DISS. ETH NO. 28265

DATA-DRIVEN MODEL-BASED CONTROL FOR NOVEL FLYING MACHINES: FROM HIGHLY UNDERACTUATED TO OVERACTUATED SYSTEMS WITH UNCERTAIN DYNAMICS

A thesis submitted to attain the degree of

DOCTOR OF SCIENCES of ETH ZURICH

(Dr. sc. ETH Zurich)

presented by

WEIXUAN ZHANG

MSc ETH in Mechanical Engineering

born on August 26, 1989
citizen of Nanjing, China

Prof. Dr. Roland Siegwart, Examiner
Prof. Dr. Raffaello D'Andrea, Co-examiner
Prof. Dr. Mark W. Mueller, Co-examiner

2022

Autonomous Systems Lab
Department of Mechanical and Process Engineering
ETH Zurich
Switzerland

© 2022 Weixuan Zhang. All rights reserved.

Abstract

The central theme of this thesis is to push the limits of aerial robotics via novel vehicle design and by developing methods that combine model-based control with machine learning approach to exploit the best aspects of both methods. Conventional quadrotors have become popular due to their mechanical simplicity and agility. Meanwhile, novel vehicle designs endow flying machines with new capabilities that a standard quadrotor lacks. This thesis investigates two designs: 1. a flying vehicle which has a single moving part and is yet able to hover and fully control its position, the Monospinner; 2. a tilt-arm overactuated (18 actuators) omnidirectional flying vehicle, the Omav. For the former design, this highly underactuated flying machine aims to answer a fundamental question about flying machine: what is the minimum number of moving parts for a controllable flying vehicle. The latter design is tailored for aerial physical interaction tasks, for which flying vehicles need to exchange forces with their surrounding environment. The Omav thus has a full actuation wrench envelope (that is, the ability of producing force and torque in arbitrary directions) and is inevitably mechanically complex.

These novel designs require new modeling and control approaches: the controlled system has to be robust against internal and external disturbances, these include modeling error due to mechanical complexity, aerodynamics, manufacturing imperfections, measurement noise, and environment disturbance. While first-principle model-based control methods are powerful for design and control of dynamical systems and applied on the Monospinner, tools from statistical learning theory help to capture the part of the Omav dynamics that is hard to model using first principles. This in turn helps model-based control method to perform optimally for such a mechanical complex flying machine. Challenges arise along with this novel approach such as the data collection efficiency and efficacy and the embedding of the learned model into the controller.

For the Monospinner, the core approach is to use model based control theory to find a vehicle design that is robust under disturbance. In particular, its translational and attitude dynamics are formulated as a twelve-dimensional state space system, which may be linearized to a linear time-invariant system amenable to controllability analysis, controller synthesis, and vehicle design. A mathematical analysis is given to show the vehicle is fully controllable in position after removing its yaw state, and in particular for the case of a vehicle with the shape of a planar object and an offset thrust location (with respect to its center of mass). The equilibrium of the resulting system has a large region of attraction such that it recovers after being thrown into the air like a frisbee.

The research into the Omav puts emphasis on algorithmic methods. This thesis presents an approach that combines a data-driven and a first-principle model for the system actuation and uses it to improve the controller. Particular attention is paid to avoid ambiguous solutions present in a standard inverse model. This is solved by an optimization problem only using forward models. In addtion, an efficient training data collection procedure is devised using an optimization problem formulation to find an informative trajectory. We present a sampling-based method that computes an approximation of the trajectory that minimizes the predictive uncertainty of the learned dynamics model. This trajectory is then executed, collecting the data to update the learned model. Last but not least, an adaptive control strategy is proposed for aerial sliding on surfaces with discontinuous change in geometry and unknown, spatially-varying friction properties. It augments a standard impedance controller using a control parameter adjustment policy, which combines proprioceptive measurements, tactile sensing and control signals as the policy input. In particular, this policy is trained in simulation with simplified actuator dynamics and yet is capable of being transferred to the robot without any adaptation. The key to this is the preserved controller structure.

Indoor experiments of the Monospinner and the Omav have been evaluated in various conditions.

Zusammenfassung

Das zentrale Thema dieser Arbeit ist die Erweiterung der Grenzen der Luftrobotik durch neuartiges Fahrzeugdesign und die Entwicklung von Methoden, die modellbasierte Steuerung mit Ansätzen des maschinellen Lernens kombinieren, um die besten Aspekte beider Methoden zu nutzen.

Aufgrund der jüngsten Entwicklungen in Sensoren, Computertechnologie und Elektromotoren sind Mikro-Luftfahrzeuge mit Rotationsflügeln mittlerweile weit verbreitet. Konventionelle Quadrokopter erfreuen sich aufgrund ihrer mechanischen Einfachheit und Wendigkeit großer Beliebtheit. Gleichzeitig verleihen neuartige Fahrzeugdesigns Flugmaschinen neue Fähigkeiten, die ein herkömmlicher Quadrokopter nicht besitzt.

Diese Arbeit untersucht zwei solcher Designs:

1. Ein Flugzeug mit nur einem beweglichen Teil, das schweben und seine Position vollständig kontrollieren kann, der Monospinner; 2. Ein tilt-arm überaktuiertes (18 Aktuatoren) omnidirektionales Flugzeug, das Omav.

Bei ersterem Design, einer stark unteraktuierten Flugmaschine, geht es darum, eine grundlegende Frage über Flugmaschinen zu beantworten: Wie viele bewegliche Teile sind notwendig, um ein steuerbares Flugzeug zu bauen? Das letztere Design ist auf Luftaufgaben zugeschnitten, bei denen Flugzeuge Kräfte mit ihrer Umgebung austauschen müssen. Das Omav verfügt über einen vollständigen Aktuationskraft-Momentenbereich (das heißt, die Fähigkeit, Kräfte und Drehmomente in beliebige Richtungen zu erzeugen) und ist zwangsläufig mechanisch komplex.

Diese neuen Designs erfordern neue Modellierungs- und Steuerungsansätze: Das gesteuerte System muss robust gegen interne und externe Störungen sein, zu denen Modellierungsfehler aufgrund von mechanischer Komplexität, Aerodynamik, Herstellungsfehlern, Messrauschen und Umgebungsstörungen gehören. Während modellbasierte Steuerungsmethoden aus erster Prinzipien für Design und Steuerung dynamischer Systeme mächtig sind und auf den Monospinner angewendet werden, helfen Werkzeuge aus der statistischen Lerntheorie dabei, den Teil der Omav-Dynamik zu erfassen, der sich schwer aus ersteren Prinzipien modellieren lässt. Dies wiederum hilft der modellbasierten Steuermethode, für eine solche mechanisch komplexe Flugmaschine optimal zu funktionieren. Mit diesem neuartigen Ansatz treten Herausforderungen auf, wie zum Beispiel die Effizienz und Wirksamkeit der Datensammlung und die Einbettung des gelernten Modells in den Regler.

Für den Monospinner ist der Kernansatz die Verwendung der modellbasierten Regelungstheorie, um ein Fahrzeugdesign zu finden, das robust gegenüber Störungen ist. Insbesondere werden seine translations- und attitude-Dynamik als zwölfdimensionales Zustandsraumsystem formuliert, das linearisiert werden kann, um ein lineares zeitinvariantes System zu erhalten, das für die Kontrollierbarkeitsanalyse, die Reglersynthese und das Fahrzeugdesign geeignet ist. Es wird eine mathematische Analyse vorgelegt, die zeigt, dass das Fahrzeug nach Entfernen des Gierzustands in der Position vollständig kontrollierbar ist, insbesondere für den Fall eines Fahrzeugs mit der Form eines planaren Objekts und einem verschobenen Schubpunkt (im Vergleich zum Schwerpunkt). Das Gleichgewicht des resultierenden Systems hat eine große Anziehungskraftsregion, so dass es sich nach dem Wurf in die Luft wie ein Frisbee wieder erholt.

Die Forschung zum Omav legt den Schwerpunkt auf algorithmische Methoden. Diese Arbeit stellt einen Ansatz vor, der ein datengetriebenes und ein erstprinzipielles Modell für die Systemanregung kombiniert und es zur Verbesserung des Reglers nutzt. Besondere Aufmerksamkeit wird darauf gelegt, mehrdeutige Lösungen zu vermeiden, die in einem standardmäßigen inversen Modell vorhanden sind. Dies wird durch ein Optimierungsproblem gelöst, das nur Vorwärtsmodelle verwendet. Zusätzlich wird ein effizientes Verfahren zur Datensammlung formuliert, um eine informative Trajektorie zu finden. Wir stellen eine samplingbasierte Methode vor, die eine Approximation der Trajektorie berechnet, die die Vorhersageunsicherheit des gelernten Dynamikmodells minimiert. Diese Trajektorie wird dann ausgeführt, um die Daten zur Aktualisierung des gelernten Modells zu sammeln. Nicht zuletzt wird eine adaptive Regelungsstrategie für das Gleiten auf Oberflächen mit diskontinuierlichem Geometriewechsel und unbekannten, räumlich variierenden Reibungseigenschaften vorgeschlagen. Es erweitert einen Standard-Impedanzregler um eine Regelungsparameteranpassungsrichtlinie, die propriozeptive Messungen, taktile Sensoren und Regelungssignale als Eingang nutzt. Insbesondere wird diese Richtlinie in der Simulation mit vereinfachten Aktordynamiken trainiert und kann dennoch ohne Anpassung auf den Roboter übertragen werden. Der Schlüssel dazu ist die erhaltene Reglerstruktur.

Indoor-Experimente des Monospinners und des Omavs wurden unter verschiedenen Bedingungen evaluiert.

Acknowledgments

This thesis would not have been possible without the huge support, contributions, help, patience, and friendship of many people.

Firstly, I would like to thank Prof. Roland Siegwart and Prof. Raffaello D'Andrea for letting me join their labs. Roland created a unique family-like working environment where I experienced exceptional friendliness, openness, mutual support, but also wanted to excel. Thank you Raff for teaching me how to think critically and showing how essential commitment and focus are. I will take them to heart.

Secondly, I would also like to thank Prof. Mark W. Mueller for being my co-examiner and all your advice calls, I highly appreciated that.

A large body of the research in this thesis is conducted under the great supervision from Lionel Ott and Marco Tognon. Your critical and constructive feedback is invaluable. Lionel: I will remember all the (virtual) coffee breaks where we chatted about research and life. Thank you Marco for all the fruitful discussions and the all-nighters we pulled together for the papers. I also thank Juan Nieto for his vision, optimism and calmness during my first year at ASL. I also received a lot of guidance and advices during my PhD, I am especially thankful to Markus Hehn, Philipp Reist, Margarita Grinvald and Edo Jelavic.

I feel privileged to experience two excellent labs during my PhD (ASL and IDSC), both of which are full of smart and interesting people. I would like to thank Max Brunner and Karen Bodie for introducing me into the great MAV team, Michael Pantic for the crazy and interesting discussions and ideas, Christian for all the satisfying Rising Sun visits, Eugenio for the help with experiments and joyful discussions and David for the joint supervision of the Drogone focus project. To the fixed-wing members: TJ, Michael, Florian, and Jay the field tests in Hinwil were challenging and fun. Rajan: it was a pleasure to TA the course Dynamics Programming and Optimal Control with you. I will forever remember the nights with the lecture notes and the food at Lucky Dumplings. I would also like to thank Michael Muehlebach and Nick for their help by carefully reading my manuscripts and providing constructive feedback. Big thanks to the IDSC folks patiently discussing my research: Mike Hamer, Tony, Carlo and Matthias. Also huge thanks to my office mates who created a great and fun environment to work in, in particular, Max Kriegleder, Rik, Tim, Chantel, Alex and Mike Allenspach. I would like express my gratitude to Helen Hanimann-Robinson for her emotional support and the administrative help from Lucy, Cornelia, Michael Riner and Katharina Munz. I am also grateful for all the after-work activities which made me feel welcome in Switzerland: Tony: thank you for taking us to the great sauna at your parents' place; Dario and Robin: the Super Kondi and the pizza dinners are always fun; Jenjen: the dinners hosted by you are always full with laughters; of course, there are all those unforgettable ASL lab events.

I would like to take the opportunity to thank for the unconditional love of my parents, who financially supported me throughout my study and open-mindedly let me choose my road. To

the A(bel)-team, who showed me how to build a family. Together with my friends they made my life so colorful. Finally, I would like to thank for the love, support and help from my girlfriend Chenxin Nie, who is always there for me during these turbulent times.

Financial Support

The research leading to the results presented in this thesis has received funding from Swiss National Science Foundation (SNSF), National Centres of Competence in Research (NCCR) Robotics, NCCR Digital Fabrication and Armasuisse.

Contents

Abstract i

Zusammenfassung iii

Acknowledgements v

1 Introduction 1
- 1.1 A controllable flying vehicle with a single moving part 4
- 1.2 Model learning for control of omnidirectional flying vehicles 4
- 1.3 Aerial physical interaction learning control 6

2 Contributions 9
- 2.1 Part A: A controllable flying vehicle 9
- 2.2 Part B: Model learning for control of omnidirectional flying vehicles 10
- 2.3 Part C: Aerial physical interaction learning control 11
- 2.4 List of Publications 12
  - 2.4.1 Publications Included in this Thesis 12
  - 2.4.2 Related Publications 12
- 2.5 List of Supervised Students 12
- 2.6 Outreach 14

3 Conclusions and Outlook 15
- 3.1 Future work 17

A. A CONTROLLABLE FLYING VEHICLE WITH A SINGLE MOVING PART 19

Paper I: Design, modeling and control of a flying vehicle with a single moving part that can be positioned anywhere in space 21
- 1 Introduction 22
- 2 Modeling and dynamics 25
- 3 Linearized system and controllability analysis 31
- 4 Control strategy 37
- 5 Design 38
- 6 Resulting vehicle 42

7 Experimental results 43
8 Conclusion 44
9 Equivalent controllability tests for the reduced state system 47
10 Controllability analysis for three special cases of the reduced state system . . . 49

B. MODEL LEARNING FOR CONTROL OF OMNIDIRECTIONAL FLYING VEHICLES **57**

Paper II: Learning dynamics for improving control of overactuated flying systems **59**
1 Introduction 60
2 Related work 61
3 Platform description 62
4 Modeling 63
5 Approach 64
6 Experimental results 68
7 Conclusion and outlook 73

Paper III: Active Model Learning using Informative Trajectories for Improved Closed-Loop Control on Real Robots **75**
1 Introduction 76
2 Modeling and Problem Statement 77
3 Generation of Informative Trajectories 80
4 Application to an aerial robot: the Omav 84
5 Experimental results 85
6 Conclusion 88

C. AERIAL PHYSICAL INTERACTION LEARNING CONTROL **89**

Paper IV: Learning Variable Impedance Control for Aerial Sliding on Uneven Heterogeneous Surfaces by Proprioceptive and Tactile Sensing **91**
1 Introduction 92
2 Preliminaries 94
3 Methodology 96
4 Experiments 101
5 Conclusion 106

Bibliography **107**

List of acronyms **116**

Curriculum Vitae **119**

Chapter 1

Introduction

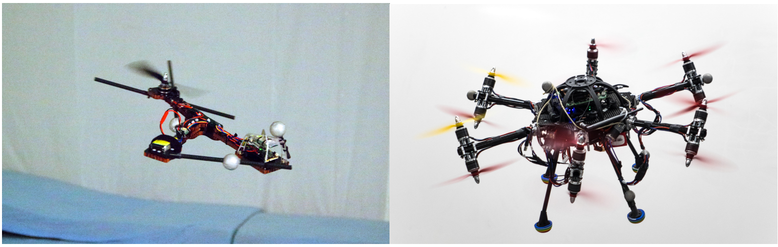
Rapid reductions in cost and advances in sensing technologies, actuation, energy storage, and computation have made aerial robots commonplace. The aerial robots market size is expected to exceed 8.5 billion US dollar by 2027 [1]. Their applications range from entertainment [2], photography, cinematography, glacier research [3], agriculture [4], industrial inspection [5] to maintenance.

Rotary wing aerial robots are capable of staying in the air by producing lift from multiple rotating propellers. They are different from the more conventional aeronautical designs such as fixed-wing aircraft [6], which rely on aerodynamic forces caused by the aircraft's forward airspeed and aerodynamic surfaces. Quadrotors are the most common ones among rotary-wing aerial robots, mainly due to their mechanical simplicity. They only have four moving parts, i.e., four motors rigidly attached to the vehicle body in a cross configuration. Their propellers are mounted away from the center of mass and pointing in a parallel direction, therefore producing a thrust of which direction is constant in the body frame. In order to translate, they have to tilt using the torque generated by the thrust difference and accelerate with the collective thrust pointing in the tilted direction [7]. Their large lever arms produce large torque and therefore high angular acceleration, which results in exceptional agility. Despite this maneuverability, their underactuation leads to a limited actuation wrench envelope and coupled translational and attitude dynamics. This is suboptimal for aerial physical interaction tasks [8], which require the robots to exert force to the environment while keeping a stable attitude. In recent years, the possibility to have aerial robots capable of interacting with the environment has attracted increasing attention. Tasks of interest include nondestructive testing inspection for bridges or chemical containers [9], house spray-painting [9], construction and drywall layouting.

Meanwhile, ongoing research keeps exploring different rotary wing designs other than quadrotors [10]. This thesis investigates two designs (Fig. 1.1): one aims to discover the minimum number of moving parts necessary for controlled flight and the other aims to address the above-mentioned shortcoming of quadrotors. To fully exploit their capabilities, these new designs require novel modeling and control approaches. This includes powerful tools from control theory such as model-based control and linear system theory [11]. Model-based control requires accurate model knowledge of the system and of the environment for high-performance control. In situations where accurate modeling using first principles[1] is difficult, controllers that learn a

---

[1] By first principles we mean the system dynamics are modeled using physical law and the parameters are empirically fitted.



(a) The Monospinner: a controllable flying vehicle with a single moving part

(b) The Omav: an omnidirectional flying vehicle with eighteen actuators

Figure 1.1: Experimental platforms in this thesis

dynamics model[2] from data (either in advance or by adapting during operation) become attractive. While the data-driven model is good at approximating the complex dynamics using data, first principle model has the advantage of good generalization [12]. This thesis aims to take advantage of both methods and thus employs a hybrid approach. That is,

> *We aim to push the limits of multicopters via novel vehicle designs and by developing methods that combine model-based control with machine learning approach to exploit the best aspects of both methods.*

Since this is a broad topic, we investigate some specific instances in this direction.

With the objective of pushing the limits of multicopters, one could ask a question: What is the minimum number of moving parts necessary for controlled flight? In addition, what tools can we use to mechanically design and control such a flying machine? In this thesis, we found out the minimum number is one and designed such a highly underactuated flying vehicle called the “Monospinner” (see Fig. 1.1(a)) using tools from linear system theory and a first-principle model-based linear controller.

It is also observed that this controller yields a non-negligible steady-state error due to modeling error despite our best modeling effort and the Monospinner’s simple mechanical structure. More generally, modeling using first principles becomes less accurate for even more complex systems. At the same time, research in aerial physical interaction tasks result in complex novel vehicle designs, for which the modeling and control are challenging [8]. The above two aspects motivates us to use an omnidirectional flying vehicle (the Omav shown in Fig. 1.1(b))) as the research platform for studying control methods combining model-based control with machine learning. Omnidirectional flying vehicles have decoupled translational and rotational dynamics and the ability to exert forces and torques in arbitrary directions. On one hand, this makes them a superior choice for physical interaction tasks compared to quadrotors. On the other hand, unlike standard quadrotors, for which a simple and accurate actuation model is available for

---

[2] By model learning we mean a black box model (e.g., a neural network) where its parameter structure bears no resemblance to the system’s physics-based dynamic modeling.

low speed maneuvers [7], only an inaccurate and simplified actuation model can be obtained for omnidirectional vehicles from first principles. This is caused by their overactuation and hence increased mechanical complexity (i.e., the number of actuators is larger than the six degrees of freedom). This disturbance from modeling error degrades their control performance, which is undesirable since the above-mentioned aerial physical interaction applications may require high-precision trajectory tracking performance. For example, drywall layouting during construction requires drawing a line with a precision of 1 mm. In addition, external model uncertainties of the interacting environments act as another disturbance during aerial interaction tasks. For instance, when equipped with an end effector which slides on a surface, the Omav experience discontinuous disturbance caused by the inaccuracies in the given surface map such as discontinuous change in geometry and unknown, spatially-varying surface friction properties. Consequently, this may destabilize the system. The above challenges can be summarized into the following question: How to achieve high performance control for omnidirectional flying vehicles under model uncertainties in both free flight and aerial physical interaction?

To address the actuation model uncertainty problem of the Omav, this thesis proposes a method to learn the residual dynamics of overactuated systems using a stochastic process. The residual dynamics are the mismatch between the vehicle's true dynamics and the simplified first-principle model. An optimization problem is formulated to compute a compensation signal to the high-level control command (i.e., wrench command) to account for these modeling errors. An efficient and effective data collection scheme for model learning is also a pertinent problem due to the high-dimensional and continuous state space of a robotic system like Omav. To address this problem, a sampling-based informative trajectory planning method is proposed. The metric for the informativeness of the trajectory is the uncertainty provided by a learned stochastic process dynamics model.

This thesis also presents a reactive adaptive control strategy for the Omav under external model uncertainty during sliding tasks. It combines proprioceptive measurements and tactile sensing to infer the geometry and friction properties of the interacting environment. This inference is used to adjust the robot's controller gain according to the varying surface properties for a good disturbance rejection. In particular, the adaptive control strategy is trained in a simulation with simplified actuator dynamics and was able to be transferred to the real robot without adaptation. This can reduce the time-consuming efforts of an accurate modeling of the actuator dynamics.

This thesis puts an emphasis on experimental investigations, which have been conducted in each individual part of the research. The experiments are performed in indoor motion capture environments, whose available measurement precision allows for isolating individual research questions and focusing on the control part of these novel flying vehicles.

This thesis is divided into three parts: the first part discusses a flying vehicle with only a moving part, the second part presents methods that improve the control performance of the Omav in free flight, and the third focuses on improving the performance in aerial interaction tasks such as sliding on surfaces.

1.1 A controllable flying vehicle with a single moving part

Highly underactuated flying vehicles have the advantages of increased reliability and reduced manufacturing and maintenance costs due to their reduced mechanical complexity. At the same time, this also leads to increased difficulty in the control of their attitude and position. Therefore, many researchers have explored the aerodynamic properties and the mass distributions of different vehicle designs that make the system's attitude passively stable ([13]–[23]): if the vehicle in hover is disturbed and tilts away or moves sideways, aerodynamic forces will damp out the lateral motion and induce a restoring moment, bringing the vehicle's attitude back to its hover state and its translational velocity to zero. The vehicle's position will not recover to its position before the disturbance, which means that its position is not passively stable. While eliminating the need for attitude sensing (using onboard sensors such as gyroscope, attitude estimation, etc.) and active attitude control, this can limit the vehicle's maneuverability, as its actuators have to counteract these restoring aerodynamic forces and moments to achieve controlled forward flight.

This thesis presents a different approach for an active position and attitude control: a highly underactuated vehicle called the "Monospinner" (see Fig. 1.1(a)) is designed without relying on aerodynamic effects (apart from the airframe drag torque and the propeller) or passive attitude stability (Paper I [P1]). It has a single moving part (its rotating propeller), and its attitude is stabilized by active feedback control. While attitude sensing is required for the Monospinner, active attitude control increases the vehicle's maneuverability. The vehicle is fully controllable in position. Tools from linear system theory and optimal control are deployed for system analysis and controller design. This includes a formulation of the Monospinner's translational and attitude dynamics in a twelve dimensional state space and its corresponding equilibrium. With the linearized system matrices at hand, the system is analyzed as a whole and its controllability leads to a mathematical conclusion to whether the vehicle is controllable in position. It is proven that the full twelve state system is not stabilizable for any vehicle configuration. However, the system may be fully controllable in position after removing the yaw state, as it does not affect the dynamics of other states. This reduced eleven state system is thus investigated. Specifically, three types of vehicle configuration under simplifying assumptions are analyzed, giving guidelines for the mechanical design of the vehicle. A linear, time-invariant controller is designed to control the hovering vehicle, and a vehicle design is found by optimizing primarily for the vehicle's mass distribution. Two robustness metrics are chosen: the ability to maintain hover under perturbations and the probability of input saturation based on a stochastic model. Experimental results show that the resulting vehicle is not only able to hover, but also has a large region of attraction such that it recovers after being thrown into the air like a frisbee.

1.2 Model learning for control of omnidirectional flying vehicles

Omnidirectional vehicles designs (e.g., the Omav in Fig. 1.1(b)) often cause complex aerodynamic effects that are non-negligible and hard-to-model from first principles, such as the aerodynamic interference between the rotors. Furthermore, manufacturing imperfections in the electronic speed controllers and motors lead to differences in thrust mapping and therefore significant torque model mismatches due to the large lever arm. These mapping differences are also hard to identify in situ or on an external measurement device that replicates the same propeller clearance in flight. This is disadvantageous from a control perspective. A standard control strategy [24] for overactuated flying vehicles includes an outer loop controller which first computes a desired wrench to reduce tracking error. This wrench is transformed into individual actuator commands by inverting the actuation model. Due to unmodeled effects, this typically results in a different actual wrench than the desired one. This leads to degraded control performance in free flight and reduces the ability to accurately track desired trajectories.

To solve the problem mentioned above, adding only integral actions in the feedback controller is often not enough, as they typically decrease the stability of the system and are not able to react to fast-changing modeling errors.

Another solution is to treat the unmodeled dynamics as disturbances and employ a disturbance observer [25], [26]. While being computationally efficient, such a reactive strategy introduces delays in tracking. One alternative solution is to identify these mismatched dynamics and compute additional feedforward signals for the standard outer loop controller in an attempt to cancel out the effect of modeling errors. This mismatch can be modeled using data-driven machine learning methods. Such methods have recently been shown to be capable to learn complex and nonlinear dynamic functions [27]. Directly learning an inverse mapping of the actuation, that is, a mapping from the wrench to the individual actuator commands, is a commonly used method [24]. However, this approach is problematic for an overactuated system due to its one-to-many mapping between wrench and actuator inputs, which is of multi-modal nature. It has the risk of producing invalid results when averaging over multiple distinct modes with standard inverse model approaches.

This thesis presents an approach (Paper II [P2]) that addresses the actuation modeling uncertainties of overactuated systems which includes omnidirectional flying vehicles. It aims to improve the trajectory tracking performance of overactuated flying systems by augmenting model-based control methods with model learning. The actuation modeling error is learned offline using a Gaussian process (GP) regressor. At each control step, a wrench command is optimized using the analytical forward model and its learned error model such that the achieved wrench is equal to a given desired wrench from the outer loop controller. An iterative optimization problem is formulated to overcome the challenge arising due to the above-mentioned one-to-many mapping, necessitating no inverse models.

A challenge related to model learning of dynamic systems is the training data collection process. A high-quality learned model relies on good representative data. The training data often has a different distribution than the test data due to several reasons: First, model uncertainties and feedback controller might lead the system to a state not encountered in a previous data collection routine. Secondly, the closed-loop dynamics change as the model used by the controller is updated. Finally, given partial model knowledge, the region of the state space that leads to the best performance is a-priori unknown.

A straightforward approach is to perform a large number of experiments to cover as much of the input space as possible during training. However, for robotic systems with a high-dimensional and continuous state space, the search space typically is too large to be covered exhaustively. Furthermore, the dynamics can change significantly during consecutive experiments, e.g., the crash of a flying vehicle could damage its motors and invalidate the previous training data. Even when considering a specific task, a good model is required in the working area of the state and input spaces, which still might be large. Thus, it is desirable to have an efficient scheme to collect training data locally around the desired task. As these learning techniques are nonparametric, common tools from parametric system identification [28], e.g., persistence of excitation, are not applicable.

This thesis proposes an active learning approach (Paper III [P3]) to exploit the statistical information captured by the GP model to infer the most informative region, thus improving sampling efficiency and efficacy. We then generate an informative trajectory that reduces the overall uncertainty in the estimated region. This trajectory is then executed in the real world to collect data. More specifically, in a first step, possible informative locations are inferred in simulation from the previously learned model. Then, different informative trajectories are sampled and evaluated according to a cost metric defined as the integral of the predictive uncertainty over these possible locations. The most informative trajectory is then selected and executed on the real robot to collect data. As a result, the model learned from this informative trajectory leads to improved trajectory tracking performance and a better generalization of the learned dynamics model. The latter is achieved because the informative trajectory reduces the uncertainty over a large region of state and input space.

1.3 Aerial physical interaction learning control

Aerial interaction tasks such as contact-based non-destructive inspections [29]–[31] require the flying vehicle to carry a sensor payload and fly along the surface while maintaining contact between the sensor and surface. This results in a sliding movement of the flying vehicle's end effector across the surface of a static environment (push and slide task defined in [8]). Many existing approaches that solve this task assume a perfect surface map is given, the surface end effector pair has constant friction coefficient, and a desired high-level pose trajectory is planned relative to the surface [30], [32]–[34]. Despite these successful demonstrations of aerial sliding on homogeneous, continuous surfaces in the aforementioned works, the following challenges remain to be investigated: 1. sensing of the inaccuracies in the given surface map such as discontinuous change in geometry and unknown, spatially-varying surface friction properties ; 2. control of the aerial vehicles in the presence of these disturbances. Firstly, the surface friction properties may vary spatially (i.e., heterogeneous surface) and cannot be directly measured. Unknown discontinuous geometry of the surface (steps, holes, curvature) can be at best partially observed using perception sensors such as a camera or a 3D time-of-flight camera due to occlusion or spatial resolution respectively. The aforementioned perception sensors could also fail on transparent surfaces or under adverse light conditions. Secondly, from the control perspective, the presence of these unexpected environment features introduces discontinuities in contact forces. The induced torque (due to the lever arm between the contact point and the center of mass) in turn causes abrupt changes in attitude and destabilizes the system. Thus, a control strategy is needed to adapt to the uncertain environment and ensure a stable flight.

To the authors' best knowledge, there is no prior work on aerial sliding on uneven, heterogeneous surfaces. Previous works in the fixed-base manipulation community demonstrated sliding on uneven surfaces using passivity [35], or adaptive force control [36], [37]. These approaches are not directly transferable to an aerial vehicle as they typically rely on repetitive executions of the same trajectory to improve their performance in trajectory tracking. For aerial robots, these might lead to instability in the first trial and thus damage the robot. One could also resort to disturbance observer-based robust control [38], which may become overly conservative. Such an approach also has the disadvantage of being slow to react, especially in the presence of noisy measurements and inaccurate process model. Consequently, the flying vehicle can struggle to handle abrupt changes in the environment. Finally, another option is to use mechanical compliance of the flying machine's end effector [29] [39], this increases its mechanical complexity and cost and further decreases its limited payload.

A novel approach to address the aforementioned sensing challenge is to use proprioceptive measurements and wrench sensing to infer the geometry and friction properties of the interacting environment. This type of approach is found mainly in quadrupedal robots like MIT Cheetah [40] and Anymal [41]. They use either command signals to infer a leg touchdown event or use the IMU signals, robot states, and control commands to infer the surface properties. They do not rely on RGB-D or Lidar sensors and are operative under adverse light conditions. Furthermore, Lee et. al. [41] use reinforcement learning to learn a policy from simulation and a student teacher approach [42] is deployed for better learning efficiency.

From a control perspective (the aforementioned second challenge), impedance control [43] is a suitable control strategy for physical interaction tasks. It introduces algorithmic compliance into the system such that the flying vehicle is less susceptible to the interaction disturbances. This has been successfully demonstrated to work for sliding tasks on homogeneous surfaces with aerial vehicles [32], [34]. Selection of the impedance gains is a trade-off between controller tracking performance and system compliance. In general, the impedance controller should have a high impedance (high tracking performance and low compliance) only when the task requires it. For this reason, a variable impedance controller [44] is an attractive option since it adaptively selects the impedance gains. In [32], for example, the controller chooses the apparent mass of the controller depending on the distance to the interacting surface. In free flight, it has a higher apparent mass for disturbance rejection. During interaction, it has a lower apparent mass for more compliance. Other applications are found on fixed-base manipulators for disturbance rejection using Lyapunov theory-based adaptive control [45], or quadrupedal walking robots [46]. With recent advances in machine learning, methods combining reinforcement learning with variable impedance control have been proposed, with examples in manipulator control on peg-in-hole tasks [47], [48], human-robot collaboration [49], and hopper jumping [50]. There exists also other learning methods such as using Gaussian processes [51]. Some of these approaches also adapt the reference trajectory in addition to variable impedance [45], [47], [48], [50].

