent of the sampling time. When summed over a one-second interval, the resulting distribution of course deviates from uniformity. The minimum and maximum possible values remain $[-0.25\,\mathrm{m}, 0.25\,\mathrm{m}]$.

6.4 Results and Discussion

The simulation results show that micro-orbiting can in fact keep the system in an orbit around the trajectory. The controllability issues and nonholonomic constraints could successfully be circumvented and a control strategy be found that keeps the system in a safe region around the trajectory. Although the simulation only considers one specific failure case, the preceding analysis in Section 3.1.3 showed that the model that was used can represent the system for all failure cases C2 and C3. Accordingly, its ability to control the system until at least three actuator of failures was demonstrated.

It has been shown that all of the three controllers could successfully control the system, which showcases again that micro-orbiting is a general strategy to circumvent the problems with controllability and nonholonomicity. It is not bound to a certain controller structure and can be implemented with different approaches.

Comparing the two MPC controllers, κ_2 has a better performance that κ_1: For κ_1, there are starting states within the range of $2r$ for which the optimization problem is not feasible. A state in this range is possible to occur, wherefore this controller in its current form might not be suitable for ensuring safety - even though, if it is possible to increase the horizon, it could become good enough.

But even if this is possible, κ_2 outperforms κ_1 as in the former controller this is not a problem at all: As the terminal set can be increased by choosing a larger horizon for the eMPC (terminal) controller, the terminal set for κ_2 can be found, within the bounds of feasibility, practically always as big as desired. This has another advantage as due to a larger $\mathcal{X}_{\text{term}}$ the feasible set is larger and therefore the horizon of κ_2 itself can be smaller. This in turn leads to a smaller online-optimization problem with lower computational costs and requirements.

Further comparing κ_{fb} and κ_2 in Figure 6.2 shows that the orbit of κ_2 converges to zero in almost a straight line while κ_{fb} does not take the direct path. As designed, this occurs because the MPC controller operates in an optimal manner, which is made possible by including more knowledge about the system into the controller.

Concerning the general differences between MPC and feedback controllers, κ_2 allows for obstacle avoidance if a suitable set $\mathcal{X}$ of allowed states is chosen. κ_{fb} is not capable of doing this, but is on the other hand of course much faster to compute, has lower computational requirements and takes less storage space.

A last difference between κ_{fb} and κ_2 is that κ_{fb} does not allow for trajectory tracking. It is debatable whether this is a problem: On the one hand, trajectory tracking might not be necessary and only convergence towards a safe point needed. On the other hand, it is not said that there is a clear path out of a dangerous zone and even less that it is the path that κ_{fb} will take. Also it can be an advantage in operation if a damaged system can autonomously return to a place where it can be repaired.

Concerning the terminal controller, another observation can be made: Usually both feedback linearization as well as (e)MPC require a good model of the system as otherwise, robustability issues can occur. The controller could however control the systems even

under strong noise which highlights one of the features of the terminal controller: Its purpose is not to be applied directly, but to derive stability conditions from it. Thus, potential robustability issues do not appear.

One last thought is that the derivations for the MPC controllers simply derived a terminal cost and a terminal set. Therefore, it should be possible to extend the results from this work with results on robust MPC. Albeit the simulations have already shown promising results even under presence of strong noise, this could be advantageous: Depending on how (in)accurate the estimation of the actuator error is, steady errors of the system model can appear justifying the implementation of further methods for robustification.

On a more general level, micro-orbiting could seem like it might suffer from two problems at first sight: The size of the radius in space-constrained environments and fuel consumption. For the first concern, the radius can be chosen freely (ref. Eq. (4.6)) and theoretically made infinitely small. This is a question of tuning as a too small radius can lead to issues with the reachability of the terminal set. The simulated freeflyer, however, emulates the Astrobee well in terms of dynamical behaviour and uses a micro-orbit that measures 1 m in diameter. This is enough for the application for example in the ISS.

The matter of fuel consumption needs consideration but does not present a significant limitation: The primary fuel expenditure is already attributable to the failed actuators - an unavoidable consequence of the failure. Furthermore, this approach deliberately addresses actuator failure scenarios, which are by definition exceptional circumstances rather than nominal operations. The system is designed for temporary operation in this mode, either to facilitate a controlled return to a safe position or, in the case of satellites, to execute a de-orbiting maneuver. As such, the additional fuel requirements for micro-orbiting are justified within the context of emergency response operations and the fuel storage can be assumed to be sufficient for this short-term operation.

6.5 Generalization to Different Systems

As a last part of this thesis, an outlook on possible extensions is given: As already intended by design, the proposed control system is easily generalizable to different target systems and requirements. Of especial interest is the generalization of the terminal controller using eMPC which is considered in Section 6.5.4.

6.5.1 Generalization on the 3D Case

The derivations in this thesis were solely made for a two-dimensional model as this is a model of the experimental platform here at KTH. The ultimate goal is however that the results should be applied on actual three-dimensional space robots. For this, the model only needs to be adjusted slightly:

Principally, the natural formulation of micro-orbiting in 3D consists of a direct transfer from the 2D case: It also involves controlling a point on the line segment starting at the center of the robot and directing along the virtual uncontrollable force. Equivalently, this

point is in the distance r from the center, where r is again a suitably chosen radius of the micro-orbit.

The only open question is how this orbit should be placed *around* the point which is tracked: In the 2D-case, there is only one option as an orbit is, geometrically speaking, a circle and there is only one way to place the circle around the target point. The orbit is still a circle in the 3D-case, but a whole sphere exists around the tracked point and the orbit could be any circle (with the same radius and center point as the sphere) that is contained in the spere.

The decision which orbit is the best one to choose is not obvious: From a conceptual perspective, the easiest way of thinking of it is to chose an orbit in a plane that is perpendicular to the direction of the trajectory. This way, the movement on the orbit and the movement along the trajectory can happen independently from each other. On the other hand, reachability or energy efficiency might make a different selection the better choice.

For a controller s.a. the first feedback-linearizing controller, it could be possible to refrain from choosing a specific orbit at all. But for MPC-based controllers, the selection is important as it determines the allocation of the control during the calculation of the terminal sets.

6.5.2 Systems With Non-symmetric Actuators and Other Systems

The freeflyers that were used in this work have a symmetric structure where two thrusters are placed on each side, and each one has the same maximal power. This does not need to be the case for more general systems, and actually it is not even for the Astrobee: This system has a different amount of thrusters on the x-axis than on the other two ones.