This thesis (Paper IV, [P4]) proposes an approach that aims to address the challenge of environmental uncertainty during physical aerial interaction for omnidirectional flying vehicles. A local reactive strategy (a neural network mapping) is designed to augment the existing impedance controller on the Omav. It adapts the impedance gain according to system signals. These include control signals, proprioceptive sensor measurements, and wrench sensing, from which the strategy implicitly infers the change of the geometry and friction coefficient of the surface in contact. Based on this knowledge, it then changes the controller parameters to keep the end effector's orientation steady under disturbances. The training of this mapping is conducted entirely in a simplified simulation and divided into two stages: in the first step, a teacher policy is devised using ground truth information about the surface properties, which is not available during execution in novel environments. The teacher policy then guides the student policy to use proprioceptive measurements instead of ground truth information via supervised learning. This significantly improves the learning efficiency. Furthermore, RGB-D or lidar sensors are not required with this approach. The learned mapping from simulation can be directly transferred to the aerial vehicle. A key distinction with other sim-to-real approaches (e.g., [27]) is that we instead use a simplified simulation and demonstrate that the control strategy can be transferred to the robot. This is largely due to the fact the existing controller structure suppresses model uncertainty and keeps the reality gap small. This is particularly advantageous for a complicated system like the Omav [24], as shown in Fig. 1.1(b), where a large amount of training is required for an accurate model learning of the whole body dynamics. As a comparison, a million samples are required for the modeling of a single one degree of freedom actuator [27]. Furthermore, if the Omav crashes or if its configuration changes, training data needs to be recollected again. Finally, compared to end-to-end learning approach (learn a mapping from the state space to the actuator space), our approach is easier to train and more data efficient, as evidenced by the work of [48].

Chapter 2

Contributions

In this chapter, we summarize the contributions resulting in the publications included in this thesis.

2.1 Part A: A controllable flying vehicle

Publication I

[P1] Weixuan Zhang, Mark W. Mueller, and Raffaello D'Andrea, "Design, modeling and control of a flying vehicle with a single moving part that can be positioned anywhere in space". In *Mechatronics*, *61*, *117-130*, 2019.

Contribution

This paper follows previous work presented at a conference [52] and extends these previous results by presenting:

- a twelve-dimensional state-space system description for the Monospinner, for which an equilibrium exists and where techniques from linear time-invariant system theory may be applied for system analysis and control design,
- a proof that the twelve-dimensional linearized system about hover is not stabilizable for any vehicle configuration,
- controllability analysis of the reduced eleven-dimensional linearized system (with yaw state removed) for three special types of vehicle configuration,
- the experimental results with a controller designed using the proposed linear system model, which enables the resulting vehicle to move anywhere in space.

2.2 Part B: Model learning for control of omnidirectional flying vehicles

Publication II

[P2] Weixuan Zhang, Maximilian Brunner, Lionel Ott, Mina Kamel, Roland Siegwart, and Juan Nieto, "Learning dynamics for improving control of overactuated flying systems". In *IEEE Robotics and Automation Letters, 2020, Volume 5, Issue 4*, 2020.

Contribution

Following contributions are made:

- Presentation of a GP-based model to capture the model-plant mismatch common among overactuated omnidirectional flying vehicles.
- Introduction of an optimization-based method using a first-principles model and a learned GP error model to select a signal correcting for the model-plant mismatch, necessitating only forward models.
- Experimental validation of the proposed approach on the Omav with a reduction of the attitude tracking error of 32% on average.

Interrelations

The model learning framework of this paper is also used for the [P3]. In particular, the model uncertainty provided by the model is used to infer region of training data collection in [P3].

Publication III

[P3] Weixuan Zhang, Marco Tognon, Lionel Ott, Roland Siegwart, and Juan Nieto, "Active Model Learning using Informative Trajectories for Improved Closed-Loop Control on Real Robots". In *Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

Contribution

The contributions of this paper are summarized as follows:

- A formal mathematical formulation of the problem of efficient data collection for learning dynamics model.
- A practical strategy to *efficiently* collect task-relevant data that improves the model-based control performance when used to update the learned model.
- Real experimental results conducted on a complex overactuated omnidirectional flying system with nonlinear dynamics and 18 actuators. For a figure-8 trajectory, two runs of trajectory flight lead to an angular acceleration tracking error reduction of 54.4%.

2.3 Part C: Aerial physical interaction learning control

Publication IV

[P4] Weixuan Zhang, Lionel Ott, Marco Tognon, and Roland Siegwart, "Learning Variable Impedance Control for Aerial Sliding on Uneven Heterogeneous Surfaces by Proprioceptive and Tactile Sensing". In *IEEE Robotics and Automation Letters*, 2022.

Contribution

The contributions made in this report include the following:

- A learning-based solution for aerial sliding tasks that senses, adapts, and remains robust against challenging interaction environment uncertainties in surface geometry and friction properties.

- An approach to address sim-to-real transfer by including a closed-loop controller to suppress model uncertainty, which allows for learning from simplified actuator dynamics.

2.4 List of Publications

My doctoral studies resulted in or contributed to the following publications. The list is sorted by year and first author's name.

2.4.1 Publications Included in this Thesis

[P1] W. Zhang, M. W. Mueller, and R. D'Andrea, "Design, modeling and control of a flying vehicle with a single moving part that can be positioned anywhere in space", *Mechatronics*, vol. 61, pp. 117–130, 2019

[P2] W. Zhang, M. Brunner, L. Ott, M. Kamel, R. Siegwart, and J. Nieto, "Learning dynamics for improving control of overactuated flying systems", *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5283–5290, 2020

[P3] W. Zhang, M. Tognon, L. Ott, R. Siegwart, and J. Nieto, "Active model learning using informative trajectories for improved closed-loop control on real robots", in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 4467–4473

[P4] W. Zhang, L. Ott, M. Tognon, and R. Siegwart, "Learning variable impedance control for aerial sliding on uneven heterogeneous surfaces by proprioceptive and tactile sensing", *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 275–11 282, 2022

2.4.2 Related Publications

[R0] W. Zhang, M. W. Mueller, and R. D'Andrea, "A controllable flying vehicle with a single moving part", in *IEEE International Conference on Robotics and Automation*, 2016

[R1] M. Brunner, K. Bodie, M. Kamel, W. Zhang, J. Nieto, and R. Siegwart, "Trajectory tracking nonlinear model predictive control for an overactuated mav", in *IEEE International Conference on Robotics and Automation*, 2020

2.5 List of Supervised Students

During my doctoral studies, a significant effort was spent on supervising student projects. Below, all supervised students and projects are listed. For projects that contributed to a publication, a citation is provided.

Master Thesis

Master student, 6 months full time, 30 ECTS

[MT0] Stefan Walser (2019):
"Model Learning of the Dynamics Model of a Tilt-arm Omnidirectional Flying Vehicle using Neural Network"

[MT1] Ahmad Roumie (2020):
"High-performance Control using Model Learning for Tiltrotor Omnidirectional Aerial Vehicles"

[MT2] Ramon Flammer (2021)
"Design and Analysis of an Omnidirectional Fixed-wing, Tilt-rotor Hybrid MAV"

[MA3] Elias Hampp (2021):
"Control for an Omni-Directional Tilt-Wing MAV"

[MT4] Jonas Wüst (2021):
"State Estimate of an Underwater Robot using DVL"

Semester Thesis

Master student, 3-4 months part time, 8 ECTS

[ST0] Mario Gini (2017):
"Improving the throw of the Monospinner"

[ST1] Ramon Flammer (2020):
"Analysis of Aerodynamic Effects on Coaxial Rotor Configurations"

[ST2] Fausto Tapia (2020):
"Sensing and Characterization of the Motor Dynamics on an Omnidirectionaly Flying Vehicle"

[ST3] Daniel Steinmann (2021):
"Rotor Speed controller for Full-pose Control of Tiltrotor MAVs"

Bachelor Thesis

Bachelor student, 3-4 months part time, 15 ECTS

[BT0] Christoph Demuth (2018):
"Improving the Throw of the Monospinner using Model Predictive Control and Controller Parameter Tuning"

[BT1] Jonas Wüst, Andrej Studer (2019):
"Design and Control of an Omnidirectional Underwater Marsupial Robot System for Inshore Operations"

[BT2] Fabian Lyck (2020):
" Tilt Arm Actuator Feedback on an Omnidirectional Flying Vehicle"

[BT3] Lukas Hänsli (2020):
"Design and Testing of a Novel Servo Actuator with Near Zero Backlash and Slip Ring for Infinite Rotation Capability"

Focus Project

10 Bachelor students, 12 months, part time, 14 ECTS

[FP0] Andrea Bale, Felix Stadler, Ian Boschung, Jonathan Becker, Luca Strässle, Lukas Hänsli, Michael Baumgartner, Nasib Naimi, Sarah Steiner, Severin Laasch (Summer 2019 - Summer 2020):
"DroGone: Design of an Multicopter that Autonomously Detects, Tracks, and Catches a Consumer Drone and Bring it back to Safety"

"Perception and Learning for Robotics" course project

2 Master students, 3-4 months part time, 4 ECTS

[PLR0] Pietro Zullo, Leon Locher (2021):
"Robot Dynamics Model Learning with Uncertainty"

"Studies on Mechatronics"

Bachelor student, 3-4 months part time, 5 ECTS

[SM0] Christoph Demuth (2018):
"Reduction of Computational Complexity in Model Predictive Control with Move Blocking"

Visiting Students and Summer Projects

[In0] Peter Werner (2019):
"Model learning using locally weighted projection regression"

[In1] Yilun Wu (Summer 2016):
"Embedded system toolchain porting for the Monospinner"

2.6 Outreach

The Monospinner was reported by BBC News, IEEE Spectrum, The Verge, Gizmodo, Daily Mail and presented by Raffaello D'Andrea at TED2016, Vancouver. A Youtube video is made to introduce this novel flying vehicle and has been viewed for more than 350 000 times.

Chapter 3

Conclusions and Outlook

This thesis presents a controllable flying vehicle with a single moving part, along with its controller design and controllability analysis using linear system theory. Furthermore, we presented a framework that uses Gaussian processes to model the system's residual dynamics. When embedded as an optimization problem in the controller, this model significantly improves trajectory tracking performance. This is demonstrated via the Omav on a complex trajectory. In addition, the learned model outputs uncertainty, from which a method is derived to prevent destabilizing the system in case of highly uncertain prediction of the model. An efficient data collection procedure is devised using an optimization problem formulation to find an informative trajectory. Finally, an adaptive control strategy is designed for aerial sliding tasks. It adapts the robot's impedance controller gain to reject disturbance caused by the environment uncertainties such as spatially changing, unknown friction properties, and local surface unevenness. This strategy is trained from a simulation with simplified actuator dynamics via reinforcement learning and can be successfully transferred to the real-world robot without further adjustment. While the above research is general in nature, we always have hardware in mind. Hardware improvement will help to improve the transferability of the proposed methods, as shown by the following example:

The next generation of the Omav

In [P2], the proposed method learns the model plant mismatch of the Omav with all the model uncertainties lumped together, while in [ST1] [ST2] [BT2], investigations have been made to discover the root causes of these model plant mismatches regarding the aerodynamics, the open-loop power train, and the tilt arm mechanism, respectively. The conclusion was that the individual differences in the power train due to manufacturing imperfection dominate the model plant mismatch. This type of modeling error is instance/component dependent. Even with a single power train replacement, the modeling error is highly likely to change. This hinders the transferability of the learned model. One may address this issue by equipping the Omav with a closed-loop power train (i.e., each individual rotor speed is regulated in closed-loop). This makes sure that each rotor is rotating at the desired rotor speed and results in the elimination of the power train differences. This rotor speed feedback also helps to identify the rotor dynamics, which improves the model knowledge for model-based control. This approach also helps to isolate and identify the less dominating model mismatch sources, such as the aerodynamic

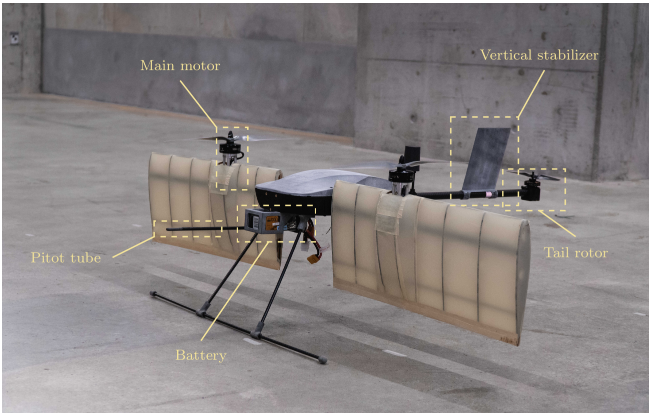


Figure 3.1: A tilt-arm omnidirection tilt-wing flying vehicle. Its wings can be separately controlled.

interference between neighbouring rotors and servo backlash in the tilt arm. These sources of model plant mismatch might be otherwise treated as noise in the model regression. Recognizing them would help to push the Omav control performance to the next level.

At the time of writing of this thesis, the next iteration of the Omav with velocity controlled electronic speed controller is already in preparation and has yet to make its maiden flight.

Long endurance physical aerial interaction

Autonomous aerial manipulators that can fly long distances increase operational coverage and thus reduce operational cost of infrastructure maintenance in remote areas . For this purpose, initial works [MT2] [MA3] demonstrated a hybrid vehicle prototype called “Soliro” (Fig. 3.1). It combines a tilt-wing design with an omnidirectional flying vehicle.

This vehicle has a tricopter configuration (two main rotors and a tail rotor) with a favorable axis of rotation (the axis connected by its two main rotors, i.e., its pitch axis). Two identical symmetrical wings are rigidly placed underneath the main rotors so that their chord line is perpendicular to the propeller disk. The main rotors and wings can be individually actively tilted along the pitch axis. Depending on the wing tilt angle at steady state, it has three modes: hover, transition, and fixed-wing. Soliro has in total five actuators.

Initial work [MT2] designed the airfoil and conducted wind tunnel tests for the system identification of its aerodynamic model. This is followed by [MA3] which designed a unified control allocation strategy and a cascaded controller for this flying vehicle. Real flight tests have been successfully conducted.

This results in a theoretical range of 27 km. From real test flights, Soliro reduces the power consumption by 30 % flying at $9\,\mathrm{m\,s^{-1}}$ in fixed-wing mode compared to hover flight.

Many future works exists for this platform, including wind disturbance estimation, model-based control and trajectory planning.

Equipped with a pitot tube, the vehicle is able to measure a projected component of the wind speed vector. At steady state level flight, the vehicle's airspeed can be derived using this measurement together with a yaw angle estimate. A Kalman filter may be designed to estimate the aerodynamic force and torque acting on the vehicle. Based on this, a model-based controller such as a model predictive controller can be designed. The model knowledge helps to improve control allocation and disturbance rejection and automate the transition phase, which needs a pilot to manually input velocity and attitude set point for now.

Furthermore, given its ability to individually control its tilt wing angle, a strategy may be developed to "manipulate" the air by tilting two wings to different angles, which leads to distinct lift and drag characteristics. This can be used on purpose for high performance attitude control.

3.1 Future work

Region of attraction analysis for the Monospinner

For the Monospinner [P1], an area of additional investigation may be the analysis of the presented linear controller and the determination of the region of attraction of its resulting equilibrium. This will reveal the subset of state space in which a given controller renders the equilibrium stable. This can help us, for example, to predict whether a hand launch will be successful based on the vehicle state when it leaves the hand.

Scalable model learning for control of the Omav

The model learning method developed in [P2] has shown to be able to approximate the model-plant mismatch of the Omav and improving its control performance when the model is embedded into a gradient-based optimization to compute a correction signal.

This work can be further extended in three ways: faster computation, feature selection and actuator dynamics.

Firstly, the critical real-time requirement on aerial vehicles poses a huge challenge to the already limited computational resource onboard the Omav. Given current learned model (three GPs with 6 dimensional input and a single output) with a hundred training data points, it takes on average 75 % cpu usage (about 7.5 ms) to solve an optimization problem on a single thread. Since the computational complexity for model prediction scales $\mathcal{O}(N^2)$, where N is the number of data samples, it would be worth to investigate methods that scales better with the size of training data. While extensive literature exists for scalable GPs [59], they are not specifically tailored for real-time applications. Other computationally efficient methods include polynomial chaos expansion needs a citation here, Locally Weighted Projection Regression [60] (initial work in [In0]) or local GPs [61]. These methods all provide analytical gradient and uncertainty estimate, and can therefore readily replace the standard GPs used in [P2].

In terms of feature selection, given enough computational resource, one may consider to take the state, the measured tilt angles, and the measured rotor speed as the input to the model. Finally, a control strategy may be designed to take the dynamics of the rotor and the servo into account. They can be embedded into the optimization problem proposed in [P2], ideally in a receding horizon fashion.

Large scale computationally efficient model learning with uncertainty

In [P2], the learned model does not only provide a predictive mean, but also predictive variance for each query location which is correlated with the confidence of the learned model. This information can be utilized in two ways: Firstly, the uncertainty can be embedded as a regularization term into the cost function that is used to optimize for a correction signal that cancels out the modeling error. By tuning the regularization parameter, one can achieve a desired trade-off between canceling the modeling error and applying uncertain input. This comes at the cost of increased computational effort in optimization.

Secondly, in outdoor flights, the Omav will experience additional disturbance from wind gusts. A Kalman filter disturbance observer can be designed where it incorporates the learned model into the process model with predictive variance as process noise. In addition, The external force disturbance can be modeled as a random walk process. In this way, one can thus distinguish between internal model error (induced by the actuator) and external model error (wind gust). This distinction is important as these disturbances require separate treatments.

Part A

A CONTROLLABLE FLYING VEHICLE WITH A SINGLE MOVING PART

Design, modeling and control of a flying vehicle with a single moving part that can be positioned anywhere in space

Weixuan Zhang, Mark W. Mueller, and Raffaello D'Andrea

Abstract

This paper presents a novel type of flying vehicle called the Monospinner, which has only one moving part, the propeller, and is yet able to hover and fully control its position. Its translational and attitude dynamics are formulated as a twelve-dimensional state space system, which may be linearized to a linear time-invariant system amenable to controllability analysis, controller synthesis, and vehicle design. It is shown that the linearized system may be both horizontally and vertically controllable in position after removing its yaw state, and in particular, this is shown for the case of a vehicle with the shape of a planar object and an offset thrust location (with respect to its center of mass). The vehicle's mass distribution is designed based on two robustness metrics: the ability to maintain hover under perturbations by means of Monte-Carlo nonlinear simulation, and the probability of input saturation based on a stochastic model. Experiments are conducted for the resulting vehicle and controller. The equilibrium of the resulting system has a large region of attraction such that it recovers after being thrown into the air like a frisbee.

Published in:
Mechatronics, *61*, *117-130*, 2019
© Open access article distributed under the Creative Commons Attribution License.
DOI: 10.1016/j.mechatronics.2019.06.004

1 Introduction

Highly underactuated flying vehicles have the advantages of increased reliability and reduced manufacturing and maintenance costs due to their reduced mechanical complexity. At the same time, this also leads to increased difficulty in the control of their attitude and position. Therefore, many researchers have explored the aerodynamic properties and the mass distributions of different vehicle designs that make the system's attitude passively stable ([13] [14] [15] [16] [17] [18] [19] [20] [21] [22]): if the vehicle in hover is disturbed and tilts away or moves sideways, aerodynamic forces will damp out the lateral motion and induce a restoring moment, bringing the vehicle's attitude back to its hover state and its translational velocity to zero. The vehicle's position will not recover to its position before the disturbance, which means that its position is not passively stable. While eliminating the need for attitude sensing (onboard sensors such as gyroscope, attitude estimation, etc.) and active attitude control, this can limit the vehicle's maneuverability, as its actuators have to counteract these restoring aerodynamic forces and moments to achieve controlled forward flight.

This paper presents a different approach: a highly underactuated vehicle (called the "Monospinner" and shown in Fig. 4.1[1]) is designed without relying on aerodynamic effects (apart from the airframe drag torque and the propeller) or attitude passive stability. It has a single moving part (its rotating propeller), and its attitude is stabilized by active feedback control. While attitude sensing is required for the Monospinner, active attitude control increases the vehicle's maneuverability. The vehicle is fully controllable in position. To the best of the authors' knowledge, there exist only two types of vehicles (the other one is the Maneuverable Piccolissimo [20]) that are both horizontally and vertically controllable with only one moving part.

This article includes a formulation of the Monospinner's translational and attitude dynamics in a twelve dimensional state space and its corresponding equilibrium. With the linearized system matrices at hand, the system is analyzed as a whole and its controllability leads to a definitive answer to whether the vehicle is controllable in position. It is shown that the full twelve state system is not stabilizable for any vehicle configuration. However, the system may be fully controllable in position after removing the yaw state, as it does not affect the dynamics of other states. This reduced eleven state system is thus investigated. Specifically, three types of vehicle configuration under simplifying assumptions are analyzed, giving guidelines for the mechanical design of the vehicle. A linear, time-invariant controller is designed to control the hovering vehicle, and a vehicle design is found by optimizing mainly for the vehicle's mass distribution. Two robustness metrics are chosen: the ability to maintain hover under perturbations and the probability of input saturation based on a stochastic model. Experimental results showed that the resulting vehicle is not only able to hover, but also has a large region of attraction such that it recovers after being thrown into the air like a frisbee.

1.1 Related work

A vehicle similar to the Monospinner is the Maneuverable Piccolissimo: it also features only one moving part (the propeller) and one actuator and is yet fully controllable in position. While aiming for small size (the vehicle is 39 millimeters in its largest dimension and 4.47 grams in

---

[1] A video showing the Monospinner can be found under `https://youtu.be/P3fM6VwXXFM`

weight), the authors designed the vehicle's mass distribution and relative rotor speed to achieve passive stability in attitude. With an offset between its thrust location and the center of mass, the whole body rotates in the air with a small tilt angle. Horizontal control is achieved by modulating its thrust at a rate of once per body revolution and thus creating net moments and forces that control its roll, pitch and position.

Highly-underactuated flying machines can be categorized into several subgroups: The first category is the samara-type vehicle, which can be traced back to the 1950's [62] and is also referred to as the Monocopter. Inspired by the maple seed (or samara), the vehicle's whole body is similar to that of a samara or a single wing and rotates around the vertical axis during flight. Rotation is usually achieved by the thrust produced by a propeller mounted at one end of the body, and the lift created by this rotation counterbalances the vehicle's weight. Through proper vehicle design, Monocopters become passively stable in attitude [63] and can hover for a trimmed open loop control input. With a servo-driven control surface installed on the wing, they may be controllable in the horizontal plane. Thus, they require two actuators to be fully controllable in position. Notable references are [14] [15] [16] [17] [18] [19], which focused on aspects related to the modeling, design, and control of the Monocopters. A more detailed study and modeling on the Monocopter's system dynamics, especially regarding its aerodynamic properties, can be found in [64] and [65].

Vehicles in the second category are equipped with one actuator (a rotating propeller), providing thrust in the vertical direction and inducing body rotation around the vertical axis, while aerodynamic dampers are installed to make sure that they are passively stable in attitude. The thrust produced goes through the center of mass and can only control the height of the vehicle. Such vehicles are presented in [13] [20], while similar vehicles exist as toys, for example the Air Hogs Vectron [66] or Flower Flutterbye Fairy [67].

The third category is the flapping-wing flying vehicle. Biologically inspired, their main propulsion comes from the flapping of a pair of wings, and aerodynamic dampers are often installed to ensure passive attitude stability. In [21], [22], the presented flying vehicles have one

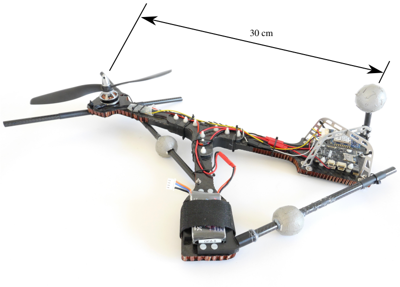


Figure 4.1: The Monospinner is approximately 30 cm in size, the frame consists of five carbon-fiber plates, and the electronics are mounted in an aluminium cage. The carbon fiber rods help to protect the propeller during landing. A more detailed list of components is given in Table 4.1.

actuator and are only controllable in height. In [68], [69], [70], the flying vehicles have at least two actuators to achieve controlled forward flight.

Traditional small scale helicopters are not passively stable in attitude and require servo-controlled swashplates for attitude control, which results in at least three actuators. In [71], the authors presented a coaxial helicopter that uses only two actuators to control the vehicle's roll, pitch, and yaw orientation, as well as maneuvering thrust. For roll and pitch control, one actuator uses a pair of passively hinged airfoil blades to mimic a conventional helicopter's cyclic control and generate torque around the roll and pitch axes. The other actuator is equipped with a conventional fixed-pitch propeller, and thrust and yaw control are achieved by the collective thrust and the differential propeller reaction torque of these two actuators. In [13], the author presented a prototype called the UNO that uses the same passive hinge mechanism to achieve horizontal, roll, and pitch control. It has one actuator (the motor) and three moving parts (the passively hinged propeller).

Another category is the flying vehicle with no moving parts. These are actuated by an ionic jet engine, which produces thrust by emitting positively charged ions and harvesting momentum from their collisions with a neutral fluid. In [72], a robotic airfish with an ionic jet and plasma ray propulsion system is presented. However, there is little information about its capabilities. In [73], the flying vehicle presented has a similar configuration to a standard quadrocopter and uses four ion thrusters (thus four actuators) instead of four propeller-based thrusters. Simulation shows controlled flight, and the vehicle prototype is able to have an open-loop, uncontrolled takeoff. Another class of vehicles with arguably no moving parts are spacecraft operating only under thrusters (e.g. lunar landers) – they typically have significant redundancy, with substantially more actuators than degrees of freedom, and thus do not fit into the category of underactuated vehicles considered in this work.

Vehicles in the last category have only fixed-pitch propellers with parallel axes of rotation as inputs, and they are fully controllable in position. In [74], [75] it is shown that a quadrocopter can maintain flight despite the complete loss of two propellers (that is, with only two propellers remaining) and in theory, control is possible after the complete loss of three propellers. The Monospinner (one propeller), the Bispinner (two propellers) [75], and the Maneuverable Piccolissimo belong to this category. The Monospinner and the Bispinner require active attitude control, whereas the Maneuverable Piccolissimo does not, since it is passively stable in attitude.

In [74], the authors derived conditions under which two degrees of freedom in attitude are controllable for three different propeller loss cases (that is, complete loss of one, two or three propellers) for a quadrocopter. They also derived in [75] a general framework for establishing attitude controllability of the vehicles in the last category and investigated a special case where a quadrocopter loses two opposing motors. In [76], a controllability test method is developed for multicopter systems with positive thrust constraints and around their conventional hover state (zero translational and rotational velocity).

This paper follows previous work presented at a conference [52] and extends these previous results by presenting:

- a twelve-dimensional state-space system description for the Monospinner, for which an equilibrium exists and where techniques from linear time-invariant system theory may be applied for system analysis and control design,
- a proof that the twelve-dimensional linearized system about hover is not stabilizable for

any vehicle configuration,

- controllability analysis of the reduced eleven-dimensional linearized system (with yaw state removed) for three special types of vehicle configuration,
- the experimental results with a controller designed using the proposed linear system model, which enables the resulting vehicle to move anywhere in space.

The remainder of this paper is organized as follows: the dynamic model of the Monospinner is given in Section 2, together with a twelve-state system description and its equilibrium solution. A linearized system is obtained and a controllability analysis is given in Section 3. A linear controller for the system is derived in Section 4, and the vehicle design based on two robustness metrics is discussed in Section 5. The resulting vehicle is presented in Section 6. Experimental results including two types of takeoff are shown in Section 7, followed by a conclusion given in Section 7.

2 Modeling and dynamics

This section provides the dynamic model for analysis and control of the Monospinner, followed by the discussion of the hover equilibrium of the resulting twelve-state system.

2.1 Dynamic model

This model is the same as the one given in [52] and summarized here for the sake of completeness. Fig. 4.2 shows some of the salient forces and quantities used in this section. The vehicle has a total mass m, and the gravity vector is denoted as $\boldsymbol{g}$. Boldface symbols like $\boldsymbol{g}$ are used throughout the paper to denote vectors in three-dimensional space. The propeller produces a thrust force of magnitude f_P in the direction of the unit vector $\boldsymbol{n}_P$. The position of the vehicle's center of mass with respect to a point fixed in the inertial frame is denoted as $\boldsymbol{s}$.

Two coordinate systems are used for the modeling: an inertial (ground-fixed) coordinate system E and a body-fixed coordinate system B. A vector expressed in a specific coordinate system is indicated by a superscript, for example $\boldsymbol{g}^E$ expresses $\boldsymbol{g}$ in coordinate system E. The body-fixed coordinate system B is oriented such that the motor arm (Fig. 4.2) is parallel with its x-axis and the propeller axis of rotation is aligned with its z-axis. The propeller force vector $\boldsymbol{n}_P^B$ is then $(0,0,1)$. The notation $(0,0,1)$ is used throughout this paper to compactly express the elements of a column vector.

The translational dynamics of the vehicle, expressed in the inertial frame E, are captured by Newton's law:

$$\ddot{\boldsymbol{s}}^E = m^{-1}\boldsymbol{n}_P^E f_P + \boldsymbol{g}^E \tag{4.1}$$

where it is assumed that the vehicle travels at low translational velocities, such that translational drag forces (such as those described in [77]) are neglected.

Let $\boldsymbol{I}_P$ denote the moment of inertia of the propeller (referred to the spin axis), and let $\boldsymbol{I}_B + \boldsymbol{I}_P$ denote the total moment of inertia of the vehicle (with respect to its center of mass). The vehicle rotates at an angular velocity $\boldsymbol{\omega}_{BE}$ with respect to the coordinate system E, where the subscript BE means the relative velocity of coordinate system B with respect to E. The propeller is located at a displacement $\boldsymbol{r}_P$ with respect to the center of mass, and its angular velocity with respect to the coordinate system E is denoted as $\boldsymbol{\omega}_{PE}$. Besides the thrust f_P, the propeller also experiences a torque of magnitude τ_P in the propeller thrust direction $\boldsymbol{n}_P$ due to the aerodynamic drag acting on the propeller blade, which is transmitted to the body through the motor. The vehicle experiences an airframe drag torque $\boldsymbol{\tau}_d$ due to the rotation of the vehicle in the air.

The angular dynamics of the system, expressed in the body-fixed coordinate system B, are formulated as:

$$\boldsymbol{I}_B^B \dot{\boldsymbol{\omega}}_{BE}^B + \boldsymbol{I}_P^B \dot{\boldsymbol{\omega}}_{PE}^B + [\![\boldsymbol{\omega}_{BE}^B \times]\!](\boldsymbol{I}_B^B \boldsymbol{\omega}_{BE}^B + \boldsymbol{I}_P^B \boldsymbol{\omega}_{PE}^B) = [\![\boldsymbol{r}_P^B \times]\!]\boldsymbol{n}_P^B f_P + \boldsymbol{n}_P^B \tau_P + \boldsymbol{\tau}_d^B \tag{4.2}$$

where $[\![\boldsymbol{a}\times]\!]$ represents the skew-symmetric matrix form of the cross product, so that $[\![\boldsymbol{a}\times]\!]\mathbf{b} = \mathbf{a} \times \mathbf{b}$ for any vectors $\mathbf{a}$ and $\mathbf{b}$ in $\mathbb{R}^3$.

Without loss of generality, it is assumed that the propeller is left-handed. The propeller's scalar speed Ω with respect to the body is usually controlled by an electronic speed controller, so that

$$\boldsymbol{\omega}_{PB}^B = (0, 0, -\Omega). \tag{4.3}$$

Note that $\boldsymbol{\omega}_{PE}^B$ in (4.2) can be decomposed as below:

$$\boldsymbol{\omega}_{PE}^B = \boldsymbol{\omega}_{PB}^B + \boldsymbol{\omega}_{BE}^B. \tag{4.4}$$

The thrust f_P produced from a stationary propeller is then assumed to be proportional to its angular velocity $\boldsymbol{\omega}_{PE}^B$ squared with the proportional coefficient κ_f [78]:

$$f_P = \kappa_f (\boldsymbol{\omega}_{PE}^B \cdot \boldsymbol{n}_P^B) |\boldsymbol{\omega}_{PE}^B \cdot \boldsymbol{n}_P^B| \tag{4.5}$$

with $\cdot$ denoting the vector inner product.

The propeller torque is assumed to be linear in the propeller thrust:

$$\tau_P = \kappa f_P \tag{4.6}$$

We neglect any potential torque effects due to blade flapping [79] or the propeller H-force [77].

It is assumed that the magnitude of the airframe drag torque $\boldsymbol{\tau}_d$ is quadratic in the vehicle's angular velocity $\boldsymbol{\omega}_{BE}^B$ [75]:

$$\boldsymbol{\tau}_d^B = - \left\| \boldsymbol{\omega}_{BE}^B \right\| \boldsymbol{K}_d^B \boldsymbol{\omega}_{BE}^B \tag{4.7}$$

where $\|\cdot\|$ denotes the Euclidean norm and $\boldsymbol{K}_d$ is a 3×3 matrix and assumed to be diagonal when expressed in the coordinate system B, which is denoted by

$$\boldsymbol{K}_d^B = \operatorname{diag}\left(K_{d,xx}, K_{d,yy}, K_{d,zz}\right). \tag{4.8}$$

It is assumed that the different propeller speeds near the operating point discussed in the paper are not significant enough to make a difference in the drag torque that the vehicle experiences. Therefore it is assumed that the propeller's contribution to the drag torque is constant and im-

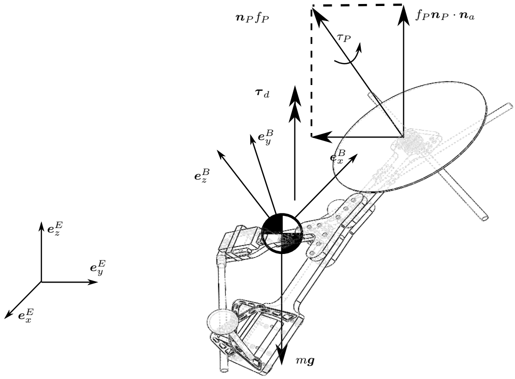


Figure 4.2: Monospinner in flight, showing some of the symbols and quantities required to model the system.

plicitly included in (4.7).

2.2 Hover solution

Similar to Section 2.1, the Monospinner's hover solution is derived in [52] and summarized here for the sake of completeness. This hover solution follows the definition of the "relaxed hover solutions" [75], which are defined as solutions that are constant when expressed in a body-fixed reference frame and where the vehicle remains substantially in one position. Specifically, these solutions allow the vehicle to have a non-zero translational acceleration (but it must average to zero) and a non-zero angular velocity.

In hover, the Monospinner's center of mass has a uniform circular motion and stays at a constant height, while the vehicle body is rotating at a constant angular velocity $\bar{\boldsymbol{\omega}}_{BE}^{B}$ in the parallel direction of gravity. Note that the overbar in this paper is always used to denote quantities that are constant in hover (i.e. the equilibrium solution). Also, a body-fixed unit vector $\boldsymbol{n}_a$ exists, which does not change when expressed in the coordinate system E. This vector may be thought of as an averaged thrust direction of the vehicle: in hover it is aligned with the thrust vector averaged over one rotation. Note that the instantaneous thrust direction may not be aligned with gravity.

Furthermore, the vector $\boldsymbol{n}_a$ is parallel to $\bar{\boldsymbol{\omega}}_{BE}$:

$$\boldsymbol{n}_a^B = \frac{\bar{\boldsymbol{\omega}}_{BE}^B}{\bar{\omega}}, \tag{4.9}$$

where $\bar{\omega}$ is the magnitude of the equilibrium angular velocity $\left\|\bar{\boldsymbol{\omega}}_{BE}^B\right\|$.

The equilibrium propeller force $\boldsymbol{n}_P^B \bar{f}_P$ can be decomposed into horizontal and vertical forces, where the horizontal force induces the circular motion and the vertical force compensates for the vehicle's weight. Thus

$$\bar{f}_P \boldsymbol{n}_P^B \cdot \boldsymbol{n}_a^B = m \left\|\boldsymbol{g}\right\|. \tag{4.10}$$

Substituting (4.9) into (4.10) yields the following solution for the equilibrium thrust

$$\bar{f}_P = \frac{m \left\|\boldsymbol{g}\right\| \bar{\omega}}{\boldsymbol{n}_P^B \cdot \bar{\boldsymbol{\omega}}_{BE}^B}. \tag{4.11}$$

In hover (i.e. setting the derivatives to zero), (4.2) becomes:

$$[\![\bar{\boldsymbol{\omega}}_{BE}^B \times]\!](\boldsymbol{I}_B^B \bar{\boldsymbol{\omega}}_{BE}^B + \boldsymbol{I}_P^B \bar{\boldsymbol{\omega}}_{PE}^B) = [\![\boldsymbol{r}_P^B \times]\!] \boldsymbol{n}_P^B \bar{f}_P + \boldsymbol{n}_P^B \bar{\tau}_P + \bar{\boldsymbol{\tau}}_d^B. \tag{4.12}$$

Note that the quantities $\bar{\boldsymbol{\omega}}_{PE}^B$, $\bar{f}_P$, $\bar{\tau}_P$ and $\bar{\boldsymbol{\tau}}_d^B$ are uniquely defined by $\bar{\Omega}$ and $\bar{\boldsymbol{\omega}}_{BE}^B$ (see (4.3), (4.4), (4.5), (4.6), (4.7)), such that we have four equations in four unknowns. The hover solution is therefore defined by the $\bar{\Omega}$ and $\bar{\boldsymbol{\omega}}_{BE}^B$ that solve (4.11)-(4.12). With the resulting $\bar{\Omega}$ and $\bar{\boldsymbol{\omega}}_{BE}^B$ (if they exist) all other quantities in hover (such as $\boldsymbol{n}_a^B$ or $\bar{f}_P$) may be calculated.

2.3 Equilibrium

In this section two frames (see Fig. 4.3) are introduced: a body frame convenient for the controllability analysis and control design, and a rotating reference frame for obtaining attitude equilibrium. Translational and attitude equilibrium is solved using the hover solution in Section 2.2.

Attitude equilibrium

For convenience, a body-fixed C-frame is introduced such that

$$\boldsymbol{n}_a^C = \boldsymbol{R}^{CB} \boldsymbol{n}_a^B = (0, 0, 1) \tag{4.13}$$

Note that (4.13) remains valid if the C-frame rotates around its z-axis. This degree of freedom may be fixed by the constraint that the propeller thrust direction $\boldsymbol{n}_P^C$ has no y component when expressed in the C-frame, that is,

$$\boldsymbol{n}_P^C = \boldsymbol{R}^{CB} \boldsymbol{n}_P^B \overset{!}{=} (*, 0, *). \tag{4.14}$$

Let $(p, q, r) := \boldsymbol{\omega}_{CE}^C$ be the body rates expressed in the C-frame. By (4.9) and (4.13) the

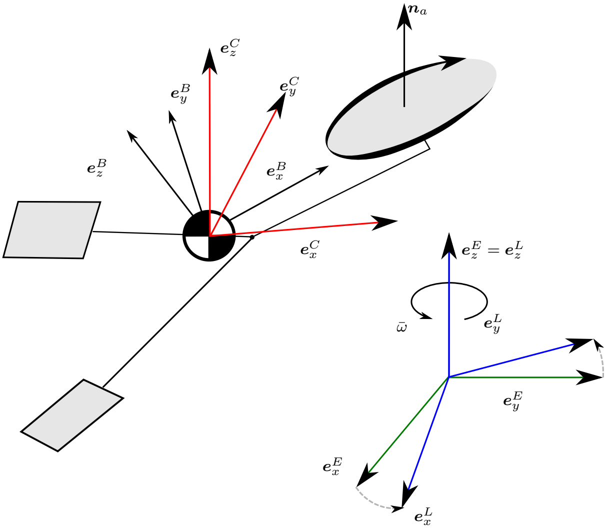


Figure 4.3: This figure illustrates the two frames introduced in Section 2.3: the body-fixed C-frame is introduced such that the body-fixed unit vector $\boldsymbol{n}_a$ is aligned with its z-axis, and the propeller force vector $\boldsymbol{n}_P^C$ has no y-component. The L-frame rotates at a constant angular speed $\bar{\omega}$ around the gravity vector and therefore the z-axis of the inertial frame E.

body rates equilibrium $\bar{\boldsymbol{\omega}}_{CE}^C$ is

$$\bar{\boldsymbol{\omega}}_{CE}^C = \bar{\boldsymbol{\omega}}_{BE}^C = \boldsymbol{R}^{CB}\bar{\boldsymbol{\omega}}_{BE}^B = \boldsymbol{R}^{CB}\boldsymbol{n}_a^B\bar{\omega} = (0, 0, \bar{\omega}). \tag{4.15}$$

In other words, at equilibrium the body-fixed C-frame is rotating at a constant angular speed $\bar{\omega}$ about the gravity vector and the yaw angle between the C and the E-frame increases linearly with time. In order to have a constant yaw equilibrium, a frame L rotating at a constant angular speed $\bar{\omega}$ around the gravity vector is introduced with

$$\boldsymbol{\omega}_{LE}^L = (0, 0, \bar{\omega}). \tag{4.16}$$

Then the vehicle's orientation may be represented by $\boldsymbol{R}^{CL}$, which relates the body-fixed frame C and the frame L. We parametrize the rotation matrix $\boldsymbol{R}^{CL}$ through the Euler Yaw-Pitch-Roll sequence, following the common aerospace convention [80], with ϕ (roll), θ (pitch), and ψ (yaw):

$$\boldsymbol{R}^{CL} = \boldsymbol{R}_x(\phi)\boldsymbol{R}_y(\theta)\boldsymbol{R}_z(\psi) \tag{4.17}$$

where

$$\boldsymbol{R}_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \tag{4.18}$$

$$\boldsymbol{R}_y(\theta) = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \tag{4.19}$$

$$\boldsymbol{R}_z(\psi) = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.20}$$

In hover, it is clear from (4.15) and (4.16) that there is only a constant yaw offset (the equilibrium yaw angle) between the C-frame and the L-frame. Therefore, the equilibrium pitch and roll angles are both zero, that is, $\bar{\theta} = \bar{\phi} = 0$. Note that the equilibrium yaw angle ($\bar{\psi}$) depends only on the choice of the initial yaw between the L and E-frame and is therefore set to zero without loss of generality. The rotation matrix $\boldsymbol{R}^{CL}$ may alternatively be parametrized with a 3-1-3 Euler angle sequence, consisting of spin, nutation, and precession [81]. This parametrization is popular for describing spinning bodies, but is less useful than the proposed yaw-pitch-roll sequence as it has a singularity at the equilibrium with zero nutation angle.

Translational equilibrium

Since in hover the center of mass of the vehicle is rotating in a circle at a constant height, its horizontal position and velocity are oscillatory when expressed in the inertial frame. Thus, the position and velocity states are expressed in the body frame C, and their dynamics are obtained by applying Euler's transformation on the position vector $\boldsymbol{s}$ and velocity vector $\boldsymbol{v}$:

$$\dot{\boldsymbol{s}}^C = \boldsymbol{v}^C - [\![\boldsymbol{\omega}^C_{CE}\times]\!]\boldsymbol{s}^C \tag{4.21}$$

$$\dot{\boldsymbol{v}}^C = \boldsymbol{R}^{CE}(\ddot{\boldsymbol{s}})^E - [\![\boldsymbol{\omega}^C_{CE}\times]\!]\boldsymbol{v}^C \tag{4.22}$$

$$= \frac{1}{m}\boldsymbol{n}^C_P f_P + \boldsymbol{R}^{CE}\boldsymbol{g}^E - [\![\boldsymbol{\omega}^C_{CE}\times]\!]\boldsymbol{v}^C \tag{4.23}$$

where $\boldsymbol{v} := \dot{\boldsymbol{s}}$ and (4.1) is substituted into (4.22).

Setting (4.23)'s left hand side to zero and substituting the hover solution into the equation yields

$$0 = \frac{1}{m}\boldsymbol{n}^C_P \bar{f}_P + \bar{\boldsymbol{R}}^{CE}\boldsymbol{g}^E - [\![\bar{\boldsymbol{\omega}}^C_{CE}\times]\!]\bar{\boldsymbol{v}}^C. \tag{4.24}$$

Recall that in hover the C-frame rotates about the gravity vector, thus $\bar{\boldsymbol{R}}^{CE}\boldsymbol{g}^E = \boldsymbol{g}^E$. Substituting the body rates equilibrium solution (4.15) and solving (4.24) yields

$$\bar{v}^C_y = -\frac{\bar{f}_P\, n^C_{P,x}}{\bar{\omega} m}, \quad \bar{v}^C_x = \frac{\bar{f}_P\, n^C_{P,y}}{\bar{\omega} m} = 0, \tag{4.25}$$

where $(n^C_{P,x}, n^C_{P,y}, n^C_{P,z}) := \boldsymbol{n}^C_P$, $(\bar{v}^C_x, \bar{v}^C_y, \bar{v}^C_z) := \bar{\boldsymbol{v}}^C$. The equilibrium state $\bar{v}^C_x$ is equal to 0 since $n^C_{P,y}$ is zero according to (4.14).

Setting the left hand side of (4.21) to zero, substituting the hover solution into it, and solving the equation yields:

$$\bar{v}^C_z = 0, \quad \bar{s}^C_y = -\frac{\bar{v}^C_x}{\bar{\omega}} = 0, \quad \bar{s}^C_x = \frac{\bar{v}^C_y}{\bar{\omega}} = -\frac{\bar{f}_P \, n^C_{P,x}}{\bar{\omega}^2 m}, \tag{4.26}$$

where $(\bar{s}^C_x, \bar{s}^C_y, \bar{s}^C_z) := \bar{\boldsymbol{s}}^C$.

Note that $\bar{s}^C_z$ does not appear in the equilibrium equations and is set to zero without loss of generality. The fact that the horizontal position equilibrium $\bar{s}^C_x$ and $\bar{s}^C_y$ cannot be set arbitrarily is simply a feature of choice of the state and the coordinate system it is represented in.

Equilibrium solution

In conclusion, the twelve-state equilibrium $(\bar{s}^C_x, \bar{s}^C_y, \bar{s}^C_z, \bar{v}^C_x, \bar{v}^C_y, \bar{v}^C_z, \bar{\phi}, \bar{\theta}, \bar{\psi}, \bar{p}, \bar{q}, \bar{r})$ is:

$$\begin{aligned} &\bar{s}^C_x = -\frac{\bar{f}_P \, n^C_{P,x}}{\bar{\omega}^2 m}, \quad \bar{s}^C_y = 0, \quad \bar{s}^C_z = 0 \\ &\bar{v}^C_x = 0, \quad \bar{v}^C_y = -\frac{\bar{f}_P \, n^C_{P,x}}{\bar{\omega} m}, \quad \bar{v}^C_z = 0, \\ &\bar{\phi} = 0, \quad \bar{\theta} = 0, \quad \bar{\psi} = 0, \\ &\bar{p} = 0, \quad \bar{q} = 0, \quad \bar{r} = \bar{\omega}. \end{aligned} \tag{4.27}$$

3 Linearized system and controllability analysis

In this section, the attitude kinematics for the Euler angles (ϕ, θ, ψ) that were introduced earlier are derived. The resulting twelve-state dynamic system is linearized about hover and the controllability analysis is subsequently given.

3.1 Linearization

The angular rates $\boldsymbol{\omega}^C_{CL}$ and the rates of the Euler angles $(\dot{\phi}, \dot{\theta}, \dot{\psi})$ have the following relationship [80]:

$$\boldsymbol{\omega}^C_{CL} = \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + \boldsymbol{R}_x(\phi) \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + \boldsymbol{R}_x(\phi)\boldsymbol{R}_y(\theta) \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix}, \tag{4.28}$$

the inverse mapping of which (that is, the mapping from $\boldsymbol{\omega}^C_{CL}$ to $(\dot{\phi}, \dot{\theta}, \dot{\psi})$) has the following form:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi)/\cos(\theta) & \cos(\phi)/\cos(\theta) \end{bmatrix} \boldsymbol{\omega}^C_{CL} \tag{4.29}$$

Note that

$$\boldsymbol{\omega}^C_{CL} = \boldsymbol{\omega}^C_{CE} - \boldsymbol{\omega}^C_{LE} = \boldsymbol{\omega}^C_{CE} - \boldsymbol{R}^{CL}\boldsymbol{\omega}^L_{LE} \tag{4.30}$$

Substituting (4.30) into (4.29) yields

$$\begin{bmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{bmatrix} = \begin{bmatrix}1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta)\\ 0 & \cos(\phi) & -\sin(\phi)\\ 0 & \sin(\phi)/\cos(\theta) & \cos(\phi)/\cos(\theta)\end{bmatrix}\left(\boldsymbol{\omega}^C_{CE} + \begin{bmatrix}\sin(\theta)\bar{\omega}\\ -\sin(\phi)\cos(\theta)\bar{\omega}\\ -\cos(\theta)\cos(\phi)\bar{\omega}\end{bmatrix}\right) \tag{4.31}$$

Introducing the state deviation from the equilibrium defined in (4.27)

$$x = (\delta s^C_x, \delta s^C_y, \delta s^C_z, \delta v^C_x, \delta v^C_y, \delta v^C_z, \delta\phi, \delta\theta, \delta\psi, \delta p, \delta q, \delta r), \tag{4.32}$$

defining the control input u as deviation of the motor force from the equilibrium motor force $\bar{f}_P$, and linearizing the system dynamics ((4.21), (4.23), (4.31) and (4.2)) about the equilibrium yield a linear, time-invariant (LTI) system:

$$\dot{x} \approx Ax + Bu. \tag{4.33}$$

Substituting the equilibrium solution $\bar{\phi} = \bar{\theta} = 0$ into (4.33) ($\bar{\psi}$ does not appear in the linearization), the system matrices A and B become

$$A = \begin{bmatrix} -[\![\bar{\boldsymbol{\omega}}^C_{CE}\times]\!] & I_3 & 0 & [\![\bar{\boldsymbol{s}}^C\times]\!]\\ 0 & -[\![\bar{\boldsymbol{\omega}}^C_{CE}\times]\!] & -[\![\boldsymbol{g}^E\times]\!] & [\![\bar{\boldsymbol{v}}^C\times]\!]\\ 0 & 0 & -[\![\bar{\boldsymbol{\omega}}^C_{CE}\times]\!] & I_3\\ 0 & 0 & 0 & A^C_S \end{bmatrix}, \quad B = \begin{bmatrix}0\\ m^{-1}\boldsymbol{n}^C_P\\ 0\\ B^C_S\end{bmatrix}. \tag{4.34}$$

Every entry of A in the above expression denotes a 3 by 3 matrix and every entry of B denotes a 3 by 1 matrix. A^C_S and B^C_S denote the linearization matrices of the Euler equation (4.2), and I_3 is an identity matrix of dimension 3. Note that the appearance of $\bar{s}^C_z$ in the system matrix A comes from the fact that the position state is formulated in the body frame. It does not, however, affect the controllability of the system pair (see Section 3.2, $\bar{s}^C_z$ does not appear in the matrices in (4.37), (4.38), and (4.39)).

3.2 Controllability analysis

In this section, controllability analysis for the linearized system is conducted to gain intuition of when it is possible to control the Monospinner. It will be shown that the full twelve-state system (from now on referred to as the full state system) is never stabilizable[2] , and the controllability test of the reduced eleven state system (with yaw state removed and from now on referred to as the reduced state system) is equivalent to the full rank tests of at most five matrices (two 4 × 4 matrices and three 3 × 4 matrices). The controllability analysis of three special cases for the reduced state system is subsequently given.

---

[2] In this article, controllability of an LTI system is defined to mean that for any initial state, there exists a control trajectory such that the system can be steered from that state to 0 in finite time, whereas stabilizability is defined to mean that for any initial state, there exists a control trajectory such that the system state converges to zero as time goes to infinity [82].

The full state system

Note that the matrix A in (4.34) is an upper block diagonal matrix. The spectrum of A is therefore the union of the spectra of the diagonal block matrices, that is,

$$\mathrm{spec}(A) = \mathrm{spec}([\![\bar{\boldsymbol{\omega}}_{CE}^{C}\times]\!]) \cup \mathrm{spec}(A_S^C) \tag{4.35}$$

The spectrum of the skew-symmetric matrix $[\![\bar{\boldsymbol{\omega}}_{CE}^{C}\times]\!]$ is $\{\bar{\omega}i, -\bar{\omega}i, 0\}$, with i denoting the imaginary unit. The eigenvalues of A are then divided into three categories: 0, $\pm\bar{\omega}i$ and the eigenvalues of A_S^C.

For a linear, time-invariant system, one could apply the Popov-Belevitsch-Hautus (PBH) test to investigate its controllability (Corollary 12.6.19, [83]), the pair (A, B) is controllable if and only if for all eigenvalues λ of A, the concatenated matrix $[\lambda I{-}A\ B] \in \mathbb{C}^{12\times 13}$ has full rank. This includes the case of eigenvalue 0, where the test matrix has the form $[-A\ B]$. Note that the third and the ninth column of the matrix A are zero vectors, meaning that the concatenated test matrix $[-A\ B]$ has at most rank 11 and therefore does not have full rank. The pair (A, B) is thus not stabilizable. Note that including the translational drag forces (such as those described in [77]) in (4.23) would not change the system's stabilizability, as they do not depend on the yaw and height of the vehicle and thus this does not change the rank of the test matrix $[-A\ B]$.

The reduced state system and equivalent controllability tests

Rearranging the states in (4.32) (moving the yaw state $\delta\psi$ to the last state) yields:

$$\tilde{A} = \begin{bmatrix} A_{11} & 0 \\ A_{21} & 0 \end{bmatrix}, \quad \tilde{B} = \begin{bmatrix} B_1 \\ 0 \end{bmatrix} \tag{4.36}$$

with $A_{11} \in \mathbb{R}^{11\times 11}$, $A_{21} \in \mathbb{R}^{1\times 11}$, $B_1 \in \mathbb{R}^{11\times 1}$ and 0 being the zero matrix with associated dimension. From (4.36), it can be seen that the yaw state does not affect the dynamics of other states.

Furthermore, changing the yaw state (the yaw angle between the L and the C-frame) in hover would not affect the direction of the averaged thrust, and therefore not the roll angle, pitch angle, and position in the inertial frame. This motivates investigating the controllability of the system without the yaw state, that is, the system matrix pair (A_{11}, B_1). Stabilizability of this reduced state system implies the ability of the system to maintain a relaxed hover solution while rejecting disturbances, remaining substantially at one point in space (though the yaw angle may not be able to simultaneously achieve some setpoint. Note that the stabilizability of the reduced system also implies that the yaw rate of the vehicle stays bounded.

The PBH test is then applied to the reduced system matrix pair (A_{11}, B_1). Applying the algebra outlined in 9, it is revealed that for the eigenvalue 0, the matrix $[-A_{11}\ B_1]$ has full rank if and only if the matrix $U_0 \in \mathbb{R}^{4\times 4}$ has full rank, where

$$U_0 = \begin{bmatrix} V_0 & -(A_S^C)^{\top} \\ m^{-1} n_{P,z}^{C} & (B_S^C)^{\top} \end{bmatrix} \tag{4.37}$$

with $V_0 = (\bar{v}_y^C, 0, 0)$.

Similarly, for the eigenvalues $\pm\bar{\omega}i$, since $[\bar{\omega}iI - A_{11}\ B_1]$ and $[-\bar{\omega}iI - A_{11}\ B_1]$ have the same rank (Fact 2.19.3, [83]), it suffices to investigate $[\bar{\omega}iI - A_{11}\ B_1]$, which has full rank if and only if the matrix $U_i \in \mathbb{C}^{4\times 4}$ has full rank (9), where

$$U_i = \begin{bmatrix} V_i & \bar{\omega}iI - (A_S^C)^\top \\ 0 & (B_S^C)^\top \end{bmatrix} \tag{4.38}$$

with $V_i = (1, -i, 0)$.

Finally, for the eigenvalues of A_S^C, assuming that its eigenvalues are distinct from 0 and $\pm\bar{\omega}i$ (otherwise we can check the rank of U_0 or U_i), its associated full rank tests are equivalent to the test of whether or not the matrix $U_s(\lambda) : \mathbb{C} \mapsto \mathbb{C}^{3\times 4}$ has full rank (9), where

$$U_s(\lambda) = \begin{bmatrix} \lambda I - A_S^C & B_S^C \end{bmatrix} \tag{4.39}$$

with $\lambda \in \operatorname{spec}(A_S^C)$.

In summary, the system pair (A_{11}, B_1) is stabilizable if and only if U_0, U_i have full rank, and $U_s(\lambda)$ has full rank for the eigenvalues of A_S^C whose real part is non-negative. Also note that obtaining the matrices A_S^C and B_S^C symbolically is nontrivial, since it requires the knowledge of the equilibrium solution to define the C-frame, and solving the nonlinear equations (4.11)-(4.12) symbolically for the equilibrium is in most cases very tedious, if not impossible.

Special cases for the reduced state system

In this section, special cases under simplifying assumptions are investigated to provide intuition of when the reduced state system matrix pair (A_{11}, B_1) is stabilizable. This may be useful since if the system is stabilizable for the simplified system equations, then it will be stabilizable for the actual system, provided that the modeling error is small enough. This stems from the fact that the eigenvalues of a matrix are continuous functions of its elements (Fact 10.11.9, [83]) that are also locally continuous at the model parameters. Therefore, the PBH test matrix does not lose rank for a perturbation of the system matrices that is small enough. Conversely, if the system is not stabilizable for the simplified system equations, it may still be stabilizable for the actual system, but it is very likely that large control efforts would be required to stabilize it.

First, it is assumed that the terms $\boldsymbol{I}_P^B \dot{\boldsymbol{\omega}}_{PE}^B$ and $\boldsymbol{I}_P^B \boldsymbol{\omega}_{PE}^B$ are negligible. For a typical vehicle design (that is, the vehicle is roughly the size of a quadrocopter described in [84]), the largest component of the propeller moment of inertia $\boldsymbol{I}_P^B$ (the moment of inertia around its body z-axis) is two orders of magnitude smaller than the smallest diagonal entries of the vehicle moment of inertia $\boldsymbol{I}_B^B$, and the equilibrium angular momentum term $\boldsymbol{I}_P^B \bar{\boldsymbol{\omega}}_{PE}^B$ is an order of magnitude smaller than $\boldsymbol{I}_B^B \bar{\boldsymbol{\omega}}_{BE}^B$. The Euler equation (4.2) thus becomes

$$\boldsymbol{I}_B^B \dot{\boldsymbol{\omega}}_{BE}^B + [\![\boldsymbol{\omega}_{BE}^B \times]\!] \boldsymbol{I}_B^B \boldsymbol{\omega}_{BE}^B = [\![\boldsymbol{r}_P^B \times]\!] \boldsymbol{n}_P^B f_P + \boldsymbol{n}_P^B \tau_P + \boldsymbol{\tau}_d^B. \tag{4.40}$$

It is also assumed that the vehicle's angular velocity with respect to the inertial frame is much smaller than the propeller's angular velocity with respect to the body, i.e., $\|\boldsymbol{\omega}_{BE}\| \ll \|\boldsymbol{\omega}_{PB}\|$, so that f_P is not a function of the body rates.

The following three special cases are then investigated:

Case 1

It is first assumed that the vehicle is a planar object (Fig. 4.4). The perpendicular axis theorem applies then, that is, for a coordinate system where the object is lying in the xy-plane, the sum of the moments of inertia about axis x and y is equal to the moment of inertia about axis z. Furthermore, the vehicle's inertia matrix is assumed to be diagonal in the B-frame. In summary, $\boldsymbol{I}_B^B = \text{diag}\,(\Theta_x, \Theta_y, \Theta_x + \Theta_y)$.

It is assumed that the propeller thrust location has a positive offset to the center of mass, that is, $\boldsymbol{r}_P^B = (l, 0, 0)$, with l being positive. It is also assumed that the vehicle's equilibrium pitch and roll rates are small, such that the airframe drag torque around the body x and y-axes is neglected:

$$\boldsymbol{\tau}_d^B = (0, 0, -K r_B |r_B|), \tag{4.41}$$

where K is a positive constant and r_B is the yaw rate in the B-frame. In a typical vehicle design, it is found that the terms $[\![\bar{\boldsymbol{\omega}}_{BE}^B \times]\!] \boldsymbol{I}_B^B \bar{\boldsymbol{\omega}}_{BE}^B$ and $[\![\boldsymbol{r}_P^B \times]\!] \boldsymbol{n}_P^B \bar{f}_P$ are at least an order of magnitude larger than the airframe drag torque around the body x and y-axes. A further reason for this assumption is that, intuitively, for such a fast, almost flat wobbling planar object, the gyroscopic effect and the offset propeller thrust dominate the roll and pitch rate dynamics, whereas the propeller torque has to be counterbalanced by the airframe drag torque in the body z-axis.

It is shown that in this case the reduced system matrix pair (A_{11}, B_1) is always *stabilizable* (see 10.1). This implies that a vehicle of flat shape is a viable choice when designing a Monospinner. A special case here is when the vehicle has the shape of a flat plate, that is, $\boldsymbol{I}_B^B = \text{diag}\,(\Theta, \Theta, 2\Theta)$. The Maneuverable Piccolissimo [20], for instance, has such an inertia distribution.

Case 2

It is assumed that the vehicle's inertia matrix has the form $\boldsymbol{I}_B^B = \text{diag}\,(\Phi, \Theta, \Theta)$, and the airframe drag matrix expressed in the body frame B has the form $\boldsymbol{K}_d^B = \text{diag}\,(J, K, K)$, where Φ, Θ, J, and K are non-zero. Here the vehicle's equilibrium pitch and roll rates may not be small, thus the aerodynamic effects in the pitch and roll axes cannot be neglected. As in case 1, it is assumed that the thrust location $\boldsymbol{r}_P^B$ is equal to $(l, 0, 0)$ with positive l. This corresponds to the case where the vehicle has the shape of a cylinder and the thrust location is aligned with its center axis (Fig. 4.5). Note that this case also includes the special case that the vehicle's mass distribution is symmetric, that is, $\boldsymbol{I}_B^B = \text{diag}\,(\Theta, \Theta, \Theta)$ (e.g. a sphere or cube).

It can be proved that the reduced state system is *not stabilizable*, since the PBH test matrix associated with the eigenvalues on the imaginary axis does not have full rank (see 10.2). Intuitively, the cross-coupling term (the term $[\![\boldsymbol{\omega}_{BE}^B \times]\!] \boldsymbol{I}_B^B \boldsymbol{\omega}_{BE}^B$ in the Euler equation) in the x-axis disappears due to the structure of the inertia matrix, so that the roll rate dynamics can be hardly influenced by other states. In addition, the propeller thrust only creates moment around the pitch axis. The reduced state system is therefore not stabilizable. This indicates that when designing a Monospinner, the design should avoid to have an inertia matrix similar to the one given in this case.

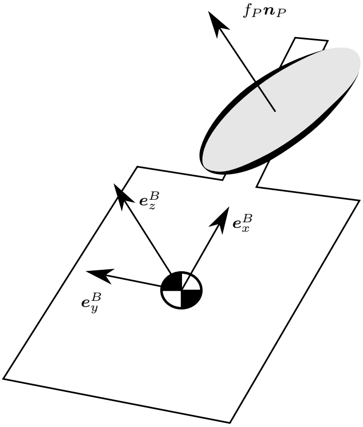


Figure 4.4: A possible shape of the vehicle in the special case 1 of the controllability analysis for the reduced state system. It is a planar object with an offset thrust location.

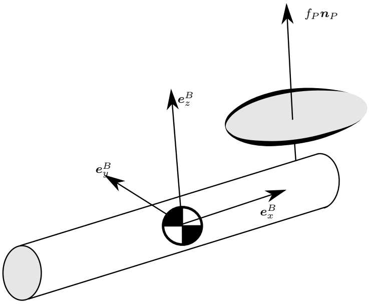


Figure 4.5: A possible shape of the vehicle in the special case 2 of the controllability analysis for the reduced state system. It has the shape of a cylinder and the thrust location goes through the cylinder's center axis.

Case 3

In this case, the propeller thrust location $\boldsymbol{r}_P$ is assumed to be equal to $(0,0,0)$. Assume the vehicle's inertia matrix has the form $\boldsymbol{I}_B^B = \mathrm{diag}\left(\Theta_x, \Theta_y, \Theta_z\right)$. Then one equilibrium of this special case is $\bar{p}_B = 0$, $\bar{q}_B = 0$, $\bar{r}_B = \sqrt{\kappa \bar{f}_P / K_{d,zz}}$, where $(\bar{p}_B, \bar{q}_B, \bar{r}_B) := \bar{\boldsymbol{\omega}}_{BE}^B$. It can be shown that the linearized reduced state system around this equilibrium is uncontrollable (10.3).

This is also intuitively easy to see, namely, due to the lack of the cross-coupling term in hover and the term $[\![\boldsymbol{r}_P^B \times]\!] \boldsymbol{n}_P^B f_P$ in the x and y-axis, the control input could influence the yaw rate dynamics, but not the roll and pitch rate dynamics. This indicates that when designing a Monospinner, the thrust location should not be too close to the center of mass.

4 Control strategy

The above analysis indicates that by giving up the control of yaw, the reduced state system may be stabilized by a state feedback controller. Recall that the vehicle's position can still be controlled.

Furthermore, the motor dynamics may have a large influence on the system, if the time constant of their response to commands is comparable to the time constants of the remainder of the system. For this reason the motor force is also included as a state, and is approximated by a first order system with time constant τ_{mot}:

$$\dot{f}_P = \tau_{\text{mot}}^{-1}(f_{\text{com}} - f_P) \tag{4.42}$$

where f_{com} is the command thrust for the propeller and f_P is the current propeller thrust.

Augmenting the deviation of the motor force from the equilibrium force (i.e. $f_P - \bar{f}_P$) as a state to the reduced state system, denoting the new state as x, and introducing the new control input $u := f_{\text{com}} - \bar{f}_P$, the augmented state system equation is then

$$\dot{x} \approx A_c x + B_c u \tag{4.43}$$

Note that although the motor force state (or equivalently, the motor speed) represents a degree of freedom of the system, including it in the state space or not does not affect the system's controllability, as the motor force is considered directly as the input to the system in the latter case. From now on, it is always assumed that the system matrix pair (A_c, B_c) is controllable, such that a stabilizing feedback controller may be designed.

An infinite-horizon linear-quadratic regulator (LQR) [85] may be readily designed with with the cost on the position states set to $1\,\mathrm{m}^{-2}\,\mathrm{s}^{-1}$, cost on the roll and pitch states set to $10\,\mathrm{rad}^{-2}\,\mathrm{s}^{-1}$, cost on the input set to $1\,\mathrm{N}^{-2}\,\mathrm{s}^{-1}$, and cost on the rest of the states set to 0, yielding a static feedback gain K:

$$u = -Kx. \tag{4.44}$$

The resulting thrust command is then:

$$f_{\text{com}} = \bar{f}_P + u. \tag{4.45}$$

Note that the controller presented here is different from the one in the conference version [52]: it is a single linear controller that regulates both translational and attitude states, whereas the controller in the conference version employs a cascaded control scheme that exploits time scale separation. This full state control strategy may bring advantages if the desired position dynamics have a similar time constant to the desired attitude dynamics. It also allows for the investigation of the stability margin of the closed-loop system and addressing the issue of actuator saturation, by designing a model predictive controller that takes the input constraint into account while considering the position at the same time.