This is however not a problem at all, as micro-orbiting as a strategy is designed to deal with the non-symmetrical errors that appear because of actuator failures and that cannot be compensated for. When moving to unsymmetrical systems, the only statement that is lost is that the system can be guaranteed to be stabilized for at least three failures. The number of recoverable actuators is likely to be different and for a similar statement, simply analysis of the new system is necessary in adaption of Chapter 3.

The idea of micro-orbiting can also be extended to systems outside of space: Essentially, it uses the properties of the group SO(2)/SO(3) to its advantage where a continuous rotation around a point, despite an ever-increasing angle, results in a constant distance to the point. Thus, principally all bodies in SO(3) allow for such a movement given suitable actuation.

Extending the concept even further leads to the question if completely different systems, e.g. chemical plants, can also be controlled using a kind of micro-orbiting. This might be possible, but needs more domain knowledge: Only if a limit-cycle around the desired setpoint can be found, the advantages of orbiting come into play.

6.5.3 Control for Non-constant Errors

Micro-orbiting in this work's formulation relies on constant errors resulting from the actuator failures. As discussed earlier in this chapter, it is however possible to robustify the MPC controllers e.g. using tube-based MPC. In this case, an estimation error of the actuator error is admissible. In a range limited by the tube-based design, fluctuations of the actuator error can then also be allowed.

6.5.4 General Use of Explicit MPC in Terminal Controllers

A contribution of this work that might prove useful in the future is the novel structure of the third MPC controller, which uses eMPC to enlarge the terminal set. The core idea lies in designing a terminal controller that combines a feedback-linearizing module with a linear eMPC controller. This combination serves two purposes: It extends the (terminal) controller's functionality beyond actuator saturation limits while simultaneously providing a terminal cost function, an approach that has to the best of the author's knowledge not been proposed yet.

In order to be applicable, a feedback-linearization needs to be possible in a certain region. If this is given, two more requirements need to be satisfied: First of all, a terminal cost for the feedback-linearizing controller needs to be derived and the cross term between the feedback-linearizing and the eMPC cost needs to be bounded. In the current formulation, the eMPC controller is due to the mathematical complexity only calculated for the states that are not contained in the feedback-linearizing controller. In mathematical terms, for the system

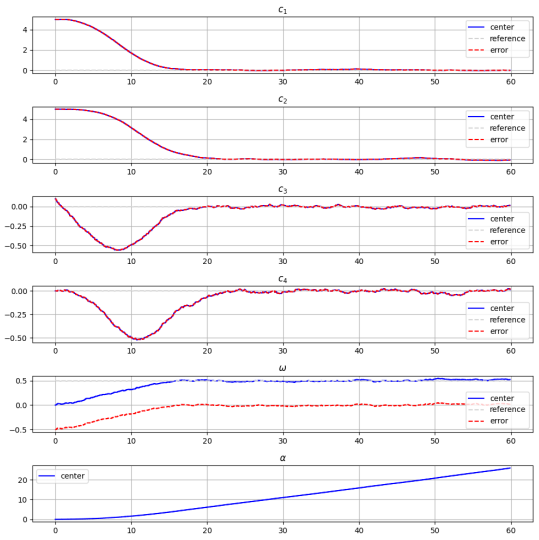
$$\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \begin{pmatrix} f_1(x_2) + Ax_1 \\ f_2(x_2) \end{pmatrix} + g(x_2)u$$

with constant matrix A and the nonlinear functions $f_1(x_2)$, $f_2(x_2)$ and $g(x_2)$, only x_2 is controlled by eMPC in the terminal controller. After the linearization step, x_1 is in turn controlled by a simple linear controller. Further research is needed to assess if and how eMPC-based terminal controllers can be calculated for systems with a different structure.

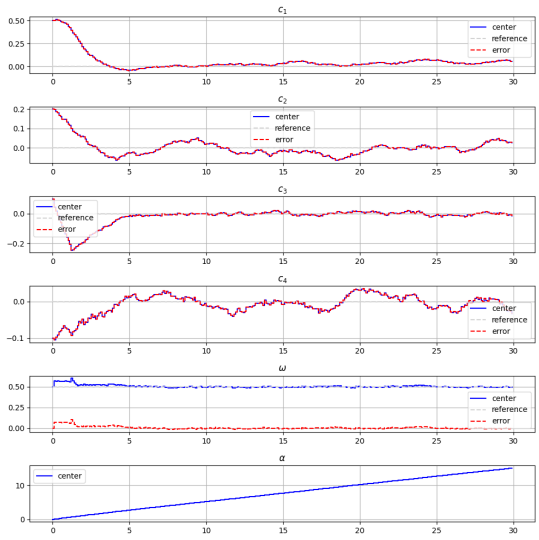
The second challenge is finding a good control allocation between the feedback-linearizing and the eMPC controller: As the combined control needs to satisfy the input constraints, it is necessary to assign available controls for each controller individually. As demonstrated in this work, sub-optimal solutions to the allocation problem may be preferable for practical implementation.

Despite these two challenges, it has clearly been demonstrated in Figure 6.1 that the benefits of this approach can be substantial. This novel combination of feedback linearization and eMPC for terminal control opens up promising directions for expanding the feasible region of MPC controllers in nonlinear systems, particularly when dealing with input constraints.

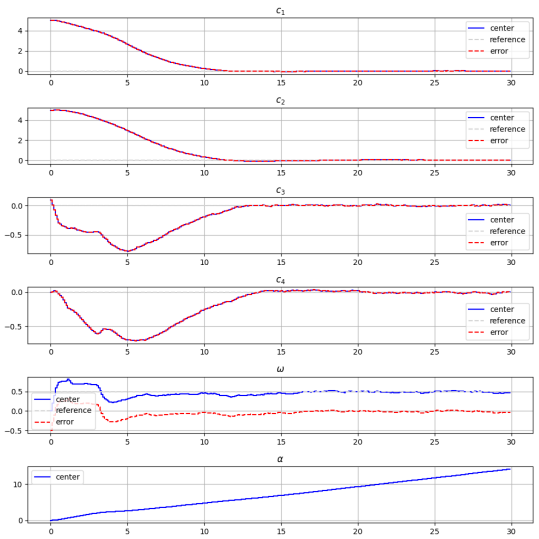
Having presented the major contributions and findings, the next chapter concludes this work with a summary and discussion of the results.