5 Design

Since the system has only limited control authority at its disposal, it is important to find the vehicle design that is least sensitive to uncertainties such as parametric uncertainties and measurement noise. This section presents the methods to find a vehicle configuration such that the vehicle is sufficiently robust against these uncertainties.

5.1 Simplified mechanical model

To allow for efficient evaluation, a simplified mechanical model is used for the analysis, where there are three major components in the vehicle: the battery, the electronics and the motor (including the propeller). The components' contribution to the composite inertia matrix is approximated as follows: the three major components are approximated as point masses and the connecting frame components are approximated as thin rods. From the inertia matrix (and by assuming that the vehicle has similar drag coefficients as the quadrocopter in [84]), the resulting vehicle's equilibrium solution and the linearized system matrices can be computed as described in the preceding sections.

By measuring the weights of the available components of the prototype, the battery is taken to have a weight of $0.06\,\mathrm{kg}$, the electronics $0.045\,\mathrm{kg}$ and the motor $0.04\,\mathrm{kg}$. The connecting rods are taken to have a length density of $0.06\,\mathrm{kg\,m^{-1}}$.

5.2 Choosing the vehicle configuration

The vehicle design focuses on optimizing over the vehicle's mass distribution. One motivation here is that a mass distribution where the cross-coupling term (i.e. the gyroscopic effect) dominates in hover would make the system's body rate dynamics more coupled and therefore easier to control.

The vehicle's approximate size and shape are based on the existing trispinner [75], with a Y-shape and a vehicle diameter of approximately 30 cm. The positions of the battery and the motor are fixed to be two vertices of an equilateral triangle, while the position of the electronics is to be determined.

A two-dimensional grid search of the position of the electronics is then conducted, where two different quality metrics are considered. The first is the probability of input saturation and is based on the linear, time-invariant model of the dynamic system. The second metric uses Monte Carlo simulations of the nonlinear system, including parameter perturbations and noise, to approximate the probability that the resulting vehicle is able to maintain hover. The probability of input saturation may be computed in closed form for a given design and is therefore cheap to evaluate, but is less informative than the Monte Carlo simulations.

Probability of input saturation

In feedback control, system noise may be amplified into the control input command and cause input saturation even if the system is near equilibrium. It is therefore important to know how measurement and process noise relates to the actual input force, specifically how likely it leads to input saturation. This is particularly true for the Monospinner: with the available motor and propeller, the hover propeller force is near saturation (about 75 percent of the maximum available thrust). In the following, a stochastic analysis is presented: a discretized version of the linear system is derived and augmented with measurement and actuator noise, which is identified by dedicated experiments. The probability that input saturation occurs may then be computed in closed-form.

Discretizing the system (4.43) with a zero-order-hold on the input $u[k]$ leads to:

$$x[k+1] = A_d x[k] + B_d u[k] \tag{4.46}$$

where A_d and B_d are the discretized system matrices.

The measurement outputs are taken to be those available on the experimental platform, that is, every state except the linear velocity. The measurement $z[k]$ is then

$$z[k] = C_d x[k] + w_{\text{meas}}[k] \tag{4.47}$$

where $w_{\text{meas}}[k] \in \mathbb{R}^9$ is the measurement noise, which is assumed to be zero-mean, white, and Gaussian. Furthermore, $C_d \in \mathbb{R}^{9\times 12}$ has the form

$$C_d = \begin{bmatrix} I_3 & 0 & 0 \\ 0 & 0 & I_6 \end{bmatrix} \tag{4.48}$$

where I_3 and I_6 are identity matrices with dimension 3 and 6 and 0 is the zero matrix with associated dimension. Clearly, the system matrix pair (A_d, C_d) is observable.

With $\hat{x}$ defined as the state estimate, a steady-state Kalman filter has the following form:

$$\hat{x}[k] = (I_{12} - K_f C_d)(A_d \hat{x}[k-1] + B_d u[k-1]) + K_f z[k] \tag{4.49}$$

where K_f is the filter gain and I_{12} is the identity matrix with dimension 12.

The controller input follows from applying the discrete LQR gain K_d. It is also assumed that white, Gaussian, and zero-mean actuator noise $w_{\text{act}}[k]$ exist and act on the system. The true control input $u_{\text{true}}[k]$ is then

$$u_{\text{true}}[k] = -K_d \hat{x}[k] + w_{\text{act}}[k]. \tag{4.50}$$

Introducing the extended state $\tilde{x}[k] = (x[k], \hat{x}[k])$ and noise $\tilde{w}[k] = (w_{\text{meas}}[k+1], w_{\text{act}}[k])$, substituting (4.50) into (4.46) yields

$$x[k+1] = A_d x[k] - B_d K_d \hat{x}[k] + B_d w_{\mathrm{act}}[k] \tag{4.51}$$

Substituting (4.51) into (4.47) and then into (4.49) leads to

$$\hat{x}[k] = K_f C_d A_d x[k-1] + \big((I_{12} - K_f C_d) A_d - B_d K_d\big)\, \hat{x}[k-1] + B_d w_{\mathrm{act}}[k-1] + K_f w_{\mathrm{meas}}[k] \tag{4.52}$$

Combining (4.50), (4.51) and (4.52) and introducing the corresponding extended system matrices $\tilde{A}$, $\tilde{B}$, $\tilde{C}$ and $\tilde{D}$, the extended system equations are:

$$\tilde{x}[k+1] = \tilde{A}\tilde{x}[k] + \tilde{B}\tilde{w}[k] \tag{4.53a}$$

$$u_{\mathrm{true}}[k] = \tilde{C}\tilde{x}[k] + \tilde{D}\tilde{w}[k]. \tag{4.53b}$$

By separation theorem for LTI systems and quadratic cost [85], the extended system (4.53a) is stable with a stable feedback controller and a stable state estimator. Thus, the extended system will reach steady state (the equilibrium) as k goes to infinity. Let $P_{\tilde{w}}$, $P_{\tilde{x}}$ and $P_{u_{\mathrm{true}}}$ be the variables' associated steady-state covariance matrices (e.g. $P_{\tilde{x}} = \mathrm{Var}\left(\tilde{x}[k]\right)$ for $k \to \infty$). Through the steady state equations of (4.53a) and (4.53b), the covariance matrices have the following relationship:

$$P_{\tilde{x}} = \tilde{A} P_{\tilde{x}} \tilde{A}^T + \tilde{B} P_{\tilde{w}} \tilde{B}^T \tag{4.54a}$$

$$P_{u_{\mathrm{true}}} = \tilde{C} P_{\tilde{x}} \tilde{C}^T + \tilde{D} P_{\tilde{w}} \tilde{D}^T. \tag{4.54b}$$

Note that (4.54a) is a discrete-time Lyapunov equation, for which a solution $P_{\tilde{x}}$ is guaranteed to exist, since $\tilde{A}$ is discrete-time asymptotically stable, and $\tilde{B} P_{\tilde{w}} \tilde{B}^T$ is positive semi-definite [83]. Furthermore, since the measurement noise variance $P_{\tilde{w}}$ is measured from experiment, and $\tilde{A}$ and $\tilde{B}$ are known, $P_{\tilde{x}}$ can be readily solved by (4.54a). Substituting the solution into (4.54b) gives the variance of the actuator $P_{u_{\mathrm{true}}}$.

Since the noise $\tilde{w}[k]$ is assumed to be Gaussian and zero-mean, $u_{\mathrm{true}}[k]$ is also Gaussian and zero-mean at steady state. As a result, the propeller thrust at equilibrium is a Gaussian random variable with mean $\bar{f}_P$ and variance $P_{u_{\mathrm{true}}}$, from which the probability of saturating the maximal allowed thrust may be calculated. Note that this allows for capturing the fact that a design with low variance may still have a high probability of saturation if it has a high mean thrust. In this way the saturation probabilities of varying positions of the electronics are computed and shown in Fig. 4.6, and the results are discussed in the following.

Monte Carlo analysis:

For each position of the electronics, the nominal hover solution is calculated and an LQR controller is designed using the costs given in the preceding section: this controller is denoted as the "nominal controller". Two hundred perturbed vehicles are then generated, by perturbing the following: inertia matrix $\boldsymbol{I}_B^B$, mass m, and drag coefficients $K_{d,xx}$, $K_{d,yy}$ and $K_{d,zz}$. Each of these parameters is perturbed by sampling within a certain percentage range of the nominal value. For each perturbed vehicle a nonlinear simulation based on the dynamic model given in

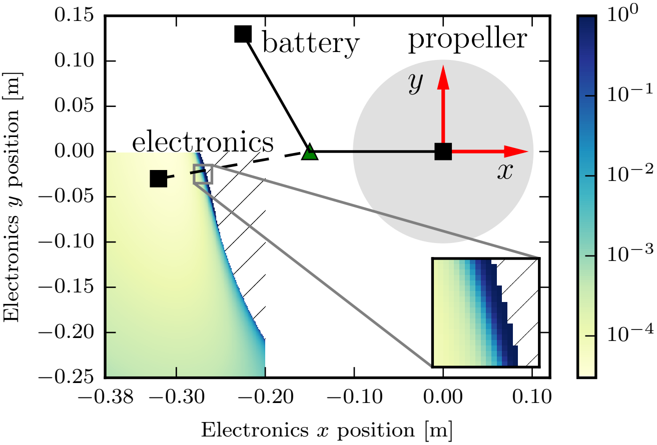


Figure 4.6: The probability of the input saturation for one time step for varying positions of the electronics. In the colored area, a grid search with resolution 0.001 m both in $\boldsymbol{x}$ and $\boldsymbol{y}$-direction is conducted. Electronics positions for which a hover solution cannot be solved are marked with hatching (the upper right corner of the color area). Note that the color bar has logarithmic scale. Note that on the boundary between the regions that has equilibrium solutions and that has no solution, there is a rapid increase of the input saturation probabilities. This is due to the rapid increase in the equilibrium motor force at this boundary. The chosen position of the electronics is also plotted.

Section 2.1 is conducted, lasting 10 simulated seconds. In addition to the perturbed parameters, actuator noise and measurement noise are simulated as in (4.47) and (4.50).

The perturbed vehicle starts at the hover equilibrium of the unperturbed system and is controlled by the nominal controller. If the vehicle has distance greater than 5 m from the reference position at the end of the simulation, it is counted as a failure case. For each candidate position of the electronics, the number of failure cases is plotted in Fig. 4.7. This number is used as an indicator of the robustness of the corresponding nominal configuration.

Discussion

Note that in both Figs. 4.6 and 4.7, there is a good, relatively flat region of electronics positions which have a similar small number of failure cases (respectively a low probability of input saturation). The electronics' position was chosen as $(-0.32, -0.03, 0)\mathrm{m}$ in the coordinate system shown, based on good performance in both metrics, and on a compromise with mechanical strength/complexity and the length of the cables required to connect the components.

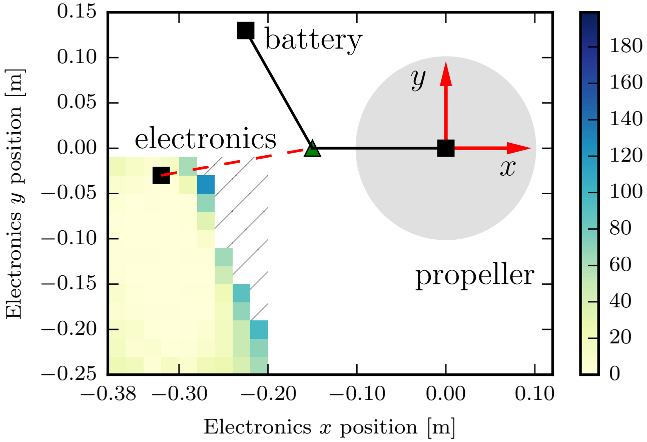


Figure 4.7: The number of failure cases of vehicles under perturbations in nonlinear simulation for varying positions of the electronics. In the colored area, a grid search with resolution 0.02 m both in $\boldsymbol{x}$ and $\boldsymbol{y}$-direction is conducted. Electronics positions for which a hover solution cannot be solved are marked with with hatching (the upper right corner of the color area). The chosen position of the electronics is also plotted.

6 Resulting vehicle

The resulting vehicle, as shown in Fig. 4.1, has a mass of 0.208 kg and the moment of inertia as below (calculated from a CAD-model):

$$\boldsymbol{I}_B^B = \begin{bmatrix} 103 & 15 & 13 \\ 15 & 307 & 4 \\ 13 & 4 & 400 \end{bmatrix} \times 10^{-5}\,\mathrm{kg\,m^2}. \tag{4.55}$$

The linearized system matrices are:

$$A_c = \begin{bmatrix} 0 & 25.65 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -25.65 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -0.004 & 0 \\ -0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0.004 & 0 & 0 \\ 0 & 0 & 0 & 0 & 25.65 & 0 & 0 & 9.81 & 0 & 0 & 0.11 & -1.30 \\ 0 & 0 & 0 & -25.65 & 0 & 0 & -9.81 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.11 & 0 & 0 & 4.63 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 25.65 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -25.65 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -3.41 & -16.64 & 1.48 & -8.95 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 19.89 & 0.64 & 10.94 & -66.19 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.04 & -6.78 & -0.53 & 3.22 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -13.33 \end{bmatrix} \tag{4.56}$$

Table 4.1: Components of the Monospinner

Component	Name
Propeller	GEMFAN GF 8045
Motor	T-Motor MN2204-28 KV:1400
Motor controller	DYS SN20A
Command radio	Laird RM024-S125-M-20
Flight controller	Custom-made flight computer
Battery	G8 Pro Lite 480mAh 3-Cell/3S 11V

$$B_c = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 13.33 \end{bmatrix}^\top . \tag{4.57}$$

Recall that the state x is

$$x = (\delta s_x^C, \delta s_y^C, \delta s_z^C, \delta v_x^C, \delta v_y^C, \delta v_z^C, \delta\phi, \delta\theta, \delta p, \delta q, \delta r, \delta f_P), \tag{4.58}$$

and the input is $u = f_{\text{com}} - \bar{f}_P$.

It can be confirmed that the pair (A_c, B_c) is controllable, and the eigenvalues of the system matrix A_c are: $\{\pm 25.6i, 0, -0.9 \pm 20.0i, -1.6, -13.3\}$.

The expected hover solution for this vehicle is

$$\bar{s}_x^C = 0.0043\,\text{m}, \quad \bar{v}_x^C = 0.11\,\text{m}\,\text{s}^{-1} \tag{4.59}$$

$$\bar{\boldsymbol{\omega}}_{BE}^B = (6.62, -2.04, 24.69)\,\text{rad}\,\text{s}^{-1} \tag{4.60}$$

$$\bar{f}_P = 2.12\,\text{N}\,. \tag{4.61}$$

Note that $\bar{s}_x^C = 0.0043\,\text{m}$ implies that the vehicle's center of mass is rotating in a circle with a radius of 4 millimeters.

Table 4.1 lists the major components of the Monospinner.

7 Experimental results

The experiments are carried out in the Flying Machine Area, an indoor aerial vehicle testbed at ETH Zurich [84]. An infrared motion capture system provides high-quality position and attitude measurements of the vehicle, which are transmitted wirelessly to the Monospinner at 50 Hz. The full state control of the vehicle are run onboard at 1000 Hz. The motor's electronic speed controller directly measures the motor speed, and these measurements are used to estimate the motor force state using (4.5). The attached video shows two types of experiments: take-off from a platform and hand-launching.

7.1 Take-off from a platform

Ideally, one would like the Monospinner to start near the equilibrium, especially in terms of its body rates: if instead the equilibrium thrust is applied when the vehicle has zero angular velocity (e.g. it is at rest on the ground) the vehicle would simply flip over. This is because the cross-coupling term (i.e. the gyroscopic effect) and the airframe drag torque are second-order terms in the angular velocity and thus negligible. Moreover, the propeller's pitch torque is larger than its yaw torque due to the vehicle's geometry: the torque to thrust ratio of the propeller is of the order of 1.5 cm, and the propeller thrust moment arm is 15 cm. Thus, a passive mechanism is designed to allow the Monospinner achieve an angular velocity close to its equilibrium before taking off. The mechanism consists of a platform, on which the Monospinner rests, connected by a bearing to the ground, so that the vehicle can freely rotate about its vector $\boldsymbol{n}_a$. The rotation is achieved solely through the propeller torque τ_P, and the thrust is slowly ramped up from zero to the equilibrium solution. Once sufficiently close to equilibrium, the full control is switched on and the vehicle takes off. A representative state history during a take-off is shown in Fig. 5.3.

The equilibrium body rates of the vehicle in hover are as below, which may be compared to the expected values in (4.60) and (4.61)

$$\bar{\boldsymbol{\omega}}_{BE} = (6.9, -1.2, 24.8)\,\mathrm{rad\,s}^{-1} \tag{4.62}$$

$$\bar{f}_P = 2.12\,\mathrm{N}\,. \tag{4.63}$$

7.2 Hand launch

Alternatively, the Monospinner can be launched by throwing it like a frisbee. This is a faster method of achieving hover than the takeoff mechanism in Section 7.1, and shows that the resulting system's equilibrium has a large region of attraction. A representative state history during a hand-launch is shown in Fig. 4.9.

8 Conclusion

This paper presents the modeling, design, and control of a flying vehicle with only one moving part and a single control input, which is able to fully control its position and may be used as novel hobbyist platforms, toys, or low-cost flying vehicles. First, the vehicle's coupled translational and attitude dynamics are formulated as a twelve state system for which an equilibrium exists. This allows for analysis of the linearized system using the powerful tools from linear system theory. Then a controllability analysis is given: It is shown that the full state system is never stabilizable, and after removing the yaw state, the reduced state system maybe fully controllable in position. In particular, the reduced state system is always stabilizable for a class of vehicles that has the shape of a planar object and an offset thrust location with respect to the center of mass. The resulting vehicle may be approximated by an instance of this class of vehicles and its corresponding system matrix pair is shown to be indeed stabilizable. An LQR controller for the reduced state system is designed and is shown to work reliably in the experiments. A vehicle design method is also presented: it optimizes mainly over the vehicle's shape

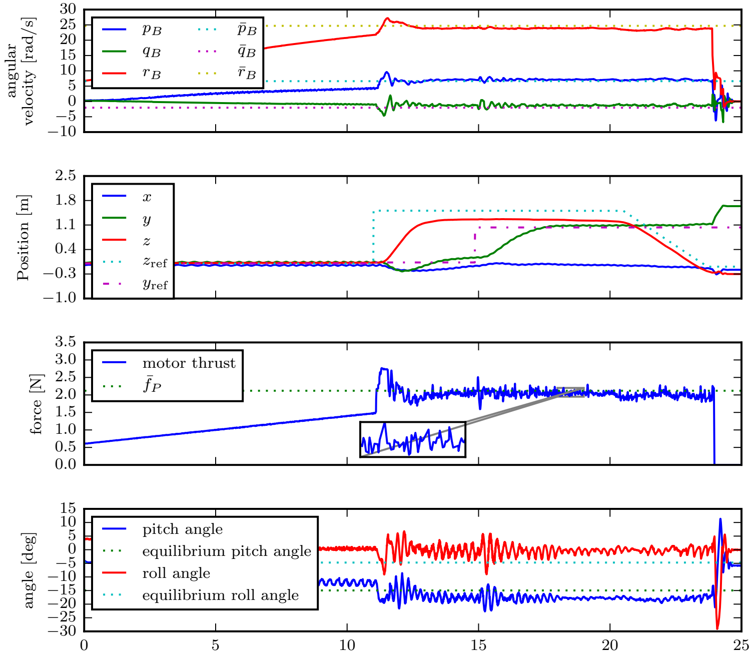


Figure 4.8: Experimental results for the Monospinner's take-off from the platform. The vehicle takes off at 11 s and lands at 20 s. At time 15 s, a reference position change of 1 m is set in the (horizontal) $\boldsymbol{y}$-direction. Note that at steady-state there is an offset between the vehicle's height z and the reference height z_{ref}. This is due to the discrepancy between the expected hover solution and the true hover solution. and it may be readily compensated by adding an integral term to the position control. The angular velocity is plotted as expressed in the body-fixed coordinate system, where $\boldsymbol{\omega}_{BE}^{B} = (p_B, q_B, r_B)$. The roll and pitch angles are the standard Euler sequence (1,2,3) angles from the E-frame to the B-frame. The attached video shows such an experiment.

and hence its mass distribution, in order to find a design that is robust against system noise and parametric uncertainties. Finally, the resulting vehicle is shown to be capable of hovering and its equilibrium has a large region of attraction such that the vehicle recovers to hover after being thrown into the air like a frisbee. An area of additional investigation may be the analysis of the presented linear controller and the determination of the region of attraction of the resulting equilibrium.

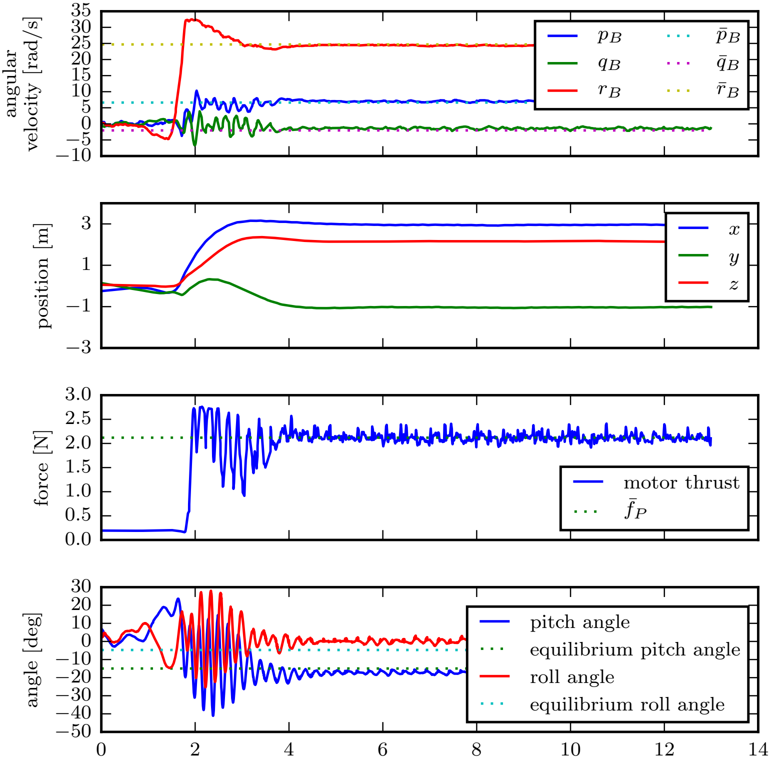


Figure 4.9: Experimental results for a successful hand launch of the Monospinner. Its initial angular velocity has about 30% deviation of the equilibrium angular velocity, and its initial roll and pitch both have about 20 degrees deviation of the equilibrium roll and pitch. The vehicle is thrown at approximately 2 s, after which the controllers are switched on. The angular velocity is plotted as expressed in the body-fixed coordinate system, where $\boldsymbol{\omega}_{BE}^{B} = (p_B, q_B, r_B)$. The roll and pitch angles are the standard Euler sequence (1,2,3) angles from the E-frame to the B-frame. The attached video shows such an experiment.

Acknowledgments

This research was partially supported by the SNF Grant 162365. The authors would like to thank Michael Egli, Marc-Andrè Corzillius and Yilun Wu for their contribution to the mechanical design, electrical design and the onboard firmware code, respectively. The authors would also like to thank Michael Muehlebach for critically reading a draft of this manuscript and for valuable discussions. The Flying Machine Arena is the result of contributions of many people, a full list of which can be found at `http://flyingmachinearena.org/`.

9 Equivalent controllability tests for the reduced state system

In this appendix it will be shown that the matrices $[-A_{11}\ B_1]$, $[\pm\bar{\omega}iI - A_{11}\ B_1]$ and $[\lambda I - A_{11}\ B_1]$ with $\lambda \in \mathrm{spec}(A_S^C)$ have full rank if and only if the matrices U_0 (4.37), U_i (4.38), and $U_s(\lambda)$ (4.39) have full rank, respectively.

According to [86], the system matrix pair (A_{11}, B_1) is uncontrollable if and only if there exists a $v \neq 0$ with

$$v^\top A_{11} = \lambda v^\top, \quad v^\top B_1 = 0, \tag{4.64}$$

where λ and its associated left eigenvector v is an uncontrollable mode. Therefore, to determine whether the test matrix $[\lambda I - A_{11}\ B_1]$ has full rank is equivalent to solving for a non-zero solution v in the equation $v^\top[\lambda I - A_{11}\ B_1] = 0$ (e.g. if there exists a non-zero v, then the test matrix does not have full rank, and vice versa). In the following, the equation will be solved for each eigenvalue of A_{11}, which are 0, $\pm\bar{\omega}i$, and the eigenvalues of the submatrix A_S^C.

Eigenvalue $\lambda = 0$

Taking the transpose of the matrices on both sides of the equation yields

$$[-A_{11}\ B_1]^\top v = 0. \tag{4.65}$$

Denote $v \in \mathbb{R}^{11}$ by $v = [v_1, v_2, v_3, v_4]$ with $v_1, v_2, v_4 \in \mathbb{R}^3$ and $v_3 = (v_{31}, v_{32}) \in \mathbb{R}^2$. In total, there are 12 equations.

Solving the first three equations of (4.65),

$$-[\![\bar{\boldsymbol{\omega}}_{CE}^C \times]\!] v_1 = 0, \tag{4.66}$$

leads to $v_1 = \alpha\bar{\boldsymbol{\omega}}_{CE}^C$, where $\alpha \in \mathbb{R}$.

The next three equations are

$$-v_1 - [\![\bar{\boldsymbol{\omega}}_{CE}^C \times]\!] v_2 = 0. \tag{4.67}$$

Substituting $v_1 = \alpha\bar{\boldsymbol{\omega}}_{CE}^C$ into (4.67) yields $\alpha = 0$ and thus $v_1 = 0$, and $v_2 = \beta\bar{\boldsymbol{\omega}}_{CE}^C$, with $\beta \in \mathbb{R}$.

From the 7th and the 8th equations it follows that

$$\begin{bmatrix} 0 & \bar{\omega} \\ -\bar{\omega} & 0 \end{bmatrix} \begin{bmatrix} v_{31} \\ v_{32} \end{bmatrix} = 0, \tag{4.68}$$

yielding $v_3 = 0$.

The last four equations are

$$[\![\bar{\boldsymbol{v}}^C \times]\!] v_2 - (A_S^C)^\top v_4 = 0 \tag{4.69}$$

and

$$m^{-1}(\boldsymbol{n}_P^C)^\top v_2 + (B_S^C)^\top v_4 = 0. \tag{4.70}$$

Its solution depends on the entries of A_S^C and B_S^C, which are functions of the vehicle's physical parameters.

In summary, the existence of the solution of (4.65) is equivalent to the existence of the solution of the following equation:

$$\underbrace{\begin{bmatrix} V_0 & -(A_S^C)^\top \\ m^{-1} n_{P,z}^C & (B_S^C)^\top \end{bmatrix}}_{=:U_0} \begin{bmatrix} v_{23} \\ v_4 \end{bmatrix} = 0 \tag{4.71}$$

with $V_0 = (\bar{v}_y^C, 0, 0)$ and v_{23} denoting the third component of v_2. Thus there exists a non-zero solution for (4.65) if and only if the matrix U_0 does not have full rank.

Eigenvalue $\lambda = \pm\bar{\omega} i$

As pointed out in Section 3.2, only the case of $\lambda = \bar{\omega} i$ needs to be investigated. The equation to be solved is

$$[i\bar{\omega} I - A_{11} \;\; B_1]^\top v = 0. \tag{4.72}$$

Solving the first three equations

$$\left(i\bar{\omega} I - [\![\bar{\boldsymbol{\omega}}_{CE}^C \times]\!]\right) v_1 = 0. \tag{4.73}$$

This leads to $v_1 = (\alpha, -i\alpha, 0)$, with $\alpha \in \mathbb{R}$.

The next three equations are

$$\left(i\bar{\omega} I - [\![\bar{\boldsymbol{\omega}}_{CE}^C \times]\!]\right) v_2 - v_1 = 0. \tag{4.74}$$

It follows that $\alpha = 0$ and thus $v_1 = 0$, and $v_2 = (\beta, -i\beta, 0)$, with $\beta \in \mathbb{R}$.

From the 7th to the 8th equations

$$\begin{bmatrix} 0 & \|\boldsymbol{g}\| \\ -\|\boldsymbol{g}\| & 0 \end{bmatrix} \begin{bmatrix} \beta \\ -i\beta \end{bmatrix} + \begin{bmatrix} i & 1 \\ -1 & i \end{bmatrix} \bar{\omega} v_3 = 0. \tag{4.75}$$

The result follows as $\beta = 0$, which leads to $v_2 = 0$, and $v_3 = (\gamma, -i\gamma)$.

The last four equations are

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \gamma \\ -i\gamma \end{bmatrix} + \left(i\bar{\omega}I_3 - A_S^C\right)^\top v_4 = 0 \tag{4.76}$$

$$(B_S^C)^\top v_4 = 0, \tag{4.77}$$

the solution of which depends on the parameters of A_S^C and B_S^C.

In summary, the existence of a non-zero solution for (4.72) is equivalent to the existence of a non-zero solution for the following equation

$$\underbrace{\begin{bmatrix} V_i & \bar{\omega}iI - (A_S^C)^\top \\ 0 & (B_S^C)^\top \end{bmatrix}}_{U_i} \begin{bmatrix} \gamma \\ v_4 \end{bmatrix} = 0 \tag{4.78}$$

where $V_i = (1, -i, 0)$. This is the case if and only if the matrix U_i does not have full rank.

Eigenvalues of A_S^C

Recall that it is assumed that the eigenvalues of A_S^C are distinct from 0 and $\pm\bar{\omega}i$ (otherwise we can check the rank of U_0 or U_i). Therefore, the upper left 9 by 9 block matrix of $[\lambda I - A_{11} \ B_1]$ has full rank, and it suffices to investigate the rank of its lower right 3 by 4 block matrix $[\lambda I - A_S^C \ B_S^C]$ (Fact 2.11.13 [83]).

10 Controllability analysis for three special cases of the reduced state system

In Section 3.2, controllability analysis is performed for three special cases of reduced state system under simplifying assumptions. In this appendix, details of derivation are shown for each case.

10.1 Controllability analysis for case 1

In this case (for assumptions see Section 3.2), we will show that the system is at least stabilizable. Let $\boldsymbol{\omega}_{BE}^B = (p_B, q_B, r_B)$. Writing out the simplified Euler equation (4.40) under the proposed assumptions for case 1 yields

$$\dot{p}_B = -q_B r_B \tag{4.79}$$

$$\dot{q}_B = p_B r_B - \frac{l}{\Theta_y} f_P \tag{4.80}$$

$$\dot{r}_B = \frac{\Theta_x - \Theta_y}{\Theta_x + \Theta_y} p_B q_B - \frac{K}{\Theta_x + \Theta_y} r_B^2 + \frac{\kappa}{\Theta_x + \Theta_y} f_P. \tag{4.81}$$

Setting the right hand side of the above three equations to zero yields three nonlinear equations, from which the equilibrium body rates $(\bar{p}_B, \bar{q}_B, \bar{r}_B)$ may be solved:

$$0 = \bar{q}_B \bar{r}_B \tag{4.82}$$

$$0 = \bar{p}_B \bar{r}_B - \frac{l}{\Theta_y} \bar{f}_P \tag{4.83}$$

$$0 = \frac{\Theta_x - \Theta_y}{\Theta_x + \Theta_y} \bar{p}_B \bar{q}_B - \frac{K}{\Theta_x + \Theta_y} \bar{r}_B^2 + \frac{\kappa}{\Theta_x + \Theta_y} \bar{f}_P . \tag{4.84}$$

Solving the above equations yields:

$$\bar{p}_B = \frac{l}{\Theta_y} \sqrt{\frac{\bar{f}_P}{\kappa}}, \quad \bar{q}_B = 0, \quad \bar{r}_B = \sqrt{\kappa \bar{f}_P}\,. \tag{4.85}$$

Linearizing (4.79), (4.80) and (4.81) around $(\bar{p}_B, \bar{q}_B, \bar{r}_B)$ and $\bar{f}_P$ yields

$$A_S^B = \begin{bmatrix} 0 & -\bar{r}_B & 0 \\ \bar{r}_B & 0 & \bar{p}_B \\ 0 & D\bar{p}_B & -2k\bar{r}_B \end{bmatrix}, \quad B_S^B = \begin{bmatrix} 0 \\ -\frac{l}{\Theta_y} \\ \frac{\kappa}{\Theta_x + \Theta_y} \end{bmatrix}. \tag{4.86}$$

where

$$D := \frac{\Theta_x - \Theta_y}{\Theta_x + \Theta_y}, \quad k := \frac{K}{\Theta_x + \Theta_y}. \tag{4.87}$$

From (4.83) and (4.84), B_S^B can be written as

$$B_S^B = (0, -\frac{\bar{p}_B \bar{r}_B}{\bar{f}_P}, \frac{k \bar{r}_B^2}{\bar{f}_P}). \tag{4.88}$$

Let $\boldsymbol{R}^{BC}$ be parametrized by the standard aeronautics Euler angle sequence with roll (ν), pitch (μ), and yaw (η) angles such that

$$\boldsymbol{R}^{BC} = \boldsymbol{R}_x(\nu) \boldsymbol{R}_y(\mu) \boldsymbol{R}_z(\eta). \tag{4.89}$$

Combining (4.9), (4.13) and (4.89) yields

$$\frac{\bar{p}_B}{\bar{\omega}} = -\sin\mu, \quad \frac{\bar{q}_B}{\bar{\omega}} = \cos\mu \sin\nu, \quad \frac{\bar{r}_B}{\bar{\omega}} = \cos\mu \cos\nu. \tag{4.90}$$

Since $\bar{q}_B = 0$ and $\bar{r}_B \neq 0$, it can be seen from (4.90) that $\sin\nu$ is equal to 0, which leads to $\nu = 0$.