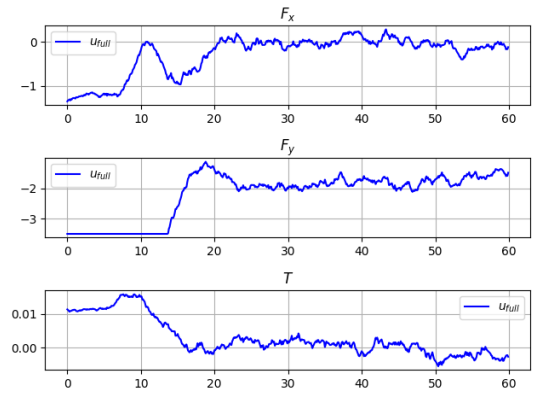
(a) Evolution of state c and error e using κ_{fb}



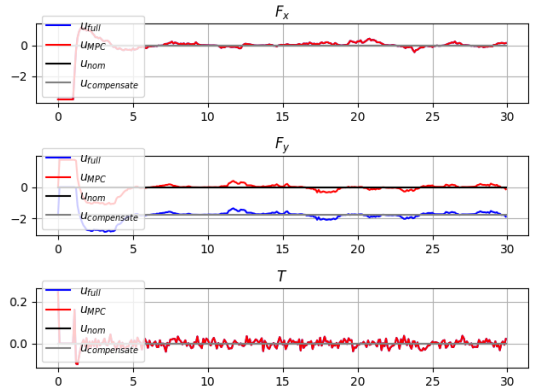
(b) Evolution of state c and error e using κ_1



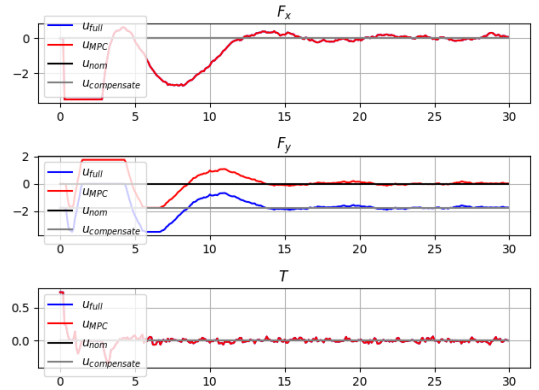
(c) Evolution of state c and error e using κ_2



(d) Inputs using κ_{fb}

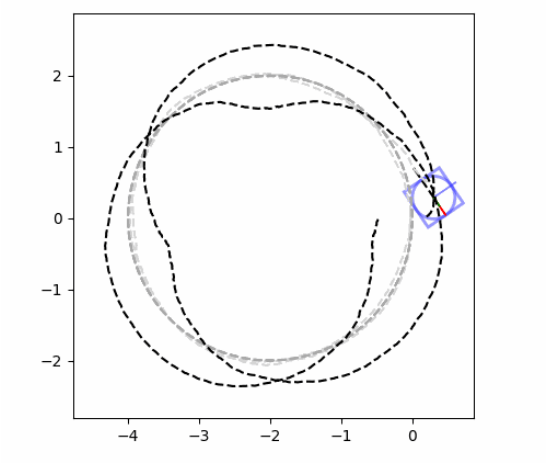


(e) Inputs using κ_1

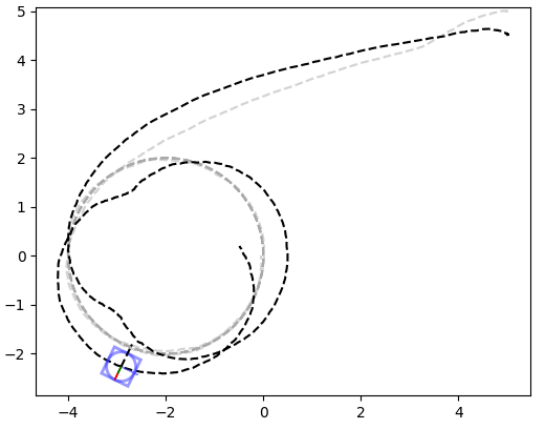


(f) Inputs using κ_2

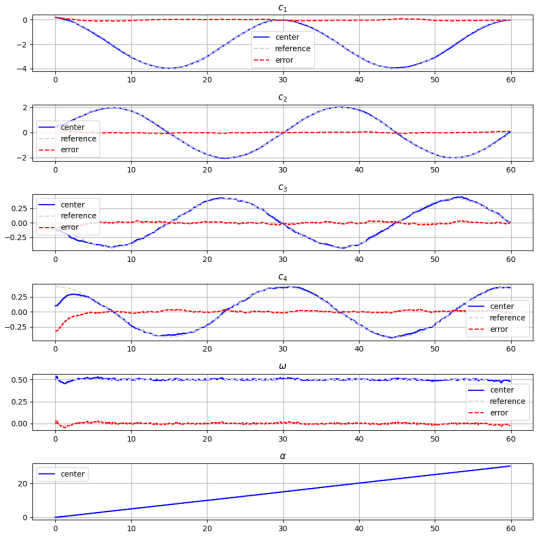
Figure 6.3: States and inputs of the different controllers for point stabilization. Supplementary to Figure 6.2.