With the second row of (4.14) the remaining degree of freedom η can be solved:

$$\cos(\nu)\sin(\mu)\sin(\eta) - \sin(\nu)\cos(\eta) = 0 \tag{4.91}$$

which yields

$$\eta = \arctan\left(\frac{\tan(\nu)}{\sin(\mu)}\right) = 0. \tag{4.92}$$

Therefore, the coordinate transformation from the C-frame to the B-frame is a rotation around the y-axis of the C-frame, that is,

$$\boldsymbol{R}^{BC} = \begin{bmatrix} \cos\mu & 0 & -\sin\mu \\ 0 & 1 & 0 \\ \sin\mu & 0 & \cos\mu \end{bmatrix} \tag{4.93}$$

and $\boldsymbol{R}^{BC}\bar{\boldsymbol{\omega}}^{C}_{CE} = \bar{\boldsymbol{\omega}}^{B}_{BE}$ leads to

$$\bar{p}_B = -\sin(\mu)\bar{\omega}, \quad \bar{r}_B = \cos(\mu)\bar{\omega}. \tag{4.94}$$

For brevity, let $\alpha = -\sin(\mu) > 0$ (since $\bar{p}_B > 0$) and $\beta = \cos(\mu) > 0$. Note that $\alpha^2 + \beta^2 = 1$.

Substituting (4.94) into A^B_S and B^B_S and applying coordinate transformation $A^C_S = \boldsymbol{R}^{CB} A^B_S \boldsymbol{R}^{BC}$ and $B^C_S = \boldsymbol{R}^{CB} B^B_S$ yields

$$A^C_S = \begin{bmatrix} -2k\beta\alpha^2\bar{\omega} & (-\beta^2 - D\alpha^2)\bar{\omega} & 2k\beta^2\alpha\bar{\omega} \\ (\beta^2 - \alpha^2)\bar{\omega} & 0 & 2\beta\alpha\bar{\omega} \\ 2k\beta^2\alpha\bar{\omega} & (-\beta\alpha + D\beta\alpha)\bar{\omega} & -2k\beta^3\bar{\omega} \end{bmatrix} \tag{4.95}$$

and

$$B^C_S = \begin{bmatrix} -\frac{k\beta^2\alpha\bar{\omega}^2}{\bar{f}_P} & -\frac{\beta\alpha\bar{\omega}^2}{\bar{f}_P} & \frac{k\beta^3\bar{\omega}^2}{\bar{f}_P} \end{bmatrix}, \tag{4.96}$$

respectively.

Substituting $n^C_{P,z} = \beta$, (4.25), (4.95), and (4.96) into U_0 (4.71) and computing its determinant yields

$$\det(U_0) = -\frac{2k\beta^4\bar{\omega}^3}{m}(\beta^2 + \alpha^2)^2, \tag{4.97}$$

which is non-zero, meaning that $[-A_{11} \;\; B_1]$ has full rank.

For the eigenvalues $\pm\bar{\omega}i$, (4.78) becomes

$$U_i = \begin{bmatrix} 1 & \bar{\omega}i + 2k\beta\alpha^2\bar{\omega} & -(\beta^2 - \alpha^2)\bar{\omega} & -2k\beta^2\alpha\bar{\omega} \\ -i & (\beta^2 + D\alpha^2)\bar{\omega} & \bar{\omega}i & \beta\alpha\bar{\omega} - D\beta\alpha\bar{\omega} \\ 0 & -2k\beta^2\alpha\bar{\omega} & -2\beta\alpha\bar{\omega} & \bar{\omega}i + 2k\beta^3\bar{\omega} \\ 0 & -\frac{k\beta^2\alpha\bar{\omega}^2}{\bar{f}_P} & -\frac{\beta\alpha\bar{\omega}^2}{\bar{f}_P} & \frac{k\beta^3\bar{\omega}^2}{\bar{f}_P} \end{bmatrix}. \tag{4.98}$$

To compute its determinant, multiply its fourth row by $-2\bar{f}_P/\bar{\omega}$ and add to the third row and then compute its determinant yields

$$\det(U_i) = -i\frac{\beta\alpha\bar{\omega}^4}{\bar{f}_P}(-(\beta^2 + D\alpha^2 - 1)). \tag{4.99}$$

Assume $\det(U_i) = 0$, then the following equation has to hold

$$\beta^2 + D\alpha^2 = 1, \tag{4.100}$$

simplifying which yields

$$\Theta_x - \Theta_y = \Theta_x + \Theta_y, \tag{4.101}$$

which is clearly a contradiction ($\Theta_y \neq 0$). Thus, $[\bar{\omega} i I - A_{11} \;\; B_1]$ has full rank.

For the eigenvalues of A_S^C, the matrix $[\lambda I - A_S^C \;\; B_S^C]$ has full rank for all λ is equivalent to the controllability of the matrix pair (A_S^C, B_S^C) (the PBH test), which is then equivalent to the full rankness of its associated controllability matrix

$$\mathcal{C} = \begin{bmatrix} B_S^B & A_S^B B_S^B & (A_S^B)^2 B_S^B \end{bmatrix}. \tag{4.102}$$

Note that the matrix pair (A_S^B, B_S^B) with substitution from (4.94) is used instead, since coordinate transformation (which is the same as change of basis) does not affect the controllability of the linear system matrix pair, and it is easier to evaluate the controllability matrix $\mathcal{C}$ using the pair (A_S^B, B_S^B).

Substituting (4.86) into $\mathcal{C}$ leads to

$$\mathcal{C} = \frac{\beta\bar{\omega}^2}{\bar{f}_P} \begin{bmatrix} 0 & \beta\alpha\bar{\omega} & -k\beta^2\alpha\bar{\omega}^2 \\ -\alpha & k\beta\alpha\bar{\omega} & \alpha\bar{\omega}^2(\beta^2 - D\alpha^2 - 2k^2\beta^2) \\ k\beta & -(D\alpha^2 + 2k^2\beta^2)\bar{\omega} & k\beta\bar{\omega}^2(3Dr_2^2 + 4k^2\beta^2) \end{bmatrix}. \tag{4.103}$$

To compute its determinant, multiply the first and second column by $k\beta\bar{\omega}$ and add it to the second and third column, respectively, which yields

$$\mathcal{C} = \frac{\beta\bar{\omega}^2}{\bar{f}_P} \begin{bmatrix} 0 & \beta\alpha\bar{\omega} & 0 \\ -\alpha & 0 & \alpha\bar{\omega}^2(\beta^2 - D\alpha^2 - k^2\beta^2) \\ k\beta & -(D\alpha^2 + k^2\beta^2)\bar{\omega} & k\beta\bar{\omega}^2(2Dr_2^2 + 2k^2\beta^2) \end{bmatrix}. \tag{4.104}$$

Again, multiply the second column by $2k\beta\bar{\omega}$ and add it to the third column

$$\mathcal{C} = \frac{\beta\bar{\omega}^2}{\bar{f}_P} \begin{bmatrix} 0 & \beta\alpha\bar{\omega} & 2k\beta^2\alpha\bar{\omega}^2 \\ -\alpha & 0 & \alpha\bar{\omega}^2(\beta^2 - D\alpha^2 - k^2\beta^2) \\ k\beta & -(D\alpha^2 + k^2\beta^2)\bar{\omega} & 0 \end{bmatrix}. \tag{4.105}$$

The determinant is then computed as

$$\det(\mathcal{C}) = \frac{k\beta^5\alpha^2\bar{\omega}^9}{\bar{f}_P^3}(D\alpha^2 + \beta^2 + k^2\beta^2). \tag{4.106}$$

Assume $\det(\mathcal{C}) = 0$, by exploiting $\alpha^2 = 1 - \beta^2$,

$$D + \beta^2(k^2 - D + 1) = 0. \tag{4.107}$$

Substituting the definition of D and k (4.87) back into the above equation yields

$$\beta^2 = \frac{\Theta_x^2 - \Theta_y^2}{-2\Theta_y^2 - 2\Theta_x\Theta_y - K^2}. \tag{4.108}$$

If $\Theta_x^2 - \Theta_y^2 \geq 0$, clearly, the left hand side of (4.108) cannot be equal to its right hand side. Thus, the matrix $\mathcal{C}$ has full rank.

If $\Theta_x^2 - \Theta_y^2 < 0$, the eigenvalues of A_S^B are guaranteed to be stable. To see this, computing the characteristic polynomial of the matrix A_S^B (eigenvalues of a matrix stay invariant under coordinate transformation) leads to

$$\det(\lambda I - A) = \lambda^3 + \underbrace{2k\beta\bar{\omega}}_{a_1}\lambda^2 + \underbrace{(\beta^2\bar{\omega}^2 - \alpha^2\bar{\omega}^2 D)}_{a_2}\lambda + \underbrace{2k\beta^3}_{a_3}\bar{\omega}^3 = 0. \tag{4.109}$$

According to the Routh-Hurwitz stability criterion, the poles of (4.109) have strictly negative parts if and only if the conditions $a_1 > 0$, $a_2 > 0$, $a_1 a_2 > a_3 > 0$ are fulfilled (Fact 11.17.2 [83]). This is clearly the case if $\Theta_x^2 - \Theta_y^2 < 0$ (i.e. $D < 0$) and recall that $k > 0$, $\beta > 0$, and $\bar{\omega} > 0$.

In conclusion, the system matrix pair (A_{11}, B_1) is at least stabilizable for this case.

10.2 Controllability analysis for case 2

In this case (for assumptions see Section 3.2), we will show that the system is not stabilizable.

The Euler equation simplifies to

$$\dot{p}_B = -\frac{J}{\Phi}p_B \left\|\boldsymbol{\omega}_{BE}^B\right\| \tag{4.110}$$

$$\dot{q}_B = -\frac{l}{\Theta}f_P - \frac{K}{\Theta}q_B \left\|\boldsymbol{\omega}_{BE}^B\right\| + \frac{\Theta - \Phi}{\Theta}p_B r_B \tag{4.111}$$

$$\dot{r}_B = \frac{\kappa}{\Theta}f_P - \frac{K}{\Theta}r_B \left\|\boldsymbol{\omega}_{BE}^B\right\| + \frac{\Phi - \Theta}{\Theta}p_B q_B. \tag{4.112}$$

Setting the left hand side of (4.110) to zero yields $\bar{p}_B = 0$.

Let the components of $\boldsymbol{R}^{BC}$ be

$$\boldsymbol{R}^{BC} = \begin{bmatrix}\boldsymbol{e}_1 & \boldsymbol{e}_2 & \boldsymbol{e}_3\end{bmatrix} = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix}, \tag{4.113}$$

where $\boldsymbol{e}_i, i = 1, 2, 3$ denote the column vectors of $\boldsymbol{R}^{BC}$, and $r_i, i = 1, ...9$ denote the entries. Since $\boldsymbol{R}^{BC}$ is a coordinate transformation matrix, the column vectors satisfy the following properties:

$$\boldsymbol{e}_1 \times \boldsymbol{e}_2 = \boldsymbol{e}_3 \tag{4.114}$$

$$\boldsymbol{e}_2 \times \boldsymbol{e}_3 = \boldsymbol{e}_1 \tag{4.115}$$

$$\boldsymbol{e}_3 \times \boldsymbol{e}_1 = \boldsymbol{e}_2. \tag{4.116}$$

$\boldsymbol{R}^{BC}\bar{\boldsymbol{\omega}}^{C}_{CE} = \bar{\boldsymbol{\omega}}^{B}_{BE}$ can be written as

$$\bar{p}_B = \bar{\omega} r_3 = 0, \quad \bar{q}_B = \bar{\omega} r_6, \quad \bar{r}_B = \bar{\omega} r_9, \tag{4.117}$$

which also leads to $r_3 = 0$.

Furthermore, by (4.14)

$$0 = n^{C}_{P,y} = \left(\boldsymbol{R}^{CB} n^{B}_{P}\right)_2 = r_8, \tag{4.118}$$

where $\left(\boldsymbol{R}^{CB} n^{B}_{P}\right)_2$ denotes the second entry of $\boldsymbol{R}^{CB} n^{B}_{P}$.

Linearizing (4.110)-(4.112) around $(\bar{p}_B, \bar{q}_B, \bar{r}_B)$ yields

$$A^{B}_{S} = \begin{bmatrix} -j\bar{\omega} & 0 & 0 \\ -c\bar{r}_B & -k\bar{\omega} & 0 \\ c\bar{q}_B & 0 & -k\bar{\omega} \end{bmatrix} - \frac{k}{\bar{\omega}} \bar{\boldsymbol{\omega}}^{B}_{BE} (\bar{\boldsymbol{\omega}}^{B}_{BE})^{\top}, \quad B^{B}_{S} = \begin{bmatrix} 0 & -\frac{l}{\Theta} & \frac{\kappa}{\Theta} \end{bmatrix}, \tag{4.119}$$

where $j := \frac{J}{\Phi}$, $k := \frac{K}{\Theta}$ and $c := \frac{\Phi - \Theta}{\Theta}$.

Substituting (4.117) into A^{B}_{S} and applying coordinate transformation $A^{C}_{S} = \boldsymbol{R}^{CB} A^{B}_{S} \boldsymbol{R}^{BC}$ and some simplifications ((4.114), (4.115), (4.116), and (4.118)), it follows that

$$A^{C}_{S} = \begin{bmatrix} -k\bar{\omega} + c r_1 r_2 \bar{\omega} + r_1^2 (k - j)\bar{\omega} & r_2^2 c\bar{\omega} + r_1 r_2 (k - j)\bar{\omega} & r_2 r_3 c\bar{\omega} \\ -r_1^2 c\bar{\omega} + r_1 r_2 (k - j)\bar{\omega} & -k\bar{\omega} - c r_1 r_2 \bar{\omega} + r_2^2 (k - j)\bar{\omega} & r_1 r_3 c\bar{\omega} \\ 0 & 0 & -2k\bar{\omega} \end{bmatrix}. \tag{4.120}$$

Substituting (4.117) into (4.111) and (4.112) and setting their left hand side to zero yields

$$-\frac{l}{\Theta} = \frac{1}{\bar{f}_P} k r_6 \bar{\omega}^2 \tag{4.121}$$

$$\frac{\kappa}{\Theta} = \frac{1}{\bar{f}_P} k r_9 \bar{\omega}^2. \tag{4.122}$$

Substituting (4.121) and (4.122) into B^{B}_{S} in (4.119) and simplifying $\boldsymbol{R}^{CB} B^{B}_{S}$ yields

$$B^{C}_{S} = \begin{bmatrix} 0 & 0 & \frac{k\bar{\omega}^2}{\bar{f}_P} \end{bmatrix}. \tag{4.123}$$

Substituting the (4.120) and (4.123) into the definition of U_i and computing its determinant leads to

$$\det(U_i) = 0. \tag{4.124}$$

This implies that the modes associated with the eigenvalues $\pm\bar{\omega}i$ are not controllable, and the system is therefore not stabilizable.

10.3 Controllability analysis for case 3

The simplified Euler equation (4.40) for this case (for assumptions see Section 3.2) has the form

$$\Theta_x \dot{p}_B = (\Theta_y - \Theta_z) q_B r_B - K_{d,xx} p_B \left\| \boldsymbol{\omega}_{BE}^B \right\| \tag{4.125}$$

$$\Theta_y \dot{q}_B = (\Theta_z - \Theta_x) p_B r_B - K_{d,yy} q_B \left\| \boldsymbol{\omega}_{BE}^B \right\| \tag{4.126}$$

$$\Theta_z \dot{r}_B = (\Theta_x - \Theta_y) p_B q_B - K_{d,zz} r_B \left\| \boldsymbol{\omega}_{BE}^B \right\| + \kappa f_P. \tag{4.127}$$

Linearizing the above three equations around the equilibrium $(0, 0, \sqrt{\frac{\kappa \bar{f}_P}{K_{d,zz}}})$ yields

$$A_S^B = \begin{bmatrix} -k_x \bar{\omega} & a \bar{r}_B & 0 \\ b \bar{r}_B & -k_y \bar{\omega} & 0 \\ 0 & 0 & -k_z \bar{\omega} \end{bmatrix}, \quad B_S^B = \begin{bmatrix} 0 \\ 0 \\ \frac{\kappa}{\Theta_z} \end{bmatrix} \tag{4.128}$$

where $k_x := K_{d,xx}/\Theta_x$, $k_y := K_{d,yy}/\Theta_y$, $k_z := K_{d,zz}/\Theta_z$, $a := (\Theta_y - \Theta_z)/\Theta_x$, and $b := (\Theta_z - \Theta_x)/(\Theta_y)$.

From (4.90) and (4.91) it can be solved that $\mu = \nu = \eta = 0$. Therefore, $\boldsymbol{R}^{BC}$ is a three dimensional identity matrix.

For the eigenvalue $\pm \bar{\omega} i$, it is clear that $\det(U_i) = 0$. Thus the system for this case is not stabilizable.

Part B

MODEL LEARNING FOR CONTROL OF OMNIDIRECTIONAL FLYING VEHICLES

Paper II

Learning dynamics for improving control of overactuated flying systems

Weixuan Zhang, Maximilian Brunner, Lionel Ott, Mina Kamel, Roland Siegwart, and Juan Nieto

Abstract

Overactuated omnidirectional flying vehicles are capable of generating force and torque in any direction, which is important for applications such as contact-based industrial inspection. This comes at the price of an increase in model complexity. These vehicles usually have non-negligible, repetitive dynamics that are hard to model, such as the aerodynamic interference between the propellers. This makes it difficult for high-performance trajectory tracking using a model-based controller. This paper presents an approach that combines a data-driven and a first-principle model for the system actuation and uses it to improve the controller. In a first step, the first-principle model errors are learned offline using a Gaussian Process (GP) regressor. At run-time, the first-principle model and the GP regressor are used jointly to obtain control commands. This is formulated as an optimization problem, which avoids ambiguous solutions present in a standard inverse model in overactuated systems, by only using forward models. The approach is validated using a tilt-arm overactuated omnidirectional flying vehicle performing attitude trajectory tracking. The results show that with our proposed method, the attitude trajectory error is reduced by 32% on average as compared to a nominal PID controller.

Published in:
IEEE Robotics and Automation Letters, 2020, Volume 5, Issue 4, 2020
© Open access article distributed under the Creative Commons Attribution License.
DOI: 10.1109/LRA.2020.3007451

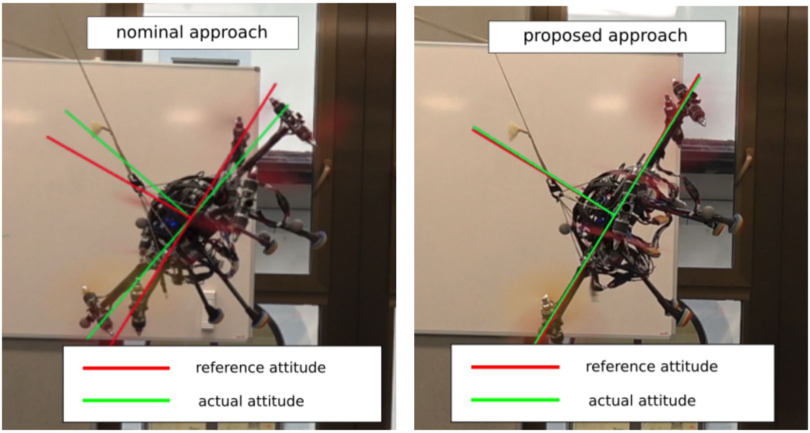


Figure 5.1: Attitude tracking performance of an overactuated flying vehicle: (left) without model learning, and (right) using our proposed learning based approach. For perfect tracking the green frame should coincide with the red frame.

1 Introduction

Omnidirectional flying vehicles ([24], [87]–[91]) are suitable for industrial inspection or interaction applications due to their decoupled translational and rotational dynamics (e.g. being able to hover at arbitrary attitudes) and the ability to exert forces and torques in arbitrary directions.

These vehicles are overactuated, resulting in a mechanical redundancy that provides the system with the necessary control authority to achieve omnidirectionality and increases their maneuverability and force/torque margin. However, the increased mechanical complexity also poses some challenges. Unlike standard quadrotors or hexacopters, for which a simple and accurate actuation model is available for low speed maneuvers [7], only an inaccurate and simplified actuation model can be obtained for omnidirectional vehicles from first principles. The actuation model is the mapping from the individual actuator commands to the total wrench, i.e., force and torque. To achieve omnidirectionality, vehicle designs often cause complex aerodynamic effects that are hard-to-model and non-negligible, such as the aerodynamic interference between the rotors. This is disadvantageous from a control perspective. A standard approach for overactuated flying vehicles is that a controller first computes a desired wrench, which, based on the inverse of the actuation model, is transformed into individual actuator commands. Due to unmodeled effects, this typically results in a different actual wrench than the intended one. This leads to degraded control performance and reduces the ability to accurately track desired trajectories.

To solve the problem mentioned above, adding only integral actions in the feedback controller is often not enough, as they typically decrease the damping or stability of the system and are not able to react to fast-changing modeling errors. Another solution is to treat the unmodelled dynamics as disturbances [25] [26] and employ a disturbance observer. While being computationally efficient, this reactive strategy introduces delay in tracking. One alternative solution is to identify these mismatched dynamics and compute additional feedforward signals in an attempt to cancel out the effect of modeling errors. One way to model this mismatch is using data-driven machine learning models. Such models have recently been shown to be viable due to their ability to learn complex and nonlinear dynamic functions [27]. Directly learning an inverse mapping of the actuation, that is, a mapping from the wrench to the individual actuator commands, is a commonly used method. However, this approach is problematic for an overactuated system due to its one-to-many mapping between wrench and actuator inputs. The multi-modal nature of this mapping has the risk of producing invalid results when averaging over multiple distinct modes with standard inverse model approaches.

This paper presents an approach aiming to improve the control of overactuated flying systems. First, the actuation modeling error is learned offline using a Gaussian process (GP) regressor. To overcome the challenge arising due to the above-mentioned one-to-many mapping, we formulate the task of finding a suitable wrench command as an iterative optimization problem. We obtain a wrench by optimizing the commanded wrench using the analytical forward model and its learned error model. This yields in expectation the desired behaviour of the system, necessitating no inverse models.

This paper makes the following contributions:

- Proposal of a GP-based model to capture the model-plant mismatch common among overactuated omnidirectional flying vehicles.
- Introduction of an optimization-based method using a first-principles model and a learned GP error model to select a signal correcting for the model-plant mismatch, necessitating only forward models.
- Experimental validation of the proposed approach on a tilt-arm overactuated flying vehicle called Omav [24] (Fig. 6.2), with a reduction of the attitude tracking error of 32% on average.

2 Related work

Optimal control such as the model predictive controller is shown to work well for standard multicopters [92]. However, it is shown in [58] that the model predictive controller has comparable performance to a standard PD controller in case of insufficient model knowledge.

In this section, we focus mainly on the review of approaches that learn the nonlinear, non-negligible model-plant-mismatch, which is then used to improve control performance. A broader review of model learning for control can be found in [93]. We categorize the related work by the types of model being used: forward models and inverse models. A forward model could be a static mapping (e.g. an actuator mapping from individual actuator commands to the resulting wrench in case of neglecting its dynamics) or a dynamics model (the mapping from the state and the control inputs to the derivative of the state), which are uniquely defined. The inverse model is the inverse of the forward model.

Inverse model approach

The inverse model approach aims to find a mapping that maps from the robot state to a feedforward control command. Several approaches [94]–[98] use a GP regression approach to model the inverse dynamics. The implied assumption of these approaches is that there is a unique mapping between the input and output data so that a direct regression is well-defined. Unfortunately, for an overactuated platform this is not the case.

Forward model approach

Forward model learning has been combined with iterative linear quadratic Gaussian algorithm [99], or model predictive control [100]–[102]. These methods use rollouts of the learned dynamics in an optimization-based framework to find a sequence of control actions in real-time. In [100]–[102], the model-plant mismatches are modeled using GPs and embedded into a model predictive controller. These methods are computationally expensive and thus they typically use approximation techniques to propagate the state distribution in the rollout. They experimentally validated these approaches ([100], [101]) on autonomous mobile vehicles. In [102] the authors demonstrated the proposed approach in the simulation of a quadrocopter. One of the aims of our work is to validate the proposed approach on a real system.

Model-based reinforcement learning control may also be employed with a learned forward model. Among these approaches, the control policy is either found through experiments [103], which needs to make sure that the system does not damage the robots, or through simulation [27], [104], [105]. More specifically, the complete actuator model [27] or the residual dynamics ([104], [105]) are first learned using either a neural network or a GP and then embedded into the simulation. A high-performance control policy is then obtained in the simulation through reinforcement learning. These approaches typically require a large amount of training data.

Another approach is given a desired state, the control inputs are solved in an equation that includes the forward model. In [106], in order to improve the tracking performance of a quadrocopter, an acceleration error model is learned using the linear regression. It is then embedded into the acceleration model equation, in which the body angular acceleration command is solved for in real-time as the control inputs. It does not take uncertainty into consideration.

3 Platform description

The Omav (Fig. 6.2) is an overactuated omnidirectional flying vehicle with six tiltable arms in a hexagonal arrangement. A coaxial rotor configuration is rigidly attached to the end of each arm. The propellers spin in the opposite direction, which helps to reduce the gyroscopic effects introduced by the rotation of the arm and thus the tilt motor efforts. The rotation of each arm can be actively controlled by a servo motor, which results in a total of 18 actuators. This setup allows for force- and pose omnidirectionality for almost all configurations and an improved hover efficiency when compared to fixed tilt omnidirectional vehicles.

4 Modeling

This section describes the dynamics of the flying vehicle and presents its actuation model.

4.1 Dynamic model

The position of the vehicle's center of mass with respect to a point fixed in the inertial coordinate system is denoted as $\boldsymbol{p} \in \mathbb{R}^3$. Boldface symbols are used throughout the paper to denote vectors. Two types of coordinate systems are used for the modeling: an inertial coordinate system E, a body-fixed coordinate system B. A vector expressed in a specific coordinate system is indicated by a superscript, for example $\boldsymbol{p}^E$ expresses $\boldsymbol{p}$ in coordinate system E. The vehicle rotates at an angular velocity $\boldsymbol{\omega}_{BE}$. The subscript BE in $\boldsymbol{\omega}_{BE}$ denotes the relative velocity of coordinate system B with respect to E.

Using a simplified physical model, the dynamics model can be written as follows [24]:

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}) + \boldsymbol{v} \tag{5.1}$$

where $\boldsymbol{x} = [\dot{\boldsymbol{p}}^E, \boldsymbol{\omega}^B_{BE}]$ and $\boldsymbol{v} = [\boldsymbol{F}^E/m, (\boldsymbol{I}^B_B)^{-1}\boldsymbol{M}^B]$ with m being the vehicle mass and $\boldsymbol{I}^B_B$ being the vehicle inertia matrix. The translational drag forces and rotational drag torques are neglected, as it is assumed that the vehicle travels at low translational and angular velocities. We assume no model-plant mismatches in (5.1) as the rigid body dynamics are typically known and the parameters such as mass or inertia can be precisely obtained through CAD model.

4.2 Approximate actuation model

The design of the Omav leads to complicated aerodynamic effects (e.g. coaxial rotor configurations and interaction between the adjacent propeller flows), which are hard to model using first-principles. Therefore, the following assumptions are made to approximate the actuation model: 1). It is assumed that the force and torque produced by a stationary propeller are proportional to its angular velocity squared, independent of the vehicle's translational and angular velocity. The resulting force and torque vectors are also assumed to be parallel with each other and perpendicular to the rotor disk. 2). Every rotor has the same torque and force coefficient. 3). The rotors do not interfere with each other, therefore the total force and torque produced by the rotors are the sum of individual force and torque vectors. 4) Body drag forces and torques are neglected due to the low translational and angular velocity of the flying vehicle. 5). It is assumed that the actuators have no temporal dynamics, that is, the past wrench commands do not affect the current wrench.

Based on the above assumptions, a nonlinear approximate mapping $h(\cdot) : \mathbb{R}^{18} \to \mathbb{R}^6$ between the individual actuator commands and the total wrench command $\boldsymbol{W}^B_{\mathrm{cmd}} = [\boldsymbol{F}^B_{\mathrm{cmd}}, \boldsymbol{M}^B_{\mathrm{cmd}}]$ expressed in the body coordinate system B may be established [24]. That is

$$\boldsymbol{W}^B_{\mathrm{cmd}} = h(\boldsymbol{u}_{\mathrm{cmd}}) \tag{5.2}$$

where $\boldsymbol{u}_{\mathrm{cmd}} \in \mathbb{R}^{18}$ contains the twelve propeller thrusts and the six tilt arm angles. For the sake of brevity, the mapping $h(\cdot)$ is not described in detail as it does not affect the subsequent derivation.

This approximate modeling of the actuation introduces modeling errors, this results in a $\boldsymbol{W}^B$ which differs from the intended $\boldsymbol{W}^B_{\mathrm{cmd}}$:

$$\boldsymbol{W}^B = h^*(\boldsymbol{u}_{\mathrm{cmd}}) = h(\boldsymbol{u}_{\mathrm{cmd}}) + \eta(\boldsymbol{u}_{\mathrm{cmd}}) \tag{5.3}$$

where $h^*(\cdot)$ is the true unknown actuation function and $\eta(\cdot)$ is an additive nonlinear modeling error function.

Note that this additive error modeling implies that $\eta(\cdot)$ and $h(\cdot)$ do not need to have the same parametric structure, which allows for more flexibility in the modeling. On the other hand, misidentification of $\eta(\cdot)$ can result in a physically implausible prediction of the generated wrench.

In addition, the functions $h(\cdot)$, $h^*(\cdot)$, and $\eta(\cdot)$ are surjective, that is, for each achievable wrench there are possibly more than one set of feasible inputs.

5 Approach

The problem we are addressing in this report is formulated as follows: Given the system dynamics (5.1), the actuation model (5.2), the unknown modeling error $\eta(\boldsymbol{u}_{\mathrm{cmd}})$, and a given desired wrench $\boldsymbol{W}^B_{\mathrm{des}}$ which is the output of a given model-based nominal controller, find the actuator commands $\boldsymbol{u}_{\mathrm{cmd}}$ such that the $\boldsymbol{W}^B_{\mathrm{des}}$ is achieved. We distinguish between $\boldsymbol{W}^B_{\mathrm{des}}$, which is the output of the controller, and $\boldsymbol{W}^B_{\mathrm{cmd}}$, which is the input to the allocation.

The proposed solution aims to find a compensation signal at the wrench level which, when applied to the nominal controller's command, results in the desired wrench being executed by the system. This reduces the problem to six dimensions since we do not search compensation signals for individual actuator commands $\boldsymbol{u}_{\mathrm{cmd}}$, as they are typically computed on a low-level flight computer that is computationally limited. Furthermore, more training data would be needed for a higher dimensional input space. The approach is summarized as follows and illustrated in Fig. 5.2: the nominal controller computes a desired wrench $\boldsymbol{W}^B_{\mathrm{des}}$ according to the reference trajectory and the state estimate. The discrepancy between the model and the plant is modeled using GPs. The regressed model is input into an optimization framework to obtain an additive compensation signal to the desired wrench, which in case of high prediction uncertainty is filtered towards zero. The resulting command wrench $\boldsymbol{W}^B_{\mathrm{cmd}}$ is then sent to the control allocation, which outputs individual actuator commands $\boldsymbol{u}_{\mathrm{cmd}}$.

This approach has the advantage that the GP and optimization block is modular and therefore can be added to the nominal controller. Furthermore, an important point in our proposed solution is that the uncertainty information provided by the GP is used to prevent uncertain predictions from destabilizing the system. This is a key for safely deploying the proposed controller in a physical system.

5.1 Nominal controller and control allocation

Let $\boldsymbol{e}_p$ denote the position error of the vehicle (i.e, the difference between the actual position $\boldsymbol{p}$ and the desired position $\boldsymbol{p}_{\mathrm{des}}$) and $\dot{\boldsymbol{e}}_p$ its time-derivative. Let D denote the reference body

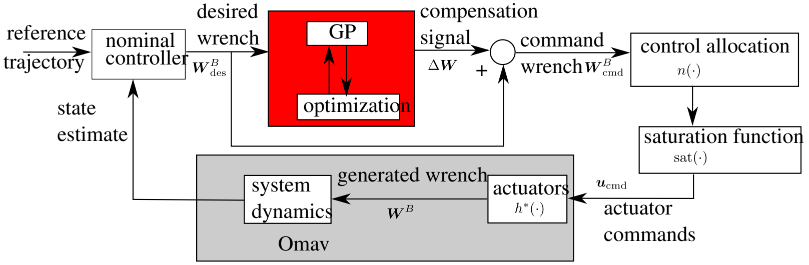


Figure 5.2: A block diagram of the proposed solution. The model-plant mismatch is modeled using GPs and regressed offline using flight data. The model is then fed into an optimization to find a compensation signal $\Delta \boldsymbol{W}$ so that the generated wrench $\boldsymbol{W}^B$ is equal to $\boldsymbol{W}^B_{\text{des}}$.

frame and define the attitude error terms as follows:

$$\boldsymbol{e}_{\mathbf{R}} = \frac{1}{2}(\mathbf{R}^{DB} - \mathbf{R}^{BD})^{\vee} \tag{5.4}$$

$$\boldsymbol{e}_{\omega} = \boldsymbol{\omega}^B_{BD} \tag{5.5}$$

where $\mathbf{R}^{DB}$ is the coordinate transformation matrix from the coordinate system D to B. The mapping $\vee$ maps a skew-symmetric matrix to its $\mathbb{R}^3$ space.

Similar to the position control in [57], the desired wrench $\boldsymbol{W}^B_{\text{des}} = [\boldsymbol{F}^B_{\text{des}}, \boldsymbol{M}^B_{\text{des}}]$ is calculated such that

$$\ddot{\boldsymbol{e}}^E_p = -2\zeta_p \omega_{n,p} \dot{\boldsymbol{e}}^E_p - \omega^2_{n,p} \boldsymbol{e}^E_p \tag{5.6}$$

$$\dot{\boldsymbol{e}}^B_{\omega} = -2\zeta_a \omega_{n,a} \boldsymbol{e}^B_{\omega} - \omega^2_{n,a} \boldsymbol{e}^B_{\mathbf{R}} \tag{5.7}$$

where (ζ_p, $\omega_{n,p}$) and (ζ_a, $\omega_{n,a}$) are a set of damping ratios and natural frequencies for the translational dynamics and attitude dynamics, respectively.

From (5.4) and (5.5) we can see that to first order, $\boldsymbol{e}_{\omega}$ is the time-derivative of $\boldsymbol{e}_{\mathbf{R}}$. This means that if the desired wrench is tracked perfectly, the translation deviation $\boldsymbol{e}_p$ and the rotational deviation $\boldsymbol{e}_{\mathbf{R}}$ will behave like a damped second order system. This makes the parameter tuning more intuitive.

Recall that the actuator dynamics are neglected, thus the wrench and the individual actuators are a static mapping. The wrench command then gets allocated through a chosen mapping $n(\cdot) : \mathbb{R}^6 \to \mathbb{R}^{18}$ [24] and a saturation function $\text{sat}(\cdot) : \mathbb{R}^{18} \to \mathbb{R}^{18}$ in the inner loop

$$\boldsymbol{u}_{\text{cmd}} = \text{sat}(n(\boldsymbol{W}^B_{\text{cmd}})) \tag{5.8}$$

Again, we omit the details of the allocation for brevity. This allocation is essentially a pseudo-inverse mapping of $h(\cdot)$. The mapping $n(\cdot)$ is injective, that is, each set of actuation has at most one corresponding wrench.

5.2 Learning model-plant mismatch

We first derive the model-plant mismatch as a function of the wrench command. Substituting (5.2) and (5.8) into (5.3) yields

$$\begin{aligned} \boldsymbol{W}^B &= h(\boldsymbol{u}_{\text{cmd}}) + \eta(\text{sat}(n(\boldsymbol{W}^B_{\text{cmd}}))) && (5.9)\\ &= \boldsymbol{W}^B_{\text{cmd}} + g(\boldsymbol{W}^B_{\text{cmd}}), && (5.10)\end{aligned}$$

where $g(\cdot) : \mathbb{R}^6 \to \mathbb{R}^6$ is the composition of the injective mapping $n(\cdot)$, $\text{sat}(\cdot)$ and the surjective mapping $\eta(\cdot)$. It may therefore be a surjective mapping. This means that for each $\boldsymbol{W}^B$ there may be more than one set of possible $\boldsymbol{W}^B_{\text{cmd}}$ that achieves it. This leads to the problem mentioned in Section 1: an inverse mapping from $\boldsymbol{W}^B$ to $\boldsymbol{W}^B_{\text{cmd}}$ is a one-to-many mapping and may not directly be learned.

The unknown error function $g(\cdot)$ is modeled as six independent GPs. We make the assumption that the outputs of these GPs are uncorrelated. A GP is a collection of random variables, any finite number of which have a joint Gaussian distribution [107], and can be viewed as a distribution over functions. It is characterized by a mean function $m(\cdot)$ and a covariance function $k(\cdot, \cdot)$, which is a positive-definite kernel function parametrized by a set of hyperparameters.

A GP-based representation was chosen because it handles stochasticity (e.g. measurement noise) naturally, its nonparametric property offers flexibility in the modeling, and gives us information about the uncertainty of predictions made. On the other hand, it is known that the GP prediction does not scale well with data (it has a computational complexity of $\mathcal{O}(N^3)$, where N is the number of data samples). Reducing the computational complexity is beyond the scope of this work. We are currently exploring the use of local GPs to handle the large state space.

We may apply Gaussian process regression to approximate the error function $g_l(\boldsymbol{W}^B_{\text{cmd}})$ according to (5.10), with the subscript l denoting the l-th entry of the function $g(\cdot)$:

$$z = g_l(\boldsymbol{\xi}) + \epsilon \qquad (5.11)$$

where $z \in \mathbb{R}$ denotes the observation of the l-th entry of $\boldsymbol{W}^B - \boldsymbol{W}^B_{\text{cmd}}$, $\epsilon \in \mathbb{R}$ is the measurement noise with Gaussian distribution $\mathcal{N}(0, \sigma^2)$ with σ^2 being the variance, and $\boldsymbol{\xi} \in \mathbb{R}^6$ denotes $\boldsymbol{W}^B_{\text{cmd}}$ for brevity for the remaining of this section.

Let $(\boldsymbol{\xi}_i, z_i)$ denote one observed data point and $X \in \mathbb{R}^{(N+1)\times 6}$, $\boldsymbol{Z} \in \mathbb{R}^{N+1}$ denote the stacked version of $(\boldsymbol{\xi}_0, ..., \boldsymbol{\xi}_N)$, $(z_0, ..., z_N)$ from $N+1$ data points. Conditioned on this data set and a query input $\boldsymbol{\xi}^*$, the expected value and variance of the l-th GP is:

$$\mu_l(\boldsymbol{\xi}^*) = m(\boldsymbol{\xi}^*) + k(\boldsymbol{\xi}^*, X)(K + \sigma^2 I)^{-1}\boldsymbol{Z} \qquad (5.12)$$

$$\sigma_l^2(\boldsymbol{\xi}^*) = k(\boldsymbol{\xi}^*, \boldsymbol{\xi}^*) + \sigma^2 - k(\boldsymbol{\xi}^*, X)(K + \sigma^2 I)^{-1}k(X, \boldsymbol{\xi}^*) \qquad (5.13)$$

where K is a $(N+1) \times (N+1)$ kernel matrix with $K_{ij} = k(\boldsymbol{\xi}_i, \boldsymbol{\xi}_j)$.

In this work we use the squared exponential covariance as the kernel

$$k(\boldsymbol{\xi}_p, \boldsymbol{\xi}_q) = \sigma_f^2 \exp(-\frac{1}{2}(\boldsymbol{\xi}_p - \boldsymbol{\xi}_q)^T \Sigma(\boldsymbol{\xi}_p - \boldsymbol{\xi}_q)). \qquad (5.14)$$

where $\boldsymbol{\xi}_p$ and $\boldsymbol{\xi}_q$ are any two inputs, Σ is the lengthscale matrix which indicates the relevance between two data points and σ_f^2 is the signal variance which is a scale factor. The hyperparameters $(\sigma, \sigma_f, \Sigma)$ for a particular data set can be selected by maximum likelihood estimation, which maximizes the likelihood of the observed outputs given hyperparameters [107].

5.3 Finding compensation signal using optimization

From (5.10) it can be seen that if $\boldsymbol{W}_{\mathrm{cmd}}^B$ is set equal to the controller output $\boldsymbol{W}_{\mathrm{des}}^B$, the actual resulting wrench $\boldsymbol{W}^B$ deviates from $\boldsymbol{W}_{\mathrm{des}}^B$ by the error function $g(\boldsymbol{W}_{\mathrm{des}}^B)$.

In order to achieve the desired wrench $\boldsymbol{W}_{\mathrm{des}}^B$, we propose to add a compensation element $\Delta\boldsymbol{W} \in \mathbb{R}^6$ to the $\boldsymbol{W}_{\mathrm{des}}^B$ when setting $\boldsymbol{W}_{\mathrm{cmd}}^B$, that is,

$$\Delta\boldsymbol{W} := \boldsymbol{W}_{\mathrm{cmd}}^B - \boldsymbol{W}_{\mathrm{des}}^B \tag{5.15}$$

A $\Delta\boldsymbol{W}$ is to be found such that when (5.15) is substituted into (5.10), the produced wrench $\boldsymbol{W}^B$ is equal to the desired wrench $\boldsymbol{W}_{\mathrm{des}}^B$:

$$\boldsymbol{W}_{\mathrm{des}}^B \overset{!}{=} \boldsymbol{W}^B = (\boldsymbol{W}_{\mathrm{des}}^B + \Delta\boldsymbol{W}) + g(\boldsymbol{W}_{\mathrm{des}}^B + \Delta\boldsymbol{W}). \tag{5.16}$$

With $g(\cdot)$ modeled as GPs, we obtain

$$\Delta\boldsymbol{W} + \mu(\boldsymbol{W}_{\mathrm{des}}^B + \Delta\boldsymbol{W}) = 0 \tag{5.17}$$

with $\mu(\cdot)$ being the mean vector function of the GPs.

An online optimization problem is formulated to approximately solve (5.17):

$$\underset{\Delta\boldsymbol{W}}{\text{minimize}} \quad \left\| \mu(\boldsymbol{W}_{\mathrm{des}}^B + \Delta\boldsymbol{W}) + \Delta\boldsymbol{W} \right\|_2^2 \tag{5.18}$$

We optimize this in real time using gradient-based optimization. If the optimal function value is smaller than 1e-4, then we approximately find the compensation according to the mean function of the GP model. Otherwise, we compare the optimal value with the cost function value with $\Delta\boldsymbol{W} = 0$, and pick the corresponding signal that yields a smaller cost.

Remark: Formulating the problem as an optimization problem has several advantages. Firstly, assuming the update rate is high enough, the previous solution is close to the current iteration solution, which is ideally suited as a warm start for the next iteration. Secondly, the search is local, therefore avoiding the jump between feasible solutions that are far away from each other. Due to the saturation function it is possible that the cost function has locally zero gradients. For this, a regularization term $||W_{\mathrm{cmd}}||_2^2$ can be added to the cost function. Finally, the cost function is easy to manipulate and tune. We can add wrench command constraints to the optimization problem or add costs that smoothen the adjacent wrench commands.

5.4 Utilize posterior uncertainty prediction

Since we are using a nonparametric method to learn the dynamics, this might lead to incorrect predictions for inputs that are far outside of the training data and thus destabilize the system. To address this issue we make use of the uncertainty information provided by the GP. After we obtained the optimal solution $\Delta \boldsymbol{W}^*$ from the optimization, the compensation signal is filtered towards zero in case of high uncertainty of the prediction

$$\Delta \boldsymbol{W} \leftarrow a\beta \Delta \boldsymbol{W}_{\mathrm{prev}} + (1-\beta)\Delta \boldsymbol{W}^* \tag{5.19}$$

where $a \in (0,1)$, $\Delta \boldsymbol{W}_{\mathrm{prev}}$ is the previously computed compensation signal,

$$\beta = \frac{1}{1+e^{-\kappa(\sigma-\sigma_{\mathrm{th}})}} \tag{5.20}$$

where $\sigma = \max\{\sigma_l(\boldsymbol{W}^B_{\mathrm{des}} + \Delta \boldsymbol{W}^*), l = 0, \ldots, 5\}$ and with some constants $\sigma_{\mathrm{th}} \in \mathbb{R}$ and $\kappa \in \mathbb{R}$.

For small prediction uncertainty β is near zero and thus the compensation signal directly takes in the optimal value $\Delta \boldsymbol{W}^*$. For σ larger than the threshold σ_{th}, β is near one. a determines how fast the compensation signal $\Delta \boldsymbol{W}$ should be filtered towards zero. κ determines the rapidness of changing of β near σ_{th}.

Once the compensation signal is obtained, it is added to the output of the controller $\boldsymbol{W}_{\mathrm{des}}$. The resulting actuator command $\boldsymbol{u}_{\mathrm{cmd}}$ is obtained through the allocation mapping $n(\cdot)$.

5.5 Comparison with disturbance observer approaches

An alternative approach is to employ an observer to estimate the unmodelled dynamics in real time and inform the controller to reject it. This approach is computationally efficient and has been shown to work well in practice. However, the approach introduces delay into tracking due to its nature of being a filter and furthermore requires expert knowledge to handcraft a disturbance model. The presented approach does not introduce delay and is less dependent on expert knowledge as it uses a data driven approach. The trade-off for this is the increased computational complexity and inability to compensate for unobserved disturbances. A good comparison of the properties of these two methods is presented in [108].

6 Experimental results

The experiments are carried out at an indoor aerial vehicle testbed at Autonomous System Lab, ETH Zurich. A motion capture system provides pose estimates at 100 Hz. The experimental vehicle platform Omav has a diameter of about 80 cm and a weight of 4 kg. The vehicle is also equipped with an onboard NUC i7 computer, which runs computationally expensive modules, and a PixHawk flight controller that takes care of the low-level, low latency tasks. Using this configuration the vehicle is able to run all the necessary algorithms onboard. For a more complete description see [24].

The proposed framework is implemented as a ROS node and evaluated by the performance of attitude trajectory tracking both in simulation and on a real platform. In simulation, the framework is evaluated on a figure-8 trajectory with the vehicle tilting up to 63 degrees in the RotorS Gazebo simulator [109]. For the real experiments, the vehicle follows a smooth feasible reference pitching trajectory from 0 to 60 degrees and back, while remaining stationary with zero roll and yaw. The maximal reference angular acceleration is $1\ \mathrm{rad\,s^{-2}}$. These trajectories were chosen as from experience the most prominent model-plant mismatch occurs on the torque level, especially when the produced force vector is close to the body xy plane, corresponding to the vehicle attitude with high roll or pitch.

6.1 GP model-plant mismatch modeling and learning

This section shows the details of the model-plant mismatch using GP in real experiments and demonstrates its prediction performance on a validation trajectory. As mentioned before, the most prominent model-plant mismatch occurs on the torque level, three single-output GPs are therefore used to model the torque model-plant-mismatch in the experiments and the force model-plant-mismatch is neglected. For the GP regression, the inputs are the wrench command $\boldsymbol{W}^B_{\mathrm{cmd}}$ and the outputs are the components of $(\boldsymbol{M}^B_{\mathrm{meas}} - \boldsymbol{M}^B_{\mathrm{cmd}})$, where $\boldsymbol{M}^B_{\mathrm{meas}}$ is obtained from a high-quality ADIS16448 IMU sensor [110]. The training data is collected while the vehicle tracks a test trajectory with a nominal controller. Sinusoidal excitation is added to the reference trajectory to collect diverse training data. This increases the probability of the learned model covering the output space during validation flight with the proposed strategy. Data points are subsampled from the experimental data offline using the k-medoids algorithm where the Euclidean squared distance between the inputs is used as the distance metric. Through empirical validation we found a hundred subsampled data points are sufficient for this trajectory. The hyperparameters of the GPs are fixed after being estimated using the maximum likelihood methods, as we assume the model-plant-mismatch characteristics along the reference trajectory are repeatable. The GPs are implemented using GPy [111], the optimization uses L-BFGS implemented from nlopt [112], and the k-medoids algorithm is taken from the PyClustering [113]. The learned model is then embedded into the online optimization framework, whose performance is then evaluated and presented in the following.

Fig. 5.3 shows the command torque $\boldsymbol{M}_{\mathrm{cmd}}$, the measured torque $\boldsymbol{M}_{\mathrm{meas}}$, and the predicted torque $\boldsymbol{M}_{\mathrm{pred}}$ of a 10 seconds trajectory. $\boldsymbol{M}_{\mathrm{pred}}$ is $\mu_{\boldsymbol{M}}(\boldsymbol{W}_{\mathrm{cmd}}) + \boldsymbol{M}_{\mathrm{cmd}}$, where $\mu_{\boldsymbol{M}}(\cdot)$ denotes the torque outputs of the GPs. Table 5.1 shows the mean and standard deviation of the absolute differences between the prediction and the measurement on the same trajectory (that is, $|\boldsymbol{M}_{\mathrm{cmd}} - \boldsymbol{M}_{\mathrm{meas}}|$ and $|\boldsymbol{M}_{\mathrm{pred}} - \boldsymbol{M}_{\mathrm{meas}}|$). It can be observed from Table 5.1 that the prediction error on the torque has been reduced by 62%, 70%, and 74% on the body x, y and z axis, respectively. In addition, the mean of the prediction error using the learned model on the body x and y axis are comparable (0.08 and 0.067 Nm), while the error on the body z axis is much larger (0.13 Nm). This is expected as the torque around the body z-axis is mainly generated by tilting the motor arms (especially during hover), while the torque around the body x- and y-axis is mainly induced by the sum of the propeller thrust times the motor arm length. While both servo dynamics and motor dynamics are neglected during the model learning, the prediction around the body z-axis is more affected since the servo dynamics are much slower than the motor dynamics.

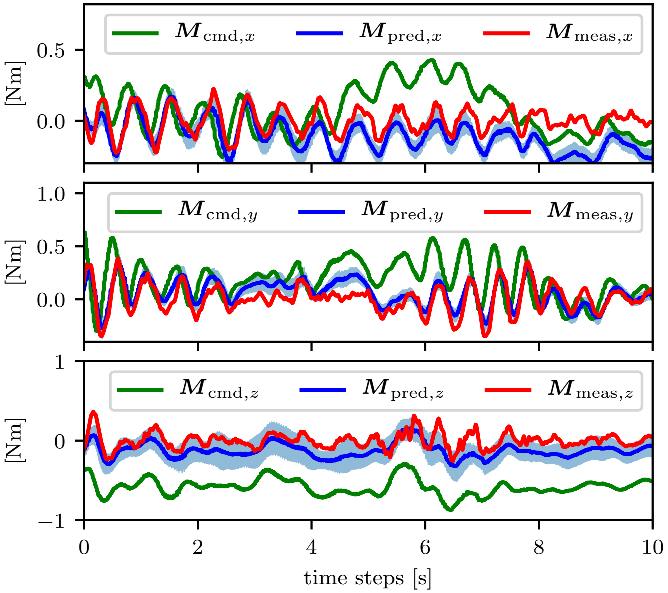


Figure 5.3: Measurement, prediction (our model), and command of the torque vector (nominal model) of a 10 s trajectory. The 3 sigma confidence region of the GP prediction is shaded in blue.

Table 5.1: Prediction performance

[Nm]	body x error	body y error	body z error
nominal mean ± std	0.21 ± 0.16	0.23 ± 0.16	0.50 ± 0.17
learned mean ± std	0.08 ± 0.06	0.07 ± 0.05	0.13 ± 0.09

6.2 Trajectory tracking

Real experiments

The experiments demonstrated in this section compare the trajectory tracking performance between the cases with and without compensation signals. Fig. 5.4 shows a box plot of the absolute values of the attitude tracking errors $|\boldsymbol{e}_{\mathbf{R}}|$ from two experiments. In the first experiment the vehicle tracks the pitch trajectory 10 times consecutively using the control strategy without the compensation, whereas the second experiment repeats the same trajectory using the proposed framework. Running onboard a NUC i7 computer, it takes on average 7.5 ms and maximally 24.0 ms for solving one optimization problem in the proposed framework.

In both cases, an integral term is added to the attitude controller (5.7). The gains of the nominal controller are tuned such that both approaches result in a stable flight in both experiments for the sake of comparison. They are $(\zeta_p, \omega_{n,p}) = (0.707, 3.5)$ and $(\zeta_a, \omega_{n,a}) = (1.3, 3.5)$ and $(0.74, 3.5)$, where the former gain is tuned for roll and pitch axis and the latter gain tuned for the yaw axis. The integral attitude gain is 0.3.

It can be seen that the medians of the absolute attitude tracking errors around the body x and y axis have reduced from 0.07, 0.087 to 0.04, 0.03, respectively, corresponding to a reduction of 43% and 65%. Their box ranges are also reduced from 0.114 and 0.10 to 0.050 to 0.040. On the other hand, there are no improvements shown on the body z axis. The median increases from 0.051 to 0.058 and the box range decreases from 0.066 to 0.054. The tracking in body z-axis has not been improved mainly due to the following two reasons: firstly, it can be seen in Fig. 5.3 that the model-plant mismatch is mostly a constant offset. In the nominal controller, this offset is already mitigated by the integrator term around the body-z axis. Secondly, as mentioned in the previous section, the servo dynamics and the motor dynamics are not learned using the GP. To further improve the tracking performance in body z axis, the input space of the GP could be augmented to include the past history of the wrench commands.

Simulation

A simulation is conducted to demonstrate the generalizability of the learned model. The training data is obtained by tilting the flying vehicle along various body axes up to 70 degrees (with sinusoidal excitation around all three body axes). 200 data points are subsampled to train the GP model. A figure 8 trajectory with the vehicle tilting up to 63 degrees is then flown to evaluate the approach. A comparison of tracking performance with and without compensation signal is shown in Fig. 5.5. The controller gains are again the same in both cases. Although this time the integral gains are set to zero. The tracking errors along 3 body axes have been reduced by $94.5\%, 87.9\%$, and 83.7%, respectively. Similar to the real experiments, the key to having a good model is to excite all body axes during the fight. This ensures that the model can provide informative predictions for all required states.

6.3 Incorporation of prediction uncertainty

The experiment presented in this section aims to demonstrate that even in face of highly uncertain predictions, the system is still robust thanks to the uncertainty provided by the model. Two experiments using the learned model are conducted, one with the uncertainty check (that is, with performing (5.19)) and the other one without. In both cases the vehicle in hover is given a step reference of 40 degrees roll. Since the training data consists of smooth pitching trajectories, the wrench command for a step input in roll is far away from the training data. Fig. 5.6 shows the experiment with the controller with uncertainty check. The standard deviation of the prediction uncertainty in body z direction increases from 0.27 to 0.5. The compensation signal is then filtered towards zero and the flying vehicle is able to remain stable. The parameters $\Sigma_{\mathrm{th}}, \kappa, a$ are heuristically chosen. In another experiment with the controller without the uncertainty check,

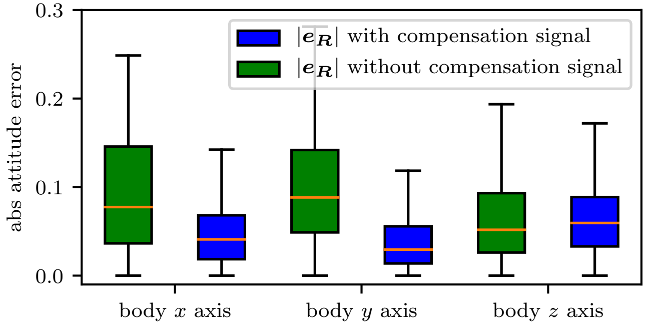


Figure 5.4: A box plot comparison of the tracking performance for a pitching trajectory with and without the compensation signal. In both experiments, the same reference trajectory is executed consecutively 10 times. Note that by small angle approximation $e_\mathbf{R}$ corresponds to the Euler angles (roll, pitch, yaw from the reference frame to the body frame) in radian.

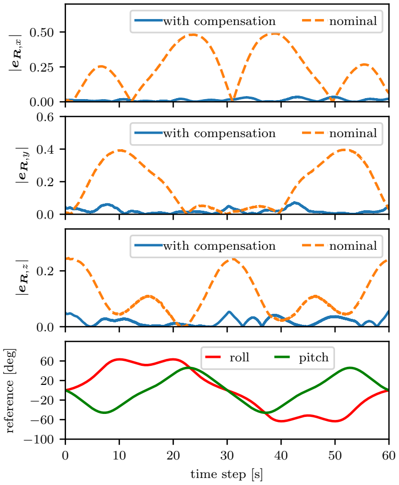


Figure 5.5: The Omav tracking a figure 8 reference trajectory with and without compensation signals in the simulation. Note that by small angle approximation $e_\mathbf{R}$ corresponds to the Euler angles (roll, pitch, yaw from the reference frame to the body frame) in radian.

the system becomes unstable due to the unreliable compensation signal[1]. In this case, it can be expected that the tracking performance of the approach with the uncertainty check cannot be better than the approach without compensation. GP will not predict well in region where no data is available.

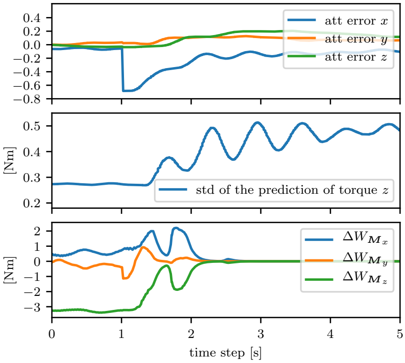


Figure 5.6: This figure shows that for a step input (around 1.0 s) around body-x direction. The desired wrench command is far away from the training data and GP prediction is uncertain at this query input. Therefore the compensation signals ($\Delta \boldsymbol{W}_{\boldsymbol{M}_x}, \Delta \boldsymbol{W}_{\boldsymbol{M}_y}, \Delta \boldsymbol{W}_{\boldsymbol{M}_z}$) get filtered towards zero according to (5.19) and the vehicle remains stable. Note that by small angle approximation $\boldsymbol{e}_{\mathbf{R}}$ corresponds to the Euler angles (roll, pitch, yaw from the reference frame to the body frame) in radian.

7 Conclusion and outlook

This paper presented an approach that addresses the control challenges caused by the model-plant mismatches of an overactuated flying vehicle. Specifically, our approach learns a well-defined forward model using a GP and avoids the multi-valued mapping of the inverse model of the overactuated systems. It finds a compensation signal through an optimization, which then cancels out the effect caused by the model-plant mismatch. In addition, the uncertainty prediction from the GP is exploited to prevent uncertain predictions from destabilizing the flying vehicle. Experiments on a real platform show that the proposed approach reduces the tracking error by 43% and 65% around body x- and y-axis and could prevent destabilizing the system in case of highly uncertain prediction of the model.

---

[1] For a visualization of the behaviors, please see the attached video

Acknowledgement

The authors would like to thank Nicholas Lawrance for thoroughly going through the manuscript. This work was partly supported by funding from NCCR Digital Fabrications, NCCR Robotics and Armasuisse.

Paper III

Active Model Learning using Informative Trajectories for Improved Closed-Loop Control on Real Robots

Weixuan Zhang, Marco Tognon, Lionel Ott, Roland Siegwart, and Juan Nieto

Abstract

Model-based controllers on real robots require accurate knowledge of the system dynamics to perform optimally. For complex dynamics, first-principles modeling is not sufficiently precise, and data-driven approaches can be leveraged to learn a statistical model from real experiments. However, the efficient and effective data collection for such a data-driven system on real robots is still an open challenge. This paper introduces an optimization problem formulation to find an informative trajectory that allows for efficient data collection and model learning. We present a sampling-based method that computes an approximation of the trajectory that minimizes the prediction uncertainty of the dynamics model. This trajectory is then executed, collecting the data to update the learned model. We experimentally demonstrate the capabilities of our proposed framework when applied to a complex omnidirectional flying vehicle with tiltable rotors. Using our informative trajectories results in models which outperform models obtained from non-informative trajectory by 13.3% with the same amount of training data. Furthermore, we show that the model learned from informative trajectories generalizes better than the one learned from non-informative trajectories, achieving better tracking performance on different tasks.

Published in:
Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), 2021
© Open access article distributed under the Creative Commons Attribution License.
DOI:10.1109/ICRA48506.2021.9560869

1 Introduction

Model-based controllers have shown to be useful in various robotics applications. Especially when accurate models are available, these controllers can exhibit impressive performance [114], [115]. Compared to model-free methods such as reinforcement learning, there is no need of training samples to train a control policy. On the other hand, it can be hard to obtain a good dynamical model for complex systems such as humanoid robots [116], race cars on uneven terrains [101], soft robots [117], and novel fully actuated multi-rotor flying vehicles [24] like the one considered in this work (see Fig. 6.2).

One approach to solve the modeling problem is to rely on learning techniques: Through interaction with the real-world and data collection a statistical dynamics model is trained, which is either directly fed into a model-based controller, e.g. [54], [100], [101], [118], or used in simulation to train a control policy [27]. One challenge for these approaches is that often the training data has a different distribution than the test data due to several reasons: first, model uncertainties and feedback controller might lead the system to a state not encountered in a previous data collection routine. Secondly, given partial model knowledge, the region of the data that leads to the best performance is a-priori unknown. Finally, the closed-loop dynamics change as the model used by the controller is updated. One could perform a large number of experiments to cover as much of the input space as possible during training. However, for robotic systems with high-dimensional and continuous state space, the search space typically is too large to be searched exhaustively. Furthermore, the dynamics can change significantly during consecutive experiments, e.g., the crash of a flying vehicle could damage its motors and invalidate the previous training data. Even when considering a specific task, a good model is required in the working area of the state and input spaces, which still might be large. Thus, it is desirable to have an efficient scheme to collect training data locally around the desired task if a precise enough first-principle parametric model is not available or hard to obtain. As these learning techniques are nonparametric, common tools from parametric system identification [28], e.g., persistence of excitation, are not applicable.

One idea is to use the statistical information learned from training data to infer the region where to sample data, thus improving sampling efficiency. This is a well-known approach in machine learning called *active learning* [119]. In this paper we exploit such an idea: we rely on the previously learned statistical model to get an estimate of the region of interest. We then generate an informative trajectory that reduces the overall uncertainty in the estimated region. This trajectory is then executed in the real world to collect data.

More specifically, in a first step, possible informative locations are inferred in simulation from the previously learned model. Then, different informative trajectories are sampled and evaluated according to a cost metric, which is defined as the integral of the predictive uncertainty over these possible locations. The most informative trajectory is then selected and executed on the real robot to collect the data. As a result, the model learned from this informative trajectory should result in improved control performance and a better generalization. The latter is achieved because the informative trajectory reduces the uncertainty over a large region of state and input space.

The contributions of this paper are summarized as follows:

- A formal mathematical formulation of the problem of efficient data collection for learning dynamics model.

- A practical strategy to *efficiently* collect task-relevant data that improves the model-based control performance when used to update the learned model.

- Real experimental results conducted on a complex overactuated omnidirectional flying system with nonlinear dynamics and 18 actuators. For a figure-8 trajectory, two runs of trajectory flight lead to an angular acceleration tracking error reduction of 54.4%

1.1 Related work

Active learning in robotics is mostly defined in a regression setting: a regression mapping between an input and an output space is to be learned while the sample complexity is minimized. The exploration of the sample space is typically driven by some metric often consisting in variants of the expected informational gain.

Considering active dynamics model learning, existing work includes the use of information gain on parameter estimates ([120], [121]), Gaussian processes ([122]–[124]), and neural networks [125]. They typically generate trajectories that minimize a defined metric, trading off between exploration and exploitation. Aside from the parameter estimates approach, little work is done on real robots.

We can also distinguish approaches depending whether the trajectory generation is performed online or offline. The online approaches are often done in a receding horizon fashion [126], where trajectories are regenerated at a certain frequency on the fly during experiments. This constant update helps reducing the distance between the desired inputs and achieved ones. However, this approach is computationally intensive. While exploring a state of interest, the robot cannot always stay stationary waiting for a new planned trajectory. Up to date, this method exists only in theoretical works validated in simulation [122], [123], [127].

The offline approach has the shortcoming that the planned trajectory has a larger distance to the executed one, but applicable on real robots. In [125], the trajectory generation is formulated as a variable-constrained problem and validated on a simulated overactuated robotic spacecraft. In [124], the input trajectories are parametrized by consecutive trajectory sections and the most informative and safe trajectory is then executed. Their formulation did not take into account closed-loop control. The method is applied on a high-pressure fluid injection system. Our investigation belongs to this approach: we make use of the previously learned model and simulations to reduce the deviation of the executed trajectory to the desired one. In this work, we demonstrate that this approach works for complex robots and efficiently improve control performance.

2 Modeling and Problem Statement

We consider a generic system whose dynamics in the discrete time domain are described by:

$$\boldsymbol{x}[k+1] = f(\boldsymbol{x}[k], \boldsymbol{u}[k]), \tag{6.1}$$

where $f(\cdot,\cdot)$ is a Lipschitz-continuous function[1] and represents the *true dynamics*. $\boldsymbol{x}[k] \in \mathcal{X} \subset \mathbb{R}^n$ and $\boldsymbol{u}[k] \in \mathcal{U} \subset \mathbb{R}^m$ describes the state and the control input of the dynamical system at time $k \in \mathbb{N}_{\geq 0}$. To simplify the notation, $\boldsymbol{x}[k]$ denotes $\boldsymbol{x}(kT)$ where $T \in \mathbb{R}_{>0}$ is the sampling time. We remark that a perfect knowledge of $f(\cdot,\cdot)$ is in general not available. We might have only an estimation of it denoted by $\hat{f}(\cdot,\cdot)$.