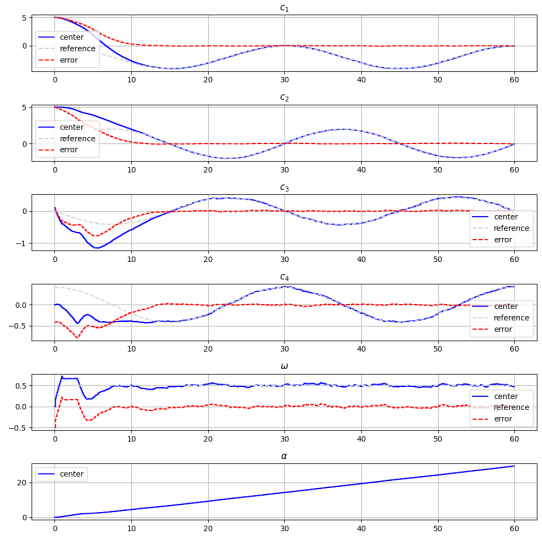
(a) Path using κ_1



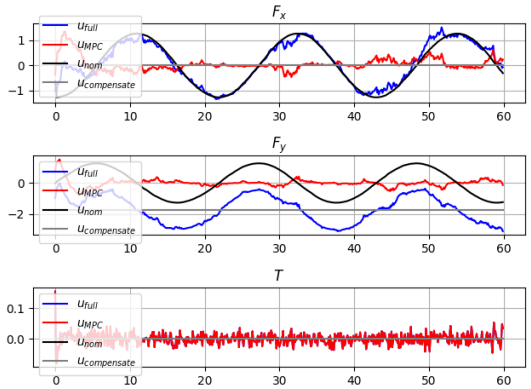
(b) Path using κ_2



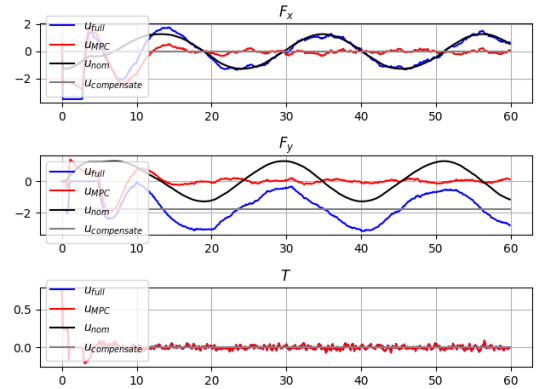
(c) Evolution of state c and error e using κ_1



(d) Evolution of state c and error e using κ_2



(e) Inputs using κ_1



(f) Inputs using κ_2

Figure 6.4: Path, states and inputs of the MPC controllers for a circular trajectory. κ_2 starting from $c = \begin{pmatrix} 5 & 5 & 0.2 & 0.3 & 0.5 & 0 \end{pmatrix}^T$ and κ_1 starting from $c = \begin{pmatrix} 0.2 & 0.5 & -0.1 & 0.1 & 0.5 & 0 \end{pmatrix}^T$ as the problem is infeasible with the other starting point.

Chapter 7

Conclusion and Outlook

In this thesis, methods for the control under actuator failures of small, mobile space robots such as the Astrobee or the SRL freeflyers are discussed. A literature review has revealed that no suitable control methods under actuator failures exist for this kind of system yet. While works exist on the control of satellites under actuator failures, the small autonomous robots have different requirement that need to be considered during control design. These come mainly from the need for collision avoidance and because these autonomous robots are designed to be smaller, more cost-effective and lighter wherefore they have fewer hardware redundancies. Thus it is desirable if they can use - both for the nominal as well as for the faulty case - all available actuators to their full extent in order to steer more agile maneuvers.

It has been found that Model Predictive Control is a suitable way of controlling these systems, which is in line with the developments in recent years. The advantage is that MPC can readily include the constraints that stem from actuator saturations and actuator failures.

According to the analysis, the actuator failures for the freeflyers do not lead to the need of a new control scheme if only one actuator fails. Instead, any control scheme that works for the nominal case can be used with adapted constraints and costs after a compensation of the error. This is not the case anymore for combinations of failures of two or three actuators in which case nonholonomic constraints appear for the system. Compensation of the resulting errors is in these cases not possible any more. An analysis of the number of cases with different nonholonomic constraints concludes that it is hardly feasible in practice to find a controller with guaranteed stability for each different case.

Thus, micro-orbiting is proposed as a viable strategy that allows to ensure stability for all combinations of actuator failures with up to three failures. This means that the system is brought on a small scale orbit, called micro-orbit, in which the partly compensated actuator failures and the orbit dynamics cancel out. By this, full compensation is not needed anymore and the remaining degrees of freedom for the inputs can be used to steer the system to a setpoint or along a trajectory.

A model is derived for the micro-orbiting system and in this formulation, the nonholonomic constraints disappear and the system becomes controllable. Three different controllers are designed, one for point-stabilization based on feedback-linearization and two for trajectory-tracking based on MPC. The two different MPC controllers differ in the design of the terminal controllers, and thus also terminal costs and sets. The terminal controller for the second MPC controller uses a combination of feedback-linearization and eMPC which allows for the derivation of a large terminal region.

Finally, the controllers are implemented and their performance is evaluated in a simulation. It is found that the feasible set for the first MPC controller could be too small because of a small terminal set. Depending on the hardware on which it is implemented, a large enough MPC horizon could be chosen to counteract this problem. This is however not the best option as the terminal, and thus also the feasible, set for the second MPC can be made even larger than necessary, thus exceeding the requirements. It is concluded that the second MPC controller is the preferred choice if the hardware requirements are satisfied for the implementation of an MPC scheme. Otherwise, the feedback-linearizing controller could be an alternative.

In future works, an experimental validation of the controllers on the physical freeflyers in the KTH Space Robotics Lab is planned. It is aimed for working on the generalizations discussed in Section 6.5, such as the generalization of micro-orbiting to 3D systems.

It is further of interest if and how the terminal controller using eMPC can be generalized for other systems. This could have the potential to increase the terminal sets of future MPC controllers and to reduce their computational requirements as smaller horizons can be chosen.

As a last point, a formal robustification of the presented methods could increase the level of safety that the controllers give. As the fault-detection methods employed together with the controller may not be 100% accurate, incorporating robust control techniques could enhance the system's resilience to uncertainties.

To summarize, this thesis contributes to the development of failsafe control schemes for autonomous space robots. The presented methods can handle the failure of up to three thrusters for the freeflyers at SRL.

References

[1] NASA, "Artemis program," https://www.nasa.gov/humans-in-space/artemis/, accessed: 2024-09-09. [Page 1.]

[2] 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. [Page 2.]

[3] C. P. Mark and S. Kamath, "Review of active space debris removal methods," *Space policy*, vol. 47, pp. 194–206, 2019. [Page 2.]

[4] V. Svotina and M. Cherkasova, "Space debris removal - review of technologies and techniques. flexible or virtual connection between space debris and service spacecraft," *Acta Astronautica*, vol. 204, pp. 840–853, 2023. [Page 2.]

[5] M. Tafazoli, "A study of on-orbit spacecraft failures," *Acta Astronautica*, vol. 64, no. 2-3, pp. 195–205, 2009. [Page 2.]

[6] NASA, "Astrobee," https://www.nasa.gov/astrobee/. [Page 2.]

[7] S. Phodapol, "Predictive controllers for load transportation in microgravity environments," Master's thesis, Kungliga Tekniska Högskolan (KTH), 2023. [Pages 2, 25, and 97.]

[8] M. G. Safonov, "Origins of robust control: Early history and future speculations," *Annual Reviews in Control*, vol. 36, no. 2, pp. 173–181, 2012. doi: https://doi.org/10.1016/j.arcontrol.2012.09.001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1367578812000363 [Page 3.]

[9] D. Mayne, S. Raković, R. Findeisen, and F. Allgöwer, "Robust output feedback model predictive control of constrained linear systems," *Automatica*, vol. 42, no. 7, pp. 1217–1222, Jul. 2006. doi: 10.1016/j.automatica.2006.03.005 [Pages 3 and 13.]

[10] J. M. Maciejowski, "Fault-tolerant aspects of MPC," in *IEE Seminar on Practical Experiences with Predictive Control (Ref. No. 2000/023)*, 2000. doi: 10.1049/ic:20000115 pp. 3/1–3/4. [Page 4.]

[11] J. M. Maciejowski and C. N. Jones, "MPC fault-tolerant flight control case study: flight 1862," *IFAC Proceedings Volumes*, vol. 36, no. 5, pp. 119–124, 2003. doi: https://doi.org/10.1016/S1474-6670(17)36480-7 5th IFAC Symposium on Fault Detection, Supervision and Safety of Technical

Processes 2003, Washington DC, 9-11 June 1997. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1474667017364807 [Page 4.]

[12] M. Kale and A. Chipperfield, "Stabilized MPC formulations for robust reconfigurable flight control," *Control Engineering Practice*, vol. 13, no. 6, pp. 771–788, 2005. doi: https://doi.org/10.1016/j.conengprac.2004.09.001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0967066104002011 [Page 4.]

[13] S. Gros, R. Quirynen, and M. Diehl, "Aircraft control based on fast non-linear MPC & multiple-shooting," in *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, 2012. doi: 10.1109/CDC.2012.6426439 pp. 1142–1147. [Page 4.]

[14] S. Magdici and M. Althoff, "Fail-safe motion planning of autonomous vehicles," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, 2016. doi: 10.1109/ITSC.2016.7795594 pp. 452–458. [Page 5.]

[15] C. Pek and M. Althoff, "Fail-safe motion planning for online verification of autonomous vehicles using convex optimization," *IEEE Transactions on Robotics*, vol. 37, no. 3, pp. 798–814, 2021. doi: 10.1109/TRO.2020.3036624 [Page 5.]

[16] T. Brüdigam, M. Olbrich, D. Wollherr, and M. Leibold, "Stochastic model predictive control with a safety guarantee for automated driving," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 1, pp. 22–36, 2023. doi: 10.1109/TIV.2021.3074645 [Page 5.]

[17] H. Bui, A. Muraleedharan, S. Sinha, H. Okuda, and T. Suzuki, "Design of fail-safe model predictive controller for sudden changes in driving scenes," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, 2021. doi: 10.1109/ITSC48978.2021.9564565 pp. 790–797. [Page 5.]

[18] P. Hang and X. Chen, "Towards autonomous driving: Review and perspectives on configuration and control of four-wheel independent drive/steering electric vehicles," *Actuators*, vol. 10, no. 8, p. 184, Aug. 2021. doi: 10.3390/act10080184 [Page 5.]

[19] D. Zhang, G. Liu, H. Zhou, and W. Zhao, "Adaptive sliding mode fault-tolerant coordination control for four-wheel independently driven electric vehicles," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 11, pp. 9090–9100, Nov. 2018. doi: 10.1109/tie.2018.2798571 [Page 5.]

[20] T. Chen, L. Chen, X. Xu, Y. Cai, H. Jiang, and X. Sun, "Passive fault-tolerant path following control of autonomous distributed drive electric vehicle considering steering system fault," *Mechanical Systems and Signal Processing*, vol. 123, pp. 298–315, May 2019. doi: 10.1016/j.ymssp.2019.01.019 [Page 5.]

[21] B. Guo and Y. Chen, "Robust adaptive fault-tolerant control of four-wheel independently actuated electric vehicles," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 5, pp. 2882–2894, May 2020. doi: 10.1109/tii.2018.2889292 [Page 5.]

[22] G. Zhang, H. Zhang, X. Huang, J. Wang, H. Yu, and R. Graaf, "Active fault-tolerant

control for electric vehicles with independently driven rear in-wheel motors against certain actuator faults," *IEEE Transactions on Control Systems Technology*, vol. 24, no. 5, pp. 1557–1572, Sep. 2016. doi: 10.1109/tcst.2015.2501354 [Page 5.]

[23] N. Lodder, C. van der Ploeg, L. Ferranti, and E. Silvas, "Optimization-based fault mitigation for safe automated driving," *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 1094–1100, 2023. doi: 10.1016/j.ifacol.2023.10.1710 [Page 5.]

[24] M. W. Mueller and R. D'Andrea, "Stability and control of a quadrocopter despite the complete loss of one, two, or three propellers," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2014. doi: 10.1109/icra.2014.6906588 [Pages 5, 6, and 49.]

[25] G. P. S. Rible, N. A. A. Arriola, and M. C. Ramos, "Fail-safe controller architectures for quadcopter with motor failures," in *2020 6th International Conference on Control, Automation and Robotics (ICCAR)*, 2020. doi: 10.1109/ICCAR49639.2020.9108038 pp. 384–391. [Page 6.]

[26] H. A. Izadi, Y. Zhang, and B. W. Gordon, "Fault tolerant model predictive control of quad-rotor helicopters with actuator fault estimation," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 6343–6348, Jan. 2011. doi: 10.3182/20110828-6-it-1002.03709 [Page 6.]

[27] F. Nan, S. Sun, P. Foehn, and D. Scaramuzza, "Nonlinear MPC for quadrotor fault-tolerant control," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5047–5054, Apr. 2022. doi: 10.1109/lra.2022.3154033 [Pages 6 and 49.]

[28] D. Tzoumanikas, Q. Yan, and S. Leutenegger, "Nonlinear MPC with motor failure identification and recovery for safe and aggressive multicopter flight," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020. doi: 10.1109/ICRA40945.2020.9196690 pp. 8538–8544. [Page 6.]

[29] B. Nejad, *Introduction to Satellite Ground Segment Systems Engineering: Principles and Operational Aspects*. Springer International Publishing, 2023. ISBN 9783031159008 [Pages 6 and 8.]