The considered task consists in a trajectory tracking problem. A desired *task state trajectory* is defined by the sequence of state values $X_t^r = (\boldsymbol{x}_t^r[0], \ldots, \boldsymbol{x}_t^r[N])$ in the time horizon $N \in \mathbb{N}_{>0}$. Throughout this paper, we use a capitalized letter to indicate a sequence of vectors with a time horizon of N. The subscript $\star_t$ is used to denote the quantities related to the task trajectory tracking problem, while the superscript $\star^r$ denotes reference state or input. We first introduce the following assumption

Assumption 1. *A model-based controller $\pi(\cdot,\cdot,\cdot)$ that is a function of a reference state $\boldsymbol{x}^r[k]$, a state $\boldsymbol{x}[k]$, and an estimated dynamics model $\hat{f}$ is provided*

$$\boldsymbol{u}[k] = \pi(\boldsymbol{x}^r[k], \boldsymbol{x}[k], \hat{f}). \tag{6.2}$$

Furthermore, if $\boldsymbol{x}[0] = \boldsymbol{x}_t^r[0]$, and $\hat{f}(\boldsymbol{x},\boldsymbol{u}) = f(\boldsymbol{x},\boldsymbol{u})$ for every $(\boldsymbol{x},\boldsymbol{u}) \in \mathcal{Z} = \mathcal{X} \times \mathcal{U}$, then

$$\boldsymbol{x}_t[k+1] = f(\boldsymbol{x}_t[k], \pi(\boldsymbol{x}_t^r[k], \boldsymbol{x}_t[k], \hat{f})) = \boldsymbol{x}_t^r[k+1], \tag{6.3}$$

for every $k = 0, \ldots, N-1$.

This condition describes a perfect tracking of the desired trajectory given a perfect modeling. We further remark that the given task state trajectory is feasible, so that there exists at least one task input trajectory to achieve it We define the sequence of inputs that provides perfect tracking as $U_t^r = (\boldsymbol{u}_t^r[0], \ldots, \boldsymbol{u}_t^r[N])$, called *task input trajectory*.

Objective 1. *Considering the closed-loop system* (6.1) *and* (6.2)*, our objective is to define an active learning method aiming at optimizing the data collection process to*

- *make it more efficient (less experiments and data points),*
- *improve the precision of the learned model,*
- *improve the generalizability of the learned model,*
- *minimize the tracking error.*

We shall show how the learning problem can be reformulated to address such objectives.

Without loss of generality, we can decompose the true dynamics into two components:

$$f(\boldsymbol{x}[k], \boldsymbol{u}[k]) = h(\boldsymbol{x}[k], \boldsymbol{u}[k]) + g(\boldsymbol{x}[k], \boldsymbol{u}[k]), \tag{6.4}$$

where $h(\cdot,\cdot)$ is called *first principles dynamics*, corresponding to the model reflecting physical laws. We consider $h(\cdot,\cdot)$ to be known. $g(\cdot,\cdot)$ is called *residual dynamics*, corresponds to all

---

[1] This is a common assumption that does not limit the validity of the work since most of the considered robotic systems have Lipschitz-continuous dynamics.

other elements not modeled by h. g is assumed unknown and we only have an estimation denoted by $\hat{g}$.

This modeling allows to exploit the knowledge we already have about the system, reducing the learning effort and making it possible to employ several model-based controllers.

Once again, it is clear that, considering the control law (6.2) with $\hat{f} = h + \hat{g}$, the closed loop system achieves perfect tracking if $\hat{g}(\boldsymbol{z}) = g(\boldsymbol{z})$ for every $\boldsymbol{z} := (\boldsymbol{x}, \boldsymbol{u}) \in \mathcal{Z}$. For simplicity we use $\boldsymbol{z}$ to denote the state-input pair $(\boldsymbol{x}, \boldsymbol{u})$. We assume that a Bayesian prior model [128] over the residual dynamics is given. That is, for a given test point $\boldsymbol{z}$, the belief of the value of $g(\boldsymbol{z})$ follows a Gaussian probability distribution $\mathcal{N}_{\boldsymbol{z}}\left(\mu(\boldsymbol{z}), \sigma^2(\boldsymbol{z})\right)$. We denote the mean and variance of $\mathcal{N}_{\boldsymbol{z}}(\cdot,\cdot)$ as $\mu(\boldsymbol{z}) \in \mathbb{R}^n$ and $\sigma^2(\boldsymbol{z}) \in \mathbb{R}^{n\times n}_{\geq 0}$, respectively. Note that the distribution is a function of the test point $\boldsymbol{z}$. We consider the estimation of the residual dynamics as $\hat{g}(\boldsymbol{z}) = \mu(\boldsymbol{z})$, which brings to $\hat{f}(\boldsymbol{z}) = h(\boldsymbol{z}) + \mu(\boldsymbol{z})$.

Let Z denote the state and input trajectory pair (X, U). The prior model can be updated to a posterior model from trajectory data subsampled from Z. In particular, the updated model is described by the posterior mean $\mu(\boldsymbol{z}|Z)$ and posterior variance $\sigma^2(\boldsymbol{z}|Z)$.

We then introduce the following assumption for the Bayesian model:

Assumption 2. *Given two sets of data from trajectory Z_1 and Z_2, for all $\boldsymbol{z} \in \mathcal{Z}$ and $j = 0, \ldots, n$, $\sigma_j^2(\boldsymbol{z}|Z_1) < \sigma_j^2(\boldsymbol{z}|Z_2)$ leads to $|\mu_j(\boldsymbol{z}|Z_1) - g_j(\boldsymbol{z})| < |\mu_j(\boldsymbol{z}|Z_2) - g_j(\boldsymbol{z})|$. Furthermore, if $\sigma_j^2(\boldsymbol{z}|Z_1)$ approaches zero, $|\mu_j(\boldsymbol{z}|Z_1) - g_j(\boldsymbol{z})|$ approaches zero. The subscript $*_j$ is used to denote the j-th vector element or j-th diagonal element.*

The intuition behind this practical assumption is that a high-quality observation (which is possible in robotic applications) near the test point reduces the uncertainty at the test point and therefore reduces the estimation error.

To improve the knowledge of $\hat{g}$, suitable data must be collected. A possible solution is to simply run the task trajectory, over and over, until sufficient data is collected to obtain a good model around (X_t^r, U_t^r), or Z_t^r. However, this would require many trials to ensure the collected data is informative enough.

Departing from this basic approach, here we aim to design an algorithm that automatically derives reference state trajectories X_i^r, called *informative state trajectories*. These trajectories aim to efficiently collect data to improve the prior model, thus reducing the task trajectory tracking error when using the control law (6.2). Let $\boldsymbol{x}_i[k]$ and $\boldsymbol{u}_i[k]$ denote the inputs and states obtained letting the closed-loop system evolve using X_i^r as reference trajectory. In details

$$\begin{aligned} \boldsymbol{x}_i[k+1] &= f(\boldsymbol{x}_i[k], \boldsymbol{u}_i[k]) \\ \boldsymbol{u}_i[k] &= \pi(\boldsymbol{x}_i^r[k], \boldsymbol{x}_i[k], \hat{f}), \end{aligned} \tag{6.5}$$

with $\boldsymbol{x}_i[0] = \boldsymbol{x}_i^r[0]$. The subscript $*_i$ is used to denote quantities related to the informative trajectory tracking problem.

The following problem is then formulated:

Problem 1. *Find X_i^r as solution of:*

$$\begin{aligned} \min_{X_i^r} \quad & \sum_{k=0}^{N} \|\boldsymbol{x}_t^r[k] - \boldsymbol{x}_t[k]\|_2^2 \\ \textit{s.t.} \quad & \boldsymbol{x}_t[k+1] = f(\boldsymbol{x}_t[k], \boldsymbol{u}_t[k]) \\ & \boldsymbol{u}_t[k] = \pi(\boldsymbol{x}_t^r[k], \boldsymbol{x}_t[k], \hat{f}_{\text{posterior}}) \\ & \hat{f}_{\text{posterior}} = h + \hat{g} \\ & \hat{g}(\boldsymbol{z}) = \mu(\boldsymbol{z}|Z_i),\ Z_i \textit{ as in } (6.5). \end{aligned} \tag{6.6}$$

3 Generation of Informative Trajectories

This section introduces an optimization problem aiming at minimizing an informative cost metric, the solution of which is equivalent to the solution of (6.6). The problem is solved by practical approximations leading to a sampling-based trajectory generation algorithm.

3.1 Minimization of the informative cost

Solving (6.6) is definitely not a trivial problem, even using sampling-based methods. In fact, since we do not know f, solving (6.6) would require to run two experiments for every sampled informative state trajectory X_i^r, using as reference firstly X_i^r and then X_t^r.

In order to make the problem feasible from a practical point of view, let us recall that using the model-based controller (6.2), we can achieve perfect tracking by having the perfect knowledge of $\hat{g}$ for all $\boldsymbol{z} \in \mathcal{Z}_t^r$ where

$$\mathcal{Z}_t^r = \{\boldsymbol{z} \in \mathcal{Z} \,|\, \exists\, k \in (0, \dots, N) \text{ s.t. } \boldsymbol{z} = \boldsymbol{z}_t^r[k]\}, \tag{6.7}$$

contains the pairs state/input that achieve perfect tracking of the task trajectory.

According to Assumption 2, a possible idea is to improve the model by minimizing the uncertainty of the prior model, i.e., $\sigma^2(\boldsymbol{z})$ for all $\boldsymbol{z} \in \mathcal{Z}_t^r$. Thus, we reformulate (6.6) as

$$\min_{X_i^r} \sum_{\mathcal{Z}_t^r} \sigma^2(\boldsymbol{z}|Z_i). \tag{6.8}$$

Recall that Z_i are computed as in (6.5). Note that the solution of (6.8) allows to minimize the modeling error (Assumption 2) which in turns leads to the minimization of tracking error (Assumption 1). Therefore, the solution of (6.8) is also the solution of Problem 1.

Notice that we focus on reducing the informative cost on the space relevant to the task instead of the entire state/input space $\mathcal{Z}$. However, from experimental considerations, we remark that improving the model only in $\mathcal{Z}_t^r$ is not enough to achieve good tracking performance. In fact, initial errors, noisy measurements, and external disturbances might make the system deviate from Z_t^r, visiting pairs input/state not included in $\mathcal{Z}_t^r$ for which the model could be imprecise. Therefore, to achieve good tracking also in these non-ideal and more realistic conditions, we propose to improve the learning of the model by solving (6.8) not only for the points in $\mathcal{Z}_t^r$, but also for the ones that are sufficiently close, i.e., for all $\boldsymbol{z} \in \mathcal{Z}_{\Delta_t^r}$ where

$$\mathcal{Z}_{\Delta_t^r} = \{\boldsymbol{z} \in \mathcal{Z} \,|\, \exists\, \boldsymbol{z}_t^r \in \mathcal{Z}_t^r \text{ s.t. } \|\boldsymbol{z} - \boldsymbol{z}_t^r\| \le \epsilon\}, \tag{6.9}$$

with[2] $\epsilon \in \mathbb{R}_{\ge 0}$ being a heuristic that can be tuned to control the exploratory behavior of the informative trajectory. Problem (6.8) becomes:

$$\min_{X_i^r} \int_{\mathcal{Z}_{\Delta_t^r}} \sigma^2(\boldsymbol{z}|Z_i) d\boldsymbol{z} \tag{6.10}$$

From now on, we refer to the objective function to be minimized as *informative cost*.

The problem cannot be solved in a closed-form way. Thus, we propose to use a sampling-based optimization method [129] that consists in sampling different informative state trajectories X_i^r and choose the one that shows the smallest informative cost. However, this approach cannot be directly employed due to some practical issues:

1. To compute the informative cost for every sampled informative trajectory we should theoretically run an experiment. This is clearly time consuming and does not meet the goals of Objective 1.
2. We do not know U_t^r. From its definition, we should know f to compute U_t^r given X_t^r. Therefore, we cannot directly compute $\mathcal{Z}_{\Delta_t^r}$.
3. It is not straightforward how we can compute the integral of the posterior variance over $\mathcal{Z}_{\Delta_t^r}$.
4. It is not straightforward how we can efficiently sample informative trajectories.

Each of these four problems are individually addressed below proposing a few approximations that make (6.10) solvable from a practical point of view. This allows for deploying the method on real robots.

3.2 Approximations of the optimization problem

Approximation of the dynamical constraints

During the search for the optimal informative state trajectory, given a candidate informative state trajectory $X_{i,\text{cand}}^r$, instead of computing the posteriori variance based on the data collected from a real experiment, Z_i, we compute it based on the data collected from a simulation of the system, $\bar{Z}_i$. In details, $\bar{Z}_i$ is the output of the simulated closed-loop system using $X_{i,\text{cand}}^r$ as

---

[2] With an abuse of notation, we consider $\|\boldsymbol{z} - \boldsymbol{z}_\star\| = \left\| [\boldsymbol{x}^\top \ \ \boldsymbol{u}^\top]^\top - [\boldsymbol{x}_\star^\top \ \ \boldsymbol{u}_\star^\top]^\top \right\|$. A weighted norm can also be used to normalize the components of state and input vectors.

reference trajectory, i.e.,

$$\begin{aligned}\bar{\boldsymbol{x}}_i[k+1] &= h(\bar{\boldsymbol{x}}_i[k], \bar{\boldsymbol{u}}_i[k]) + \boldsymbol{g}'(\bar{\boldsymbol{z}}_i^j[k]) \\ \bar{\boldsymbol{u}}_i[k] &= \pi(\boldsymbol{x}^r_{i,\mathrm{cand}}[k], \bar{\boldsymbol{x}}_i[k], \hat{f}),\end{aligned} \tag{6.11}$$

where $\boldsymbol{g}'(\bar{\boldsymbol{z}}_i^j[k])$ is a sample of the Bayesian model of the residual dynamics, using the probability distribution $\mathcal{N}_{\bar{\boldsymbol{z}}_i^j[k]}(\cdot,\cdot)$. The bar $\bar{*}$ is used to denote quantities related to the simulation throughout this paper.

Approximation of $\mathcal{Z}_{\Delta_t^r}$

Since we do not know f, we cannot compute U_t^r, and therefore neither $\mathcal{Z}_{\Delta_t^r}$. In this section we show how we can get an estimation of $\mathcal{Z}_{\Delta_t^r}$, denoted by $\hat{\mathcal{Z}}_{\Delta_t^r}$, exploiting the current estimation of f.

We firstly uniformly sample the state and input spaces, $\mathcal{X}$ and $\mathcal{U}$, creating the sets $\mathcal{X}' \subset \mathcal{X}$ and $\mathcal{U}' \subset \mathcal{U}$, respectively. We then simulate the closed-loop system M times using X_t^r as reference, We obtain M state and input trajectories $\bar{Z}_t^j$ where

$$\begin{aligned}\bar{\boldsymbol{x}}_t^j[k+1] &= h(\bar{\boldsymbol{x}}_t^j[k], \bar{\boldsymbol{u}}_t^j[k]) + \boldsymbol{g}'(\bar{\boldsymbol{z}}_t^j) \\ \bar{\boldsymbol{u}}_t^j[k] &= \pi(\boldsymbol{x}_t^r[k], \bar{\boldsymbol{x}}_t^j[k], \hat{f}).\end{aligned} \tag{6.12}$$

Finally, we compute $\hat{\mathcal{Z}}_{\Delta_t^r}$ as

$$\begin{aligned}\hat{\mathcal{Z}}_{\Delta_t^r} = \{\boldsymbol{z} \in \mathcal{X}' \times \mathcal{U}' \mid \exists\, k \in (0,\dots,N) \text{ and} \\ j \in (1,\dots,M) \text{ s.t. } \left\|\boldsymbol{z} - \bar{\boldsymbol{z}}_t^j[k]\right\| \leq \epsilon\}.\end{aligned} \tag{6.13}$$

Similar to (6.9), the threshold ϵ is a heuristic that controls the exploration of the informative trajectory. With large ϵ, the optimal trajectory should show a more exploratory behavior.

Approximation of the informative cost

We replace $\mathcal{Z}_{\Delta_t^r}$ with $\hat{\mathcal{Z}}_{\Delta_t^r}$ in (6.10) and this integral can be approximately solved using numerical integration such as Monte-Carlo integration: we uniformly sample S pairs $\boldsymbol{z}^j$ in $\mathcal{Z}_{\Delta_t^r}$, where $j = 1,\dots,S$, creating the set $\mathcal{Z}'_{\Delta_t^r}$. We then approximate the informative cost in (6.10) as

$$\frac{V}{S} \sum_{\boldsymbol{z}^j \in \hat{\mathcal{Z}}'_{\Delta_t^r}} \sigma^2(\boldsymbol{z}^j | Z_i), \tag{6.14}$$

where V is the volume $\int_{\hat{\mathcal{Z}}_{\Delta_t^r}} d\boldsymbol{z}$. Notice that for the different sampled informative trajectories, V and S remain constant and therefore can be omitted in the optimization.

Parametrization of the informative trajectory

Since we want to improve the knowledge of the model in $\mathcal{Z}_{\Delta_t^r}$, it is natural to think that the informative state trajectory X_i^r should be "close" to the task state trajectory X_t^r. Therefore, given a generic k, we define $\boldsymbol{x}_i^r[k]$ such that

$$\boldsymbol{x}_i^r[k] = \boldsymbol{x}_t^r[k] + \delta\boldsymbol{x}[k]. \tag{6.15}$$

Now, sampling informative state trajectories means sampling "deviations" from the task state trajectory. To reduce the sampling space, which has the same dimension of $\mathcal{X}$, we parametrize $\delta\boldsymbol{x}$ using the Discrete Fourier Transform (DFT)

$$\delta\boldsymbol{x}[k] = \frac{1}{P}\sum_{p=0}^{P-1} \boldsymbol{\Theta}_{\boldsymbol{x}}^\top \boldsymbol{e}_p e^{j\frac{2\pi p}{P}k}, \tag{6.16}$$

where $P \in \mathbb{N}_{>0}$, j is the complex operator, $\boldsymbol{e}_p \in \mathbb{R}^P$ is a vector with 1 in place p and 0 elsewhere, and $\boldsymbol{\Theta}_{\boldsymbol{x}} \in \mathcal{O}_{\boldsymbol{x}} \subset \mathbb{R}^{n \times P}$ is the state parameter matrix.

From sampling every state of the informative trajectory, we now samples only fewer parameters. Furthermore, the rationale behind the use of DFT parametrization is that it gives us a more intuitive control of the frequencies of excitation. We can use fewer parameters to generate excitation signals that are spread through frequencies of interest. Intuitively, the deviation signal can be seen as an excitation signal added around the task state trajectory. As a result, the algorithm inherently explores locally around the task trajectory.

3.3 Sampling-based optimization algorithm

Considering the previous simplifications, (6.10) becomes

$$\begin{aligned} \min_{\boldsymbol{\Theta}_{\boldsymbol{x}}} \quad & \sum_{\boldsymbol{z}^j \in \hat{\mathcal{Z}}'_{\Delta_t^r}} \sigma^2(\boldsymbol{z}^j | \bar{Z}_i) \\ \text{s.t.} \quad & \hat{\mathcal{Z}}_{\Delta_t^r} \text{ as in (6.13)}, \ \ \boldsymbol{z}^j \text{ as in (6.12)}, \\ & \boldsymbol{x}_i^r[k] = \boldsymbol{x}_t^r[k] + \delta\boldsymbol{x}[k], \ \ \delta\boldsymbol{x}[k] \text{ as in (6.16)}. \end{aligned} \tag{6.17}$$

Practically, to solve (6.17) we used a Monte-Carlo sampling-based method. The algorithm follows the next steps which require the simulation of the system only:

1. Uniformly sample a set of parameters $\boldsymbol{\Theta}_{\boldsymbol{x}} \in \mathcal{O}_{\boldsymbol{x}}$ and compute several informative state trajectories as in (6.15) and (6.16);

2. Simulate multiple times the system with the sampled residual model $\boldsymbol{g}'$ according to the prior model. Each informative state trajectories computed at step 1 is used as reference;

3. For every simulation, collect the data relative to the performed trajectory, $\bar{Z}_i$, and update the Bayesian model of the residual dynamics;

4. Compute $\hat{\mathcal{Z}}_{\Delta_t^r}$ as explained in 3.2;

5. Evaluate the information cost in $\hat{\mathcal{Z}}_{\Delta_t^r}$ according to (6.14) associated to every new updated model;

6. Select the informative state trajectory corresponding to the minimum information cost.

Once the informative state trajectory supposed to provide the best model update is selected, it is used as reference in a real experiment. The relative collected data, Z_i, is then employed to update the prior model. The full process can be repeated from step 1), to find a new state informative trajectory that would allow to further improve the model accuracy, and in turn to reduce the tracking error.

Remark: Note that the quality of the approximate solution depends on the quality of the prior model. Therein lies the purpose of this algorithm: Within each iteration, the quality of the prior model improves, and the solution to the approximated problem converges towards the true optimum. Consequently, this helps improving the prior model.

4 Application to an aerial robot: the Omav

This section shows how the above framework is applied on an omnidirectional flying vehicle, called *Omav* [24]. The Omav (Fig. 6.2) is an overactuated omnidirectional flying vehicle with six tiltable arms in a hexagonal arrangement. A coaxial rotor configuration is rigidly attached to the end of each arm. The rotation of each arm can be actively controlled by a servo motor, which results in a total of 18 actuators. Although the setup enhances the motion and interaction capabilities, aerodynamic disturbances among the rotors, unknown servo dynamics, backlashes, and other mechanical inaccuracy are difficult to be modeled and included in standard model-based controller. This makes Omav a suitable testbed to validate the proposed method for active model learning.

The state of the Omav is given by $\boldsymbol{x} = [\boldsymbol{p}^\top \;\; \boldsymbol{\eta}^\top \;\; \dot{\boldsymbol{p}}^\top \;\; \boldsymbol{\omega}^\top]^\top \in \mathcal{X} \subset \mathbb{R}^{12}$. In order, $\boldsymbol{x}$ includes the position, attitude (expressed in Euler angles), linear velocity, and angular velocity of the vehicle. As input of the system we consider the commanded wrench, i.e., the total force and moment commanded to the vehicle[3], $\boldsymbol{u} = [\boldsymbol{f}_{\mathrm{cmd}}^\top \;\; \boldsymbol{\tau}_{\mathrm{cmd}}^\top]^\top \in \mathcal{U} \subset \mathbb{R}^6$. We assume that an allocation policy is implemented to transform $\boldsymbol{u}$ into low level commands for the servos and the motors [24]. Finally, the dynamics of the Omav can be written as in (6.4), where h is derived using standard Newton-Euler equations. Notice that h is linear with respect to the input and can be written as

$$h(\boldsymbol{z}) = l(\boldsymbol{x}) + \boldsymbol{u}, \tag{6.18}$$

where $l(\boldsymbol{x})$ includes all the terms that do not depend on $\boldsymbol{u}$.

On the other hand, g includes all previously mentioned unmodeled dynamic behaviors that cannot be easily captured with first principles. Considering the last six row of the dynamics (the linear and angular accelerations), we can consider $g(\boldsymbol{z})$ as the mismatch between the commanded wrench and the actuated one.

---

[3] For simplicity, we consider force and moment scaled by mass and inertia, respectively.

The controller tries to implement a feedback linearization control law with a PID action on the position and attitude errors. In particular, given a reference task trajectory, X_t^r, and a priori model for g, the controller $\pi(\boldsymbol{x}_t^r[k], \boldsymbol{x}[k], \hat{f})$ tries to find the input $\boldsymbol{u}[k]$ that solves the following optimization problem

$$\min_{\boldsymbol{u}[k]} \|\boldsymbol{x}^{\star}[k] - l(\boldsymbol{x}[k]) - \boldsymbol{u}[k] - \hat{g}(\boldsymbol{u}[k])\| , \tag{6.19}$$

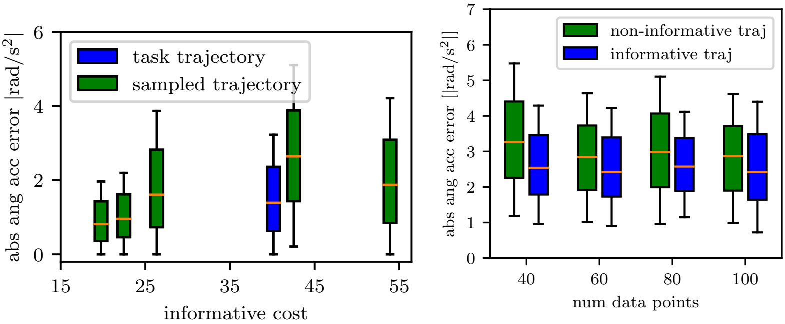
where $\boldsymbol{x}^{\star}[k] = \boldsymbol{K}(\boldsymbol{x}_t^r[k] - \boldsymbol{x}[k]) + \boldsymbol{K}_I \sum_{j=0}^{k}(\boldsymbol{x}_t^r[j] - \boldsymbol{x}[j])$ is the PID action, with $\boldsymbol{K}, \boldsymbol{K}_I \in \mathbb{R}^{12\times 12}$ positive definite matrices. For the details about the implementation of such an optimization, we refer the interested reader to [54].

From experimental observations, we remark that the residual dynamics regarding the differential kinematics and linear acceleration (first nine rows) is almost negligible with respect to the one regarding the angular acceleration (last three rows). In other words, the mismatch between commanded and actual force is much smaller than the one between commanded and actual torque. For this reason, in this first work, we focus on the attitude dynamics, applying the proposed active dynamics learning only on the last three rows of the system dynamics. These mismatches are modeled as three independent single-output Gaussian processes with $\boldsymbol{u}$ as the training input and the torque model mismatch as training output. We neglect the rotational drag torque acting on the vehicle since the vehicle is mostly operating with low angular velocities, thus $\hat{g}$ is modeled independent of the state.

5 Experimental results

The experimental platform is the omnidirectional aerial vehicle in Fig. 6.2: the Omav. The Omav weighs 4 kg and is equipped with a NUC i7 computer and a PixHawk flight controller. This configuration allows to run all the necessary algorithms onboard implemented in a ROS framework. A motion capture system provides pose estimates at 100 Hz. For a more complete description of the testbed see [24].

As stated in Section 4, the proposed method has been implemented and evaluated focusing on the rotational dynamics. For the learned Gaussian process model, data points are subsampled from the experimental data using the k-medoids [130] algorithm where the Euclidean squared distance between the inputs is used as the distance metric. Throughout the experiments, squared exponential kernels are used. The deviation $\delta\boldsymbol{x}[k]$ is sampled around x, y, z-axis on the angular acceleration level, constraining to be below 2 Hz. Note that this is equivalent to giving $\delta\boldsymbol{x}[k]$ on the angular velocity. For simplicity, we limit the number of frequencies P to 2 and allow the frequency locations to be sampled along with its magnitude. This yields a total of 12 coefficients to be sampled. The simulation framework is set up using RotorS Gazebo simulator [109]. In this section we use “non-informative trajectory” to describe the case where only the task trajectory is used to collect the data to update the model.



(a) A comparison of the tracking performance using the model learned from sampled trajectories and task trajectory.

(b) A comparison of the tracking performance between informative trajectory and task trajectory for the same number of data points.

Figure 6.1: Comparisons of tracking performance.

5.1 Correlation between informative cost and tracking error

An experiment is conducted to investigate whether the tracking error defined in problem (6.6) is correlated to the informative cost defined in (6.17). The Omav is asked to follow a pitching trajectory up to 60 degrees in pitch and 1 rad/s^2 in pitch angular acceleration, similar to previous work [54]. A prior model is built by collecting the data from executing the task trajectory. Next, five sampled candidate informative trajectories and the task trajectory are executed and six learned models are built accordingly. They are then evaluated on the test data generated by the prior model. ϵ is heuristically tuned by simulation computing the average distance between the commanded wrench and the achieved wrench. The tracking performance of the angular acceleration[4] along the y-axis using these models are shown in Fig. ??. It can be seen that there is a clear correspondence between the informative cost and the tracking error. Furthermore, the model learned from the task trajectory does not yield the best tracking performance.

5.2 Comparison between informative and task trajectory

To compare the efficiency of the informative and non-informative trajectory, a figure-8 in attitude (with roll and pitch up to 26 degrees) with constant position is given as a task trajectory (see Fig. 6.3). We compute the prior model running the task trajectory for the first time. Then 20 trajectories are randomly generated and evaluated in simulation as explained in Section 3.3 using the prior model. The most informative trajectory (lowest informative cost) and the task trajectory are then executed and the data are recorded for both trajectories. We subsampled 20,

---

[4] Notice that evaluating the angular acceleration tracking is equivalent to evaluate the error between actual and commanded torque which strongly depends on the model accuracy.

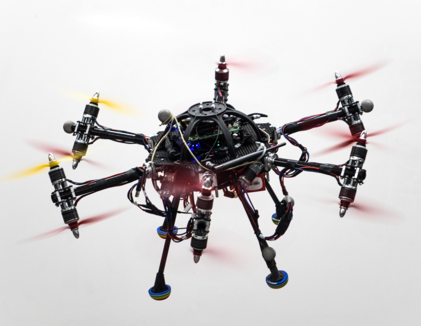


Figure 6.2: The omnidirectional flying vehicle (Omav) used to experimentally validate our method.

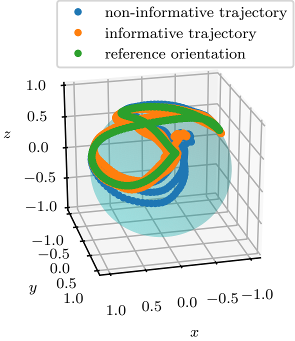


Figure 6.3: Tracking of a body-fixed unit vector $(1,1,1)/\sqrt{3}$ is plotted on a unit sphere.

40, 60, 80 data points from the experiments running each trajectory and built a model for each of these combinations by augmenting the prior model with these data points. The hyperparameters of the Gaussian processes are reoptimized. The models are then used in the controller to track the task trajectory in real experiments for validation. Tracking performance are evaluated in Fig. ?? as the average of the absolute angular acceleration over all three axes. It can be noted that for the same amount of data points, the informative trajectory always outperforms the non-informative trajectory in term of both mean tracking error and corresponding variance. On average the performance[5] of informative trajectories outperforms the non-informative one by 13.3%.

5.3 Comparison of the model generalizability

To test the generalizability of the model learned from the informative trajectory, a modified figure-8 trajectory with higher pitch and roll reference angles (up to 43 degrees) is used. As can be seen in the phase plot in Fig. 6.4, the state input pairs of the modified figure-8 extend up to twice of the original one. In this case, both models from the informative trajectory and the non-informative trajectory have 100 data points. It can be seen from Table 6.1 that the model learned from informative trajectory yields better tracking performance, especially around the z-axis.

---

[5] By performance of a trajectory we mean the tracking performance using the updated controller with the data collected from that trajectory.

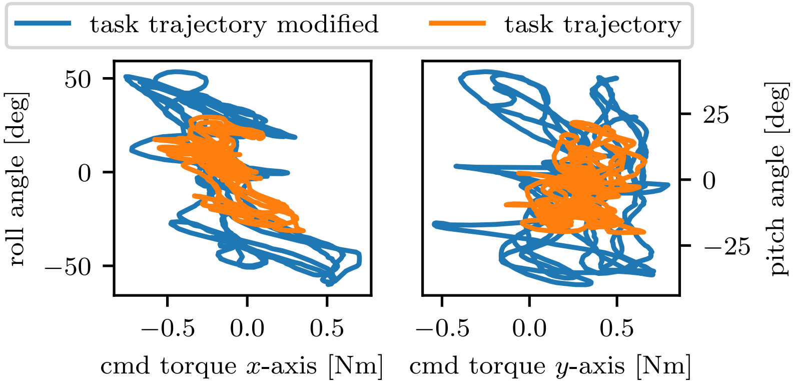


Figure 6.4: Phase plots of the task trajectory and modified task trajectory. It can be observed that although the modified trajectory extend beyond the task trajectory, the model learned from the informative trajectory helps to reduce the tracking.

	x-axis	y-axis	z-axis
non-informative	38.4%	41.7%	23%
informative	43.2%	57.9 %	62%

Table 6.1: Angular acceleration tracking error reduction with respect to the case without model learning in percentage.

6 Conclusion

This work presents a practical framework that effectively and efficiently collects data points for the learning of models used at the control level to significantly improve tracking performance on real robots. We experimentally demonstrate the validity of the method on an overactuated aerial robot, the Omav, whose dynamics is complex and difficult to learn. Experimental results show that the learned model from informative trajectories is efficient in data points collection and generalizes on modified trajectories.

Part C

AERIAL PHYSICAL INTERACTION LEARNING CONTROL

Paper IV

Learning Variable Impedance Control for Aerial Sliding on Uneven Heterogeneous Surfaces by Proprioceptive and Tactile Sensing

Weixuan Zhang, Lionel Ott, Marco Tognon, and Roland Siegwart

Abstract

The recent development of novel aerial vehicles capable of physically interacting with the environment leads to new applications such as contact-based inspection. These tasks require the robotic system to exchange forces with partially-known environments, which may contain uncertainties including unknown spatially-varying friction properties and discontinuous variations of the surface geometry. Finding a solution that senses, adapts, and remains robust against these environmental uncertainties remains an open challenge. This paper presents a learning-based adaptive control strategy for aerial sliding tasks. In particular, the gains of a standard impedance controller are adjusted in real-time by a neural network policy based on proprioceptive and tactile sensing. This policy is trained in simulation with simplified actuator dynamics in a student-teacher learning setup. The real-world performance of the proposed approach is verified using a tilt-arm omnidirectional flying vehicle. The proposed controller structure combines data-driven and model-based control methods, enabling our approach to successfully transfer directly and without adaptation from simulation to the real platform. We attribute the success of the sim-to-real transfer to the inclusion of feedback control in the training and deployment. We achieved tracking performance and disturbance rejection that cannot be achieved using fine-tuned state of the art interaction control method.