[30] C. M. Pong, "Autonomous thruster failure recovery for underactuated spacecraft," Master's thesis, Massachusetts Institute of Technology, 2010. [Pages 7, 8, 29, and 37.]

[31] C. M. Pong, A. Saenz Otero, and D. W. Miller, "Autonomous thruster failure recovery on underactuated spacecraft using model predictive control," *Guidance and Control 2011: Proceedings of the 34th Annual AAS Rocky Mountain Section Guidance and Control Conference*, vol. 141, pp. 107–126, Feb. 2011, version: Author's final manuscript. [Pages 7 and 8.]

[32] P. Crouch, "Spacecraft attitude control and stabilization: Applications of geometric control theory to rigid body models," *IEEE Transactions on Automatic Control*, vol. 29, no. 4, pp. 321–331, 1984. [Pages 7 and 8.]

[33] C. I. Byrnes and A. Isidori, "On the attitude stabilization of rigid spacecraft," *Automatica*, vol. 27, no. 1, pp. 87–95, 1991. doi: https://doi.org/10.1016/0005-

1098(91)90008-P. [Online]. Available: https://www.sciencedirect.com/science/article/pii/000510989190008P [Page 7.]

[34] H. Krishnan, N. H. McClamroch, and M. Reyhanoglu, "Attitude stabilization of a rigid spacecraft using two momentum wheel actuators," *Journal of Guidance, Control, and Dynamics*, vol. 18, no. 2, pp. 256–263, Mar. 1995. doi: 10.2514/3.21378 [Page 7.]

[35] V. Coverstone-Carroll, "Detumbling and reorienting underactuated rigid spacecraft," *Journal of Guidance, Control, and Dynamics*, vol. 19, no. 3, pp. 708–710, 1996. [Page 7.]

[36] D. Casagrande, A. Astolfi, and T. Parisini, "Global asymptotic stabilization of the attitude and the angular rates of an underactuated non-symmetric rigid body," *Automatica*, vol. 44, no. 7, pp. 1781–1789, Jul. 2008. doi: 10.1016/j.automatica.2007.11.022 [Page 7.]

[37] J. S. Hall, M. Romano, and R. Cristi, "Quaternion feedback regulator for large angle maneuvers of underactuated spacecraft," in *Proceedings of the 2010 American Control Conference*, 2010. doi: 10.1109/ACC.2010.5531481 pp. 2867–2872. [Page 7.]

[38] D. Aguilar-Marsillach, S. Di Cairano, and A. Weiss, "Abort-safe spacecraft rendezvous in case of partial thrust failure," in *2020 59th IEEE Conference on Decision and Control (CDC)*, 2020. doi: 10.1109/CDC42340.2020.9303782 pp. 1490–1495. [Page 8.]

[39] ——, "Fail-safe rendezvous control on elliptic orbits using reachable sets," in *2020 American Control Conference (ACC)*, 2020. doi: 10.23919/ACC45564.2020.9147957 pp. 4920–4925. [Page 8.]

[40] D. Aguilar-Marsillach, S. Di Cairano, U. Kalabić, and A. Weiss, "Fail-safe spacecraft rendezvous on near-rectilinear halo orbits," in *2021 American Control Conference (ACC)*, 2021. doi: 10.23919/ACC50511.2021.9483328 pp. 2980–2985. [Page 8.]

[41] D. Aguilar-Marsillach, S. Di Cairano, and A. Weiss, "Abort-safe spacecraft rendezvous on elliptic orbits," *IEEE Transactions on Control Systems Technology*, vol. 31, no. 3, pp. 1133–1148, 2023. doi: 10.1109/TCST.2022.3216077 [Page 8.]

[42] S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge: Cambridge University Press, 2004. ISBN 0-521-83378-7 [Pages 13, 43, 60, 61, and 71.]

[43] J. P. Hespanha, *Linear Systems Theory*. Princeton: Princeton University Press, 2023. ISBN 9781400890088. [Online]. Available: https://doi.org/10.23943/9781400890088 [Pages 15 and 53.]

[44] A. Bloch, *Nonholonomic Mechanics and Control*, P. S. Krishnaprasad and R. Murray, Eds. Springer New York, 2015. ISBN 9781493930173 [Page 15.]

[45] R. W. Brockett, "Asymptotic stability and feedback stabilization," *Differential geometric control theory*, vol. 27, no. 1, pp. 181–191, 1983. [Pages 15 and 36.]

[46] P. Roque, W. S. Cortez, L. Lindemann, and D. V. Dimarogonas, "Corridor MPC: Towards optimal and safe trajectory tracking," in *2022 American Control Conference (ACC)*, 2022. doi: 10.23919/ACC53348.2022.9867764 pp. 2025–2032. [Page 15.]

[47] A. Lindquist, J. Sand, and X. Hu, "An introduction to mathematical systems theory," 2023. [Page 16.]

[48] T. A. Badgwell and S. J. Qin, *Model-Predictive Control in Practice*. London: Springer London, 2015, pp. 756–760. ISBN 978-1-4471-5058-9. [Online]. Available: https://doi.org/10.1007/978-1-4471-5058-9_8 [Page 16.]

[49] M. N. Zeilinger, "Real-time model predictive control," Ph.D. dissertation, ETH Zurich, 2011. [Page 17.]

[50] L. Persson, "Model predictive control for cooperative rendezvous of autonomous unmanned vehicles," phdthesis, KTH Royal Institute of Technology, 2021. [Page 17.]

[51] J. M. Manzano Crespo, "Learning-based model predictive control for constrained nonlinear systems," Ph.D. dissertation, University of Seville, 2020. [Page 17.]

[52] J. B. Rawlings and K. R. Muske, "The stability of constrained receding horizon control," *IEEE transactions on automatic control*, vol. 38, no. 10, pp. 1512–1516, 1993. [Page 18.]

[53] M. Johansson, "Lecture notes on linear quadratic and model predictive control," Sep. 2023. [Pages 18 and 73.]

[54] D. Mayne, J. Rawlings, C. Rao, and P. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000. doi: https://doi.org/10.1016/S0005-1098(99)00214-9. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0005109899002149 [Page 18.]

[55] H. Chen and F. Allgöwer, "A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability," *Automatica*, vol. 34, no. 10, pp. 1205–1217, 1998. [Pages 18, 55, 63, and 64.]

[56] J. Matschek, T. Bäthge, T. Faulwasser, and R. Findeisen, "Nonlinear predictive control for trajectory tracking and path following: An introduction and perspective," in *Handbook of Model Predictive Control*, S. V. Raković and W. S. Levine, Eds. Birkhäuser, 2019. [Page 19.]

[57] T. Faulwasser and R. Findeisen, "A model predictive control approach to trajectory tracking problems via time-varying level sets of lyapunov functions," in *2011 50th IEEE Conference on Decision and Control and European Control Conference*, 2011. doi: 10.1109/CDC.2011.6160492 pp. 3381–3386. [Pages 20 and 21.]

[58] T. Faulwasser, "Optimization-based solutions to constrained trajectory-tracking and path-following problems," Ph.D. dissertation, RWTH Aachen, 2013. [Pages 20 and 21.]

[59] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, "The explicit linear

quadratic regulator for constrained systems," *Automatica*, vol. 38, no. 1, pp. 3–20, 2002. [Pages 21, 23, and 72.]

[60] A. Alessio and A. Bemporad, "A survey on explicit model predictive control," *Nonlinear Model Predictive Control: Towards New Challenging Applications*, pp. 345–369, 2009. [Page 21.]

[61] M. Lazar and W. P. M. H. Heemels, "A semi-explicit MPC set-up for constrained piecewise affine systems," in *2003 European Control Conference (ECC)*, 2003. doi: 10.23919/ECC.2003.7085037 pp. 695–700. [Page 23.]

[62] F. Scibilia, S. Olaru, and M. Hovd, "Approximate explicit linear MPC via delaunay tessellation," in *2009 European Control Conference (ECC)*. IEEE, 2009, pp. 2833–2838. [Page 23.]

[63] M. N. Zeilinger, C. N. Jones, and M. Morari, "Real-time suboptimal model predictive control using a combination of explicit MPC and online optimization," *IEEE Transactions on Automatic Control*, vol. 56, no. 7, pp. 1524–1534, 2011. doi: 10.1109/TAC.2011.2108450 [Page 23.]

[64] A. Bemporad, "A multiparametric quadratic programming algorithm with polyhedral computations based on nonnegative least squares," *IEEE Transactions on Automatic Control*, vol. 60, no. 11, pp. 2892–2903, 2015. [Page 23.]

[65] M. F. Smith and C. Provencher, "Astrobee systems engineering design overview," NASA Ames Research Center, Tech. Rep., 2015. [Pages 25 and 26.]

[66] A. Perucca and E. Torti, "Congruence theorems for convex polygons involving sides, angles, and diagonals," *International Journal of Geometry*, vol. 12, no. 2023, 2023. [Pages 30 and 31.]

[67] G. M. Ziegler, *Lectures on polytopes*, 7th ed., ser. Graduate Texts in Mathematics ; Volume 152. New York: Springer Science+Business Media, 1995. ISBN 1-4613-8431-1 [Page 31.]

[68] A. De Luca and M. D. Di Benedetto, "Control of nonholonomic systems via dynamic compensation," *Kybernetika*, vol. 29, no. 6, pp. 593–608, 1993. [Page 36.]

[69] E. Schuster, "Lecture notes in nonlinear systems and control," 2024. [Page 36.]

[70] H. K. Khalil, *Nonlinear systems*, 3rd ed. Upper Saddle river: Prentice Hall, 2002. ISBN 0-13-067389-7 [Pages 36 and 51.]

[71] W. Seeman and J. Hofinger, "Lecture notes in engineering mechanics 3 and 4," Oct. 2018. [Page 39.]

[72] A. Freddi, A. Lanzon, and S. Longhi, "A feedback linearization approach to fault tolerance in quadrotor vehicles," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 5413–5418, 2011. doi: https://doi.org/10.3182/20110828-6-IT-1002.02016 18th IFAC World Congress. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1474667016444678 [Page 49.]

[73] R. Olfati-Saber, "Exponential ε-tracking and ε-stabilization of second-order nonholonomic se(2) vehicles using dynamic state feedback," in *Proceedings of the 2002 American Control Conference (IEEE Cat. No.CH37301)*. IEEE, 2002. doi: 10.1109/acc.2002.1024548 [Page 49.]

[74] ——, "Near-identity diffeomorphisms and exponential ε-tracking and ε-stabilization of first-order nonholonomic se(2) vehicles," in *Proceedings of the 2002 American Control Conference (IEEE Cat. No.CH37301)*, vol. 6, 2002. doi: 10.1109/ACC.2002.1025398 pp. 4690–4695 vol.6. [Page 49.]

[75] H. Sedrakyan and N. Sedrakyan, *The HM-GM-AM-QM Inequalities*. Cham: Springer International Publishing, 2018, pp. 21–43. [Online]. Available: https://doi.org/10.1007/978-3-319-77836-5_3 [Page 59.]

[76] V. Nevistic and M. Morari, "Constrained control of feedback-linearizable systems," in *Proc. 3rd European Control Conference ECC'95*, 1995, pp. 1726–1731. [Page 66.]

[77] H. Chen and F. Allgöwer, *Nonlinear Model Predictive Control Schemes with Guaranteed Stability*. Dordrecht: Springer Netherlands, 1998, pp. 465–494. ISBN 978-94-011-5094-1. [Online]. Available: https://doi.org/10.1007/978-94-011-5094-1_16 [Page 66.]

[78] T. M. Apostol, "Calculus," *Wiley*, vol. 2, 2000. [Page 67.]

[79] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "Casadi: a software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, pp. 1–36, 2019. [Page 75.]

[80] T. Marucchi, "pympc," https://github.com/TobiaMarcucci/pympc, 2024. [Page 75.]

Appendix A

Control Allocation

In Section 3.1.2, the model was formulated using the resulting force and torque $\boldsymbol{F} \in \mathbb{R}^3$ for simplicity. This approach simplifies the control problem by avoiding the need for the controller to handle ambiguities arising from different input combinations that produce the same system behavior. Generally, it is desirable to utilize the lowest-energy input to maximize to maximize battery life and operational duration. The system described here is also referred to as a control allocator in the literature. It's worth noting that the presented solution can be easily reconfigured by adjusting the input bounds.