Published in:
IEEE Robotics and Automation Letters, 2022
© Open access article distributed under the Creative Commons Attribution License.

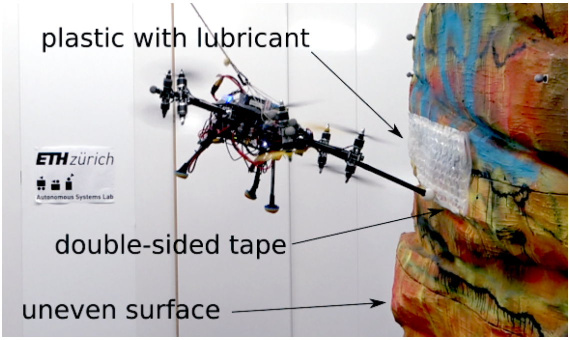


Figure 7.1: A tilt-arm omnidirectional flying vehicle is sliding along surfaces with different friction properties and geometries. The end effector is in contact with a double-sided tape while sliding on a rock like papier-mâché.

1 Introduction

Aerial interaction tasks such as contact-based inspections [29]–[31] require the flying vehicle to slide a sensor along the surface and maintain contact. These surfaces vary in spatial geometry and friction properties. This poses two challenges: Firstly, the surface friction property cannot be directly measured and may change discontinuously if the surface consists of different materials (i.e., heterogeneous surface). In addition, the perception of the surface geometry tends to be impaired by sensor noises, occlusions, and low spatial resolution, especially when the geometry is discontinuous (e.g. steps or holes). Secondly, from the control perspective, the presence of these unknown environment features introduces uncertainties in contact forces. The induced interaction wrench disturbance may cause large abrupt changes in the robot dynamics, which easily destabilizes the system. Existing approaches show interactions with simple continuous surfaces (e.g., planes, cylinders, etc), where the geometry is assumed to be known and friction properties are assumed to be identical everywhere (i.e., homogeneous) [32]–[34]. These approaches also require expert knowledge to perform manual tuning for each environment to achieve optimal performance. How to design a solution that senses, adapts and remains robust to the environmental uncertainties remains an open challenge.

This work presents a method to overcome these challenges. More specifically, a variable impedance controller that uses proprioceptive and tactile sensing to adjust controller gains according to environmental changes.

There exists literature on sliding on partially-known and uneven surfaces in the manipulation community, using passivity [35] or adaptive force control [36], [131]. They do not directly extend to aerial robots, as force control introduces safety challenges when aerial robots transition from contact flight to free flight. One could also employ disturbance observer-based robust control [38]. Such an approach has the disadvantage of being slow to react, especially in the presence of noisy measurements and inaccurate process models. Consequently, the flying vehicle will struggle to handle abrupt changes in the environment. Finally, another option is to use mechanical compliance at the flying machine's end effector [39]. However, this increases the mechanical complexity and cost while further decreasing the system's payload.

An alternative approach to address the aforementioned sensing challenge is to use proprioceptive measurements and tactile sensing to react to changes in the geometry and friction properties of the environment. This type of approach can be found in quadrupedal robots like the MIT Cheetah [40] and Anymal [41]. These robots use either control signals to infer a leg touchdown event or use IMU signals, robot states, and control commands to implicitly infer the surface properties. Compared to visual sensing, these two sensing modalities are more direct in measuring the contact and are more suitable in sensing the discontinuous features of the environment. However, they were never exploited in the field of aerial manipulation to face unknown environments.

From a control perspective, impedance control [132] introduces algorithmic compliance and has shown success for aerial sliding tasks on homogeneous surfaces [34]. Selection of the impedance gains is a trade-off between controller tracking performance and system compliance. Depending on the task and environment, different impedance parameters may perform optimally [133]. A variable impedance controller [44], [134] is thus an attractive solution and has shown promising results in aerial manipulation [135] and [32]. To sense and adapt to the environmental uncertainties during aerial sliding tasks, one can vary the impedance parameters using the previously mentioned two sensing modalities. This nonlinear mapping may also be learned using deep reinforcement learning (RL) techniques [136], which have become a popular tool to generate highly nonlinear and effective control policies using neural networks. Examples combining reinforcement learning with variable impedance control can be found in legged robots [50] and manipulator [47]–[49], [137]. For flying vehicles, learning from simulation instead of real-world is the preferred approach, since a failure typically leads to a crash. In particular, student teacher setups [42] are prosmising to improve learning efficiency, since a teacher policy can use privileged information from simulation to guide the student policy.

Considering the problem of aerial physical interaction with heterogeneous and uneven surfaces, taking inspiration from different robotic domains, this paper presents a novel control method that combines the benefits of proprioceptive and tactile sensing, variable impedance control and reinforcement learning. A neural network policy is learned to adapt the stiffness gains of a standard impedance controller according to proprioceptive and tactile sensing. The intuition is that these two sensing modalities jointly capture the interaction environmental properties and the learned policy can therefore adjust the controller to be robust against these disturbances. The training of this policy is conducted entirely in simulation in a student-teacher setup. This solution allows for a direct transfer of the learned policy from a simplified closed-loop simulation to an omnidirectional aerial vehicle (Omav as shown in Fig. 7.1 and described in [24]), significantly improving its robustness and control performance during interaction and outperforming fine-tuned impedance controller.

Our contributions are as follows:

- A learning-based solution for aerial sliding tasks that senses, adapts, and remains robust against challenging interaction environment uncertainties in surface geometry and friction properties.
- An approach to address sim-to-real transfer by including a closed-loop controller to sup-

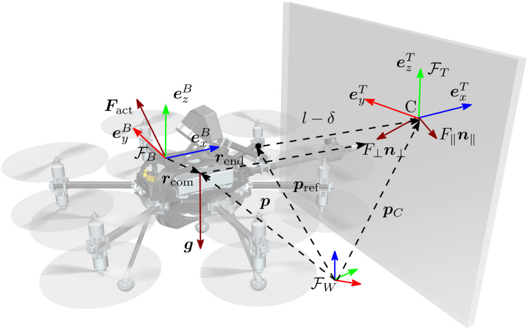


Figure 7.2: Omav in interaction, showing some of the symbols and quantities required to model the system.

press model uncertainty, which allows for learning from simplified actuator dynamics.

The above contributions have been validated experimentally using an Omav with a rigid single-body end effector for the task of sliding along surfaces with different friction properties and uneven geometry.

2 Preliminaries

In this section we provide a brief overview of the models used to represent robot dynamics and environment interactions before describing the basics of impedance control.

2.1 Robot dynamics

It is assumed that the flying vehicle has a single-body end effector rigidly attached to its body (see Fig. 7.2). The robot is modeled as a single rigid body and its dynamics are expressed using Newton-Euler method in free flight and interaction are given by the following equation

$$\boldsymbol{M}\dot{\tilde{\boldsymbol{v}}} + \boldsymbol{C}\tilde{\boldsymbol{v}} + \boldsymbol{g} = \boldsymbol{w}_{\text{act}} + \boldsymbol{w}_{\text{dist}}, \tag{7.1}$$

where $\boldsymbol{M} \in \mathbb{R}^{6\times 6}$ is the symmetric positive definite inertia matrix and $\boldsymbol{C} \in \mathbb{R}^{6\times 6}$ contains the centrifugal and Coriolis terms, and $\boldsymbol{g} \in \mathbb{R}^6$ is the gravity. The generalized velocity $\tilde{\boldsymbol{v}} \in \mathbb{R}^6$ represents the center of mass velocity and body rates of the system. The generalized acceleration vector are denoted as $\dot{\tilde{\boldsymbol{v}}}$. The terms $\boldsymbol{w}_{\text{act}}$ and $\boldsymbol{w}_{\text{dist}} \in \mathbb{R}^6$ are both stacked force and torque vectors acting on the system generated by rotor actuation and disturbance sources (e.g., contact or wind disturbances), respectively.

2.2 Interaction with the environment

When the robot is sliding along the surface with its end effector, the force disturbance $\boldsymbol{f}_{\text{dist}}$ has three sources: 1. environmental aerodynamic effects, e.g. ground effects, wall effects, and wind gusts, 2. actuation modeling errors, and 3. the contact force $\boldsymbol{f}_{\text{con}}$. Both 1) and 2) are assumed to be negligible as they are at least an order of magnitude smaller than that of the contact force [54], [138]. The treatment of wind gust disturbance is referred to future work.

During interaction flights, the robot is assumed to have a single contact point with the uneven surface at C (Fig. 7.2). A local contact frame $\mathcal{F}_T$ is attached to the contact point such that its x-axis is normal to the tangent plane at C. The contact force acting on the end effector expressed in the body frame $\mathcal{F}_B$ is modeled as follows:

$$\boldsymbol{f}^B_{\text{dist}} = \boldsymbol{f}^B_{\text{con}} = \boldsymbol{R}^{BT}(F_\perp \boldsymbol{n}^T_\perp + F_{||}\boldsymbol{n}^T_{||}), \tag{7.2}$$

where $F_\perp$ is the scalar normal force and $F_{||}$ is the scalar friction force. The coordinate transformation matrix $\boldsymbol{R}^{BT}$ transforms a vector from $\mathcal{F}_T$ to $\mathcal{F}_B$. The unit normal vector $\boldsymbol{n}^T_\perp$ is perpendicular to the tangent plane at C and $\boldsymbol{n}^T_{||}$ is the sliding force direction parallel to the tangent plane at C. The relative orientation $\boldsymbol{R}^{BT}$ is assumed to be partially known due to imperfect map. In addition, the end effector and the contact force creates a torque around the vehicle's center of mass with the lever arm length denoted as l.

When the end effector is sliding with nonzero velocity on the surface, a Coulomb friction model is assumed, i.e.:

$$F_{||} = \mu(\boldsymbol{p}_C)F_\perp, \tag{7.3}$$

where $\mu(\boldsymbol{p}_C)$ is the friction coefficient that can vary spatially across the surface, depending the position of contact $\boldsymbol{p}_C$.

2.3 Impedance controller with constant gains

An impedance controller with constant control gains is a common approach used for aerial sliding tasks [24] and used in this paper as the baseline approach.

Given a desired sliding path on the surface, a reference pose trajectory is designed based on the given surface map . This reference trajectory consists of a desired center of mass position which results in a end effector position that is always behind the sliding surface by a constant distance $\delta \in \mathbb{R}$ (Fig. 7.2), also denoted as the penetration level. For the attitude part of the trajectory, the vector along the tool arm should align with the contact frame x-axis.

Given this desired reference trajectory, an impedance controller with constant control gains has the following form:

$$\begin{aligned}\boldsymbol{w}_{\text{act}} &= \boldsymbol{C}\tilde{\boldsymbol{v}} + \boldsymbol{g} + (\boldsymbol{M}\boldsymbol{M}^{-1}_{\text{des}} - \mathbb{I}_6)\boldsymbol{w}_{\text{dist}} \\ &- \boldsymbol{M}\boldsymbol{M}^{-1}_{\text{des}}(-\boldsymbol{M}\dot{\tilde{\boldsymbol{v}}}_{\text{ref}} + \boldsymbol{D}_{\text{des}}\tilde{\boldsymbol{e}}_v + \boldsymbol{K}_{\text{des}}\tilde{\boldsymbol{e}}_s),\end{aligned} \tag{7.4}$$

with $\tilde{\boldsymbol{e}}_s \in \mathbb{R}^6$, containing the position and attitude tracking error as shown in [24].

$\boldsymbol{M}_{\text{des}}, \boldsymbol{D}_{\text{des}}, \boldsymbol{K}_{\text{des}} \in \mathbb{R}^{6\times 6}$ are the desired inertia, damping, and stiffness matrices, respectively.

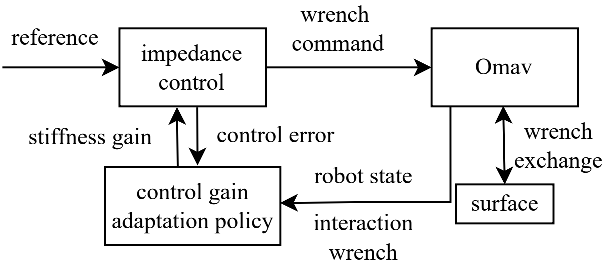


Figure 7.3: Variable impedance learning controller. It augments the control strategy depicted in [24] by adding a control gain adaptation policy, which takes as input the state, control error and the wrench measurements from the wrench sensor and map them to impedance gain.

Plugging (7.4) into (7.1) results in

$$\boldsymbol{M}_{\text{des}}\dot{\tilde{\boldsymbol{e}}}_v + \boldsymbol{D}_{\text{des}}\tilde{\boldsymbol{e}}_v + \boldsymbol{K}_{\text{des}}\tilde{\boldsymbol{e}}_s = \boldsymbol{w}_{\text{dist}}. \tag{7.5}$$

This implies that an impedance controller shapes the closed-loop system as a second-order system. Note that there will be inevitable pose error due to the contact wrench acting on the flying vehicle. The wrench command $\boldsymbol{w}_{\text{act}}$ is then allocated through a chosen mapping and a saturation function to individual actuator commands (for more details see [24]).

3 Methodology

3.1 Problem statement

The goal is to enable a flying vehicle to accurately follow a trajectory planned based on an imperfect map while remaining stable and staying in contact with an uneven surface which has unknown discontinuities in geometry and unknown friction properties $\mu(\boldsymbol{p}_C)$. Given a task-space reference trajectory, we assume the robot has access to contact wrench measurements $\boldsymbol{w}_{\text{meas}}$ via a force torque sensor, and it is controlled by an impedance controller with a gain-adjusting policy π_θ, which is parametrized by θ. To achieve the above goal, we propose a strategy to find a deterministic policy π_θ that adjusts the controller's gains to fulfill the following criteria: 1. minimize the tracking error $||\tilde{\boldsymbol{e}}_v||^2 + ||\tilde{\boldsymbol{e}}_s||^2$ where $||\cdot||$ denotes the Euclidean norm,; 2. ensure $\boldsymbol{f}^B_{\text{con}} > 0$; 3. ensure the platform stability.

3.2 Variable impedance learning controller

The proposed approach adds a control gain adaptation policy to the standard impedance controller (7.4). The policy adapts the impedance controller gains based on the proprioceptive measurements and the tactile feedback via the adaptive unit described in Fig. 7.3. In particular, the desired stiffness $\boldsymbol{K}_{\text{des}}(\boldsymbol{z}) \in \mathbb{R}^{6\times 6}$ is a function of the m-dimensional measurements $\boldsymbol{z} \in \mathbb{R}^m$:

$$\boldsymbol{K}_{\text{des}}(\boldsymbol{z}) = \boldsymbol{K}_{\min} + (\boldsymbol{K}_{\max} - \boldsymbol{K}_{\min})\,\text{diag}\{\pi(\boldsymbol{z})\} \tag{7.6}$$

where m is the dimension of the measurements, $\boldsymbol{K}_{\min}$ and $\boldsymbol{K}_{\max}$ are diagonal positive semidefinite matrices, and $\pi : \mathbb{R}^m \to \mathbb{R}^6$ where $\mathbf{0} < \pi(\cdot) < \mathbf{1}$ with vector $\mathbf{0}$ and $\mathbf{1}$ of dimension 6. These constraints on the mapped value make sure the adaptive gains have lower and upper bounds. The lower bound $\boldsymbol{K}_{\min}$ ensures a minimum tracking performance while the upper bound $\boldsymbol{K}_{\max}$ prevents the system from instabilities caused by actuator saturation and system delay. These limits are both derived empirically through experiments. For brevity, in the following we use π instead of π_θ.

To obtain a damped second-order system, we impose a fixed relationship between $\boldsymbol{D}_{\text{des}}$ and $\boldsymbol{K}_{\text{des}}(\boldsymbol{z})$. The desired damping $\boldsymbol{D}_{\text{des}}$ is varied with the square root of the diagonal components of $\boldsymbol{K}_{\text{des}}(\boldsymbol{z})$:

$$\boldsymbol{D}_{\text{des}} = 2\zeta\sqrt{\boldsymbol{K}_{\text{des}}(\boldsymbol{z})}, \tag{7.7}$$

where ζ is a damping ratio. While we assume that the desired stiffness and damping can be well tracked, the desired inertia is in practice challenging to track as it requires an accurate actuation control [34]. We therefore set $\boldsymbol{M}_{\text{des}}$ equal to $\boldsymbol{M}$ and the adaptation of $\boldsymbol{M}_{\text{des}}$ is deferred to future works.

With $\boldsymbol{M} = \boldsymbol{M}_{\text{des}}$ and (7.7) inserted into (7.4), the following adaptive controller command can be obtained,

$$\boldsymbol{w}_{\text{act}} = \boldsymbol{M}\dot{\boldsymbol{v}}_{\text{ref}} - 2\zeta\sqrt{\boldsymbol{K}_{\text{des}}(\boldsymbol{z})}\tilde{\boldsymbol{e}}_v - \boldsymbol{K}_{\text{des}}(\boldsymbol{z})\tilde{\boldsymbol{e}}_s + \boldsymbol{C}\tilde{\boldsymbol{v}} + \boldsymbol{g}. \tag{7.8}$$

With (7.8) plugged into (7.1), the closed-loop error dynamics are shaped as a second-order system,

$$\boldsymbol{M}_{\text{des}}\dot{\tilde{\boldsymbol{e}}}_v + 2\zeta\sqrt{\boldsymbol{K}_{\text{des}}(\boldsymbol{z})}\tilde{\boldsymbol{e}}_v + \boldsymbol{K}_{\text{des}}(\boldsymbol{z})\tilde{\boldsymbol{e}}_s = \boldsymbol{w}_{\text{dist}}. \tag{7.9}$$

Note that changing the stiffness $\boldsymbol{K}_{\text{des}}(\boldsymbol{z})$ affects the interaction wrench $\boldsymbol{w}_{\text{dist}}$. To see this, consider (7.9) at steady state $\dot{\tilde{\boldsymbol{e}}}_v = \tilde{\boldsymbol{e}}_v = \mathbf{0}$ and projected along the surface normal direction. We obtain

$$k_\perp(\boldsymbol{z})\delta = F_\perp, \tag{7.10}$$

where $k_\perp(\boldsymbol{z})$ is the position stiffness gain in the surface normal direction. Assuming the end effector is in contact with the surface, which results in a constant position tracking error δ, $F_\perp$ is thus proportional to $k_\perp(\boldsymbol{z})$. This is the intuition of why we adapt the stiffness gain to counteract surface disturbances.

3.3 Reinforcement learning of gain adaptation policy

We assume that the mapping π can be modeled as a discrete-time continuous Markov decision process (MDP). An MDP is defined by a state space $\mathcal{S}$, an action space $\mathcal{A}$, a scalar reward function $\mathcal{R}$, and the transition probability P that dictates the stochastic system dynamics. A learning agent selects an action $\boldsymbol{a}$ from its policy π and receives a reward r. The objective of the RL framework is to find an optimal policy π^* that maximizes the discounted sum of rewards over an infinite time horizon:

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\tau(\pi)} \sum_{t=0}^{\infty} \gamma^t r[t] \tag{7.11}$$

where $\gamma \in (0, 1)$ is the discount factor, and $\tau(\pi)$ is the trajectory distribution under policy π, with t denoting the discretized time indices. The reward $r[t]$ at k is

$$\begin{aligned} r[t] = &-l_{\boldsymbol{e}_R} \left\| \boldsymbol{e}_R[t] \right\|^2 - l_p \left\| \boldsymbol{e}_p[t] \right\|^2 - l_d \left\| d[t] \right\|^2 \\ &- l_{\boldsymbol{\omega}} \left\| \boldsymbol{\omega}[t] \right\|^2 - l_{\boldsymbol{a}} \left\| \frac{\boldsymbol{a}[t]}{\|\boldsymbol{a}[t]\|} - \frac{\boldsymbol{a}[t-1]}{\|\boldsymbol{a}[t-1]\|} \right\|^2 \end{aligned} \tag{7.12}$$

$l_\star$ with subscripts $\star$ denotes the corresponding weight chosen such that all individual reward terms are at the same order of magnitude. $\boldsymbol{e}_R[t]$ and $\boldsymbol{e}_p[t]$ denotes the attitude and translational tracking error. $d[t] = (\boldsymbol{p}^T + \boldsymbol{r}_{\text{end}}^T - \boldsymbol{p}_C^T) \cdot \boldsymbol{e}_x^T$ denotes the scalar distance along the x-axis of the local contact frame between the end effector and the surface and $\cdot$ denotes the dot product between two vectors. The purpose of this term is to make sure that the policy keeps the end effector in contact with the surface. $\boldsymbol{\omega}[t]$ denotes the angular velocity. We penalize large angular velocities to avoid instabilities. The action $\boldsymbol{a}[t]$ is the output of the policy $\pi(\boldsymbol{z})$ at time t. The associated term is to make sure that the control inputs are smooth. The loss components are designed to reduce the tracking error while keeping contact with the surface without causing discontinuities in the actions and thus the actuator commands. These terms are in line with the problem statement in Sec. 3.1. Note that since the simulation of each individual actuator dynamics is omitted. The energy consumption of the flying vehicle can only be indirectly inferred and is therefore not included in the reward function.

The policy π is a fully connected neural network with three hidden layers of 32 units, its activation functions being leaky ReLu, and its last layer being a Sigmoid layer which guarantees to map to a bounded interval. For training we use the off-the-shelf RL algorithm proximal policy optimization (PPO) [139], a policy gradient algorithm that has been demonstrated to work for variable impedance control in contact tasks with a manipulator [140].

3.4 Simulation using simplified dynamics

To allow for efficient evaluation and training of the policy π, a simplified dynamics simulation is used. The flying vehicle is simulated as a single rigid body and the simulation of the individual actuator dynamics are approximated collectively as a single process. A saturation function on the wrench command is implemented, the output of which is delayed and set as external force and torque directly acting on the robot. Both the saturation threshold and the system delay are a conservative estimate of the empirically obtained actuation limits. This ensures that the actuator limits are well respected and the closed-loop system behaves like a delayed second-order system as designed. The inertia and mass are obtained from CAD. Although the actuator dynamics are simplified, special attention is paid to identify the correct center of mass position and the relative position of the end effector in the body frame ($\boldsymbol{r}_{\text{com}}$ and $\boldsymbol{r}_{\text{end}}$ in Fig. 7.2). They together determine the induced torque disturbance from a given contact force, which is

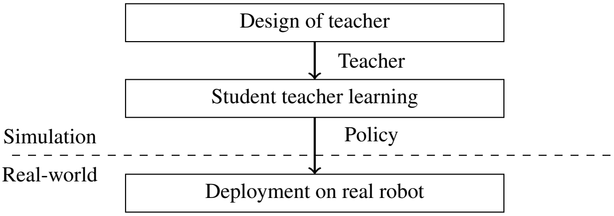


Figure 7.4: This block diagram shows the main steps in the student-teacher learning of the control gain adaptation policy.

essential for the simulation to learn the correct disturbance rejection strategy.

To simulate the interaction environment, surfaces that have different friction coefficients[1] are generated and concatenated together. Therefore, when the robot's end effector slides across the border between two surfaces with different friction properties, it experiences discontinuous changes in interaction forces. Furthermore, each surface can have a different height that leads to an uneven surface as a whole.

3.5 Learning from simulaiton

To efficiently learn an optimal policy that determines the adaptive stiffness $\boldsymbol{K}_{\mathrm{des}}(\boldsymbol{z})$ (see (7.6)), a student teacher learning approach [42] is deployed.

Fig. 7.4 provides an overview of this approach: Firstly, we design a teacher with access to privileged (ground-truth) information to dynamically select the desired stiffness in the variable impedance controller. Then a policy is learned to emulate the teacher and may be further improved using RL. The policy can be directly transferred to real-world without any additional sim-to-real adaptation..

The intuition behind the student teacher learning is that the teacher has access to the privileged information is much easier to design or train in an RL setting. We can also embed empirical tuning experience or other adaptive variable impedance strategies into the teacher. This is helpful for challenging tasks, as we find out empirically a direct reinforcement learning always lead to instabilities of the flying vehicle and prevents successful learning.

Teacher design

The teacher π_{t} serves as a guidance policy for the student policy. It makes use of the privileged information $\boldsymbol{z}_{\mathrm{gt}}$. Compared to $\boldsymbol{z}$, it contains additional information from the simulation, to which the student policy does not have access in real deployment.

In this work, we employ either a simple handcrafted policy π_{t} or a neural network learned from simulation using reinforcement learning as the teacher. Upon a rough surface, the hand-

---

[1] The same material was used for the end effector throughout this paper. Thus, for the sake of brevity, we only talk about surface friction coefficients when it would be more accurate to talk about friction pairs between the end effector and the surface.

crafted policy decreases the translational stiffness gain to reduce the normal force $F_{\perp}$ (see (7.10)) and therefore the torque disturbance, while the angular stiffness gain is also increased to better reject the torque disturbance. For reinforcement learning of the teacher, the teacher is learned from scratch using the privelged information, including friction coefficient at contact point and the surface normal vector in the vicinity of the contact.

Student learning

The control gain adaptation policy $\pi^*(\boldsymbol{z})$ is bootstrapped via supervised learning with the following loss function

$$\pi^* = \arg\min_{\pi} \|\pi_{\mathrm{t}}(\boldsymbol{z}_{\mathrm{gt}}) - \pi(\boldsymbol{z}))\|^2 . \tag{7.13}$$

where the feature $\boldsymbol{z}$ contains control signal, state estimate, IMU signals and interaction wrench measurements, with the first three signals being proppriocеptive sensing and the last one being tactile sensing. They indirectly provide the information about the interaction between the environment and the robot.

Training data is collected by rolling out the simulation using the teacher. For each rollout, the robot first approaches the surface, gets into contact and starts sliding following the desired trajectory. The policy $\pi^*(\boldsymbol{z})$ can be further refined using RL.

Data processing

Noisy observations (especially the interaction wrench measurements) are first-order low-pass filtered before they are input to the student policy. To have a smooth change in the stiffness gain $\boldsymbol{K}_{\mathrm{des}}(\boldsymbol{z})$, the output of the student policy is also low-pass filtered before they are used to compute $\boldsymbol{K}_{\mathrm{des}}(\boldsymbol{z})$.

3.6 Remark

Stability We remark that the stability using RL may be ensured using the concept of passivity [141], or Lyapunov methods [142]. This is left as future work. Instead we discuss here practical measures to face possible causes of the instability. Those are mainly twofold: actuator saturation due to the rapid changes in the control gains or high gains and low gains which is incapable of stabilizing this open-loop unstable system. Thus we implemented the following strategies: 1. Lower and upper bounds on the stiffness gain as shown in (7.6); 2. Slew rate limit on the gains are empirically determined and only applied in deployment as precaution ; 3. The output of the policy $\pi_{\mathrm{s}}^*(\boldsymbol{z})$ is filtered for a smooth control signal in training and deployment.

Sim-to-real transfer The sim-to-real gap, i.e., the mismatch between simulation and reality, is a challenging problem when learning from simulation, and can limit the transferability of the learned policy. Frequent causes of sim-to-real gaps are inaccurate modeling of the actuator dynamics and delays in the system [27]. This is especially a problem for end-to-end learning approaches. Without feedback controller in the loop the learning procedure relies on the accuracy of the open-loop dynamics simulation and the sim-to-real gap can diverge exponentially. However, given a well-designed feedback controller (e.g., the one one presented in Fig. 7.3), which shapes the system to a desired second-order system (7.9) and suppresses model uncertainty, the gap between the reality and the simulation is kept small. This is particularly advantageous for a complicated system like the Omav, where a large amount of training data is required for an accurate model learning of the whole body dynamics (18 actuators and 6 degrees of freedom). As a comparison, a million samples are required for the modeling of a single one degree of freedom actuator of quadrapedal walking robot [27]. What is worse, if the Omav crashes or if its configuration changes, training data needs to be recollected again for an accurate model learning. Our approach does not require such a meticulous effort.

4 Experiments

4.1 Experimental setup

The experimental set-up and the platform are shown in Fig. 7.1 and Fig. 7.6. The experiments are carried out at the indoor aerial robotic testbed of the Autonomous System Lab, ETH Zurich. The Omav weighs $4.5\,\mathrm{kg}$ and is equipped with an NUC i7 computer and a PixHawk flight controller. For a more complete description of this platform see [24]. A pole end effector is rigidly attached to the vehicle and points outwards for a sufficient margin between the propellers and the surface. A ping pong ball is attached at the pole tip to ensure a single contact during sliding movements. A force torque sensor from Rokubi, mounted to the end effector, measures the interaction wrench. There are three interaction environments for real experiments: sand paper on a flat white board to investigate interaction with heterogeneous surface, a step of $2\,\mathrm{cm}$ on a white board (Fig. 7.6) to investigate interaction with discontinuous surface geometry, and finally, a rock-like structure (Fig. 7.1) which combines both traits to form a challenging environment typically seen in real applications. A motion capture system provides pose estimates for both the robot and the whiteboard/rock at $100\,\mathrm{Hz}$. The robot is unaware of their surface properties and the local unevenness on the surface. The task trajectory is to follow a straight line trajectory parallel to the gravity vector with zero pitch and roll while sliding on the vertical surface. The penetration level δ is set to $0.07\,\mathrm{m}$, which leaves sufficient margin to ensure contact under the state estimate uncertainty.

For the simulation we use RaiSim [143], a cross-platform multi-body physics engine for robotics. During training in the simulation, for each rollout, the robot approaches the surface and starts sliding along the surface for 15 seconds, which emulates a task trajectory from the real-world experiments. During each rollout, the Omav reaches a speed of $0.2\,\mathrm{m\,s^{-1}}$ and slides across surfaces with different friction coefficients. For each interaction environment, we set up a different interaction environment in the simulation and learn a different policy to treat each problem separately. Find a single policy that tackles different environments can be studied in the context of continual learning and is planned for future work.

To train the teacher or student policy using RL, a hundred simulation instances are spawned with slightly perturbed vehicle and environment properties. In each epoch, these instances are simultaneously simulated to obtain a batch of rewards for stochastic gradient policy optimization. The policy was trained on a single NVIDIA 3060Ti GPU, which takes from 2 to 8 hours depending on the interacting environment.

Since there is only a straight line to follow, the dimension of the measurement vector $\boldsymbol{z}$ is reduced to six: filtered pitch velocity, pitch error, position control error in the sliding direction, linear velocity, friction force filtered, normal force filtered. The output to the action space are the linear gain in the surface normal direction and the angular gain in the pitch direction, i.e. the axis of torque disturbance. The rest of the controller gains is kept constant.

4.2 Sim-to-real transfer

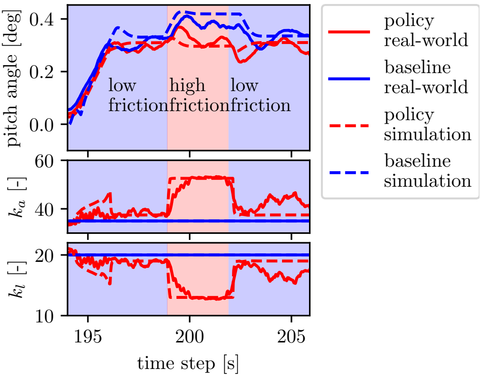


Figure 7.5: Comparison between baseline and our approach while the robot approaches and slides across two surfaces with different friction coefficients, both in simulation (dashed line) and real-world (solid line). k_a and k_l denote the angular stiffness gain and translation stiffness gain, respectively.

Fig. 7.5 demonstrates the policy transfer from simulation to reality. It compares four experiments completing the same sliding task with different controllers (baseline or our approach) and different set-ups (simulation or real experiment). The plots are aligned using the measured force impact when the Omav enters from free flight into contact with the whiteboard.

For evaluation, the Omav must slide across three concatenated surfaces: the first is the whiteboard with low friction, then a sand paper with high friction, and finally the whiteboard again. The friction coefficients are empirically estimated through force torque sensor measurements, with which three surfaces are generated in the simulation that replicate the experimental evaluation set-up. The approach in evaluation is a policy π^* defined in the (7.13) (without further refinement using RL) and a baseline impedance controller with constant control gains. The policy is trained in a simulation environment with six different surface friction coefficients (0.05, 0.15, 0.25, 0.45, 0.55, 0.62) that covers a range of friction coefficients. During collection of data using the teacher, each surface is randomly assigned one of the six friction coefficients to robustify the learned policy.

Fig. 7.5 demonstrates that the learned adaptive control strategy can be successfully transferred from simulation to the real-world. Given the same controller (either baseline or our approach), the simulation and the real-world experiments result in a close similarity of the respective robot state (the pitch angle) and controller gains (the angular stiffness gain k_a and translational stiffness gain k_l). The pitch angle is shown since it has the most obvious correlation with the surface friction coefficient given the baseline controller.

4.3 Sliding across a heterogeneous flat surface

Fig. 7.5 also showcases the performance of the regressed student policy on heterogenous surfaces. In this subsection, we only evaluate the real experimental data. While sliding on the surface, the contact force unavoidably leads to a tilted angle of the Omav. With the baseline controller, the vehicle tilts on average $4.9°$ after encountering a high friction surface (from $199\,\mathrm{s}$ to $202\,\mathrm{s}$), whereas with our approach, the