The problem of control allocation can be formulated as a constrained convex optimization problem of the form

$$\begin{aligned} \underset{\boldsymbol{F}_{i,j}}{\text{minimize}} \quad & \frac{1}{2}\boldsymbol{F}_{i,j}^T \boldsymbol{F}_{i,j} && \text{(A.1a)} \\ \text{subject to} \quad & \boldsymbol{F} = \boldsymbol{D}\boldsymbol{F}_{i,j} + \boldsymbol{F}_{\text{fault}}, && \text{(A.1b)} \\ & \boldsymbol{0} \leq \boldsymbol{F}_{i,j} \leq F_{\text{bound}} && \text{(A.1c)} \end{aligned}$$

where for each element of F_{bound} holds

$$F_{\text{bound},i,j} = \begin{cases} F_{\max} & \text{if actuator is working} \\ 0 & \text{if actuator has failed} \end{cases} \tag{A.2}$$

This problem can easily be solved using standard numerical solvers.

It is however also possible to solve the problem explicitly, allowing for faster solution times. This is described in the following: First of all, as for all thrust forces holds $F_{i,j} \geq 0$, another representation for the inputs can be

$$F_i = F_{i,1} - F_{i,2}, \;\; i = 1, 2, 3, 4 \tag{A.3}$$

where the constraints can be given as

$$F_i \in [F_{i-}, F_{i+}]. \tag{A.4}$$

Note that two opposing failed thrusters simply imply $F_i = F_{i-} = F_{i+}$. Then the underdetermined system simplifies to

$$\boldsymbol{F} - \boldsymbol{F}_{\text{fault}} = \begin{pmatrix} F_x^{\text{des}} \\ F_y^{\text{des}} \\ F_T^{\text{des}} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ d & -d & d & -d \end{pmatrix} \begin{pmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \end{pmatrix} \tag{A.5}$$

respectively

$$\begin{pmatrix} F_x^{\text{des}} \\ F_y^{\text{des}} \\ F_z^{\text{des}} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \end{pmatrix} \tag{A.6}$$

where $F_z^{\text{des}} = \frac{1}{2}\left(F_x^{\text{des}} + F_y^{\text{des}} + \frac{F_T^{\text{des}}}{d}\right)$.

The latter equation can be used to derive the sets of possible input combinations that satisfies the constraint Eq. (A.4) as well as the input constraints: Fix the third variable $F_3 = c$ from which follows the solution of Eq. (A.6)

$$F_3 = c \tag{A.7a}$$
$$F_4 = F_y^{\text{des}} - c \tag{A.7b}$$
$$F_1 = F_z^{\text{des}} - c \tag{A.7c}$$
$$F_2 = F_x^{\text{des}} - F_1 = F_x^{\text{des}} - F_z^{\text{des}} + c. \tag{A.7d}$$

From the constraints in Eq. (A.4) and Eq. (A.7), the according constraint for c follows as

$$\begin{aligned} c \in \mathcal{C} = [&\max(F_{3-}, F_y^{\text{des}} - F_{4+}, F_x^{\text{des}} - F_{1+}, F_x^{\text{des}} - F_z^{\text{des}} + F_{2-}), \\ &\min(F_{3+}, F_y^{\text{des}} - F_{4-}, F_x^{\text{des}} - F_{1-}, F_x^{\text{des}} - F_z^{\text{des}} + F_{2+})] \end{aligned} \tag{A.8}$$

with which the optimization problem can be reformulated as an optimization in one variable as

$$\underset{c}{\text{minimize}} \quad \frac{1}{2}(F_1^2 + F_2^2 + F_3^2 + F_4^2) \tag{A.9a}$$
$$\text{subject to} \quad c \in \mathcal{C}, \tag{A.9b}$$
$$F_i \text{ as by Eq. (A.7).} \tag{A.9c}$$

All solutions for this optimization problem satisfy constraint A.1b and the optimal solution can simply be found by derivation of Eq. (A.9a) for c and checking of the boundary constraints.

Appendix B

Simulation Parameters

At this place, the parameters used for the simulation in Chapter 6 are given. The physical values were taken from [7] and stem from the freeflyers in KTHs Space Robotics Lab.

Parameter	Value
Physical parameters	
m	$14.5\,\mathrm{kg}$
J	$0.370\,\mathrm{kg\,m^2}$
d	$0.14\,\mathrm{m}$
$F_{\max}$	$1.75\,\mathrm{N}$
Simulation environment	
Sampling time environment δ_{env}	$0.01\,\mathrm{s}$
Sampling time controller δ	$0.1\,\mathrm{s}$
Noise on position	Uniform distribution in the range of of $\delta_{\mathrm{env}}[-0.25\,\mathrm{m}, 0.25\,\mathrm{m}]$
Noise on orientation	Uniform distribution in the range of of $\delta_{\mathrm{env}}[-0.25\,\mathrm{rad}, 0.25\,\mathrm{rad}]$
Failed actuators	$F_{3,1} = F_{4,1} = F_{\max}$
Controller tuning	
Radius r for micro-orbiting (ω_0 follows from the virtual fault)	$0.5\,\mathrm{m}$
Feedback controller κ_{fb}	
State cost for calculating the feedback gain	$\mathrm{diag}\begin{pmatrix}1 & 1 & 1 & 1 & 1\end{pmatrix}$
Input cost for calculating the feedback gain	$\mathrm{diag}\begin{pmatrix}10 & 10 & 10\end{pmatrix}$
MPC controller κ_1 using linear feedback	
MPC horizon	20 steps
State cost Q_e	$\mathrm{diag}\begin{pmatrix}5 & 5 & 0.2 & 0.2 & 100\end{pmatrix}$

Control cost Q_u	$\mathrm{diag}\begin{pmatrix}1 & 1 & 1\end{pmatrix}$
State cost for calculating the linear controller	$\mathrm{diag}\begin{pmatrix}1 & 1 & 1 & 1 & 1\end{pmatrix}$
Control cost for calculating the linear controller	$\mathrm{diag}\begin{pmatrix}10 & 10 & 10\end{pmatrix}$
MPC controller κ_2 using eMPC	
MPC horizon	20 steps
State cost Q_e	$\mathrm{diag}\begin{pmatrix}1 & 1 & 1 & 1 & 1\end{pmatrix}$
Control cost Q_u	$\mathrm{diag}\begin{pmatrix}0.1 & 0.1 & 0.1\end{pmatrix}$
k_ω	0.1
eMPC state cost $\hat{Q}_\mathrm{e}$	$\mathrm{diag}\begin{pmatrix}1 & 1 & 1 & 1 & 1\end{pmatrix}$
eMPC control cost $\hat{Q}_\mathrm{u}$	$\mathrm{diag}\begin{pmatrix}10 & 10 & 10\end{pmatrix}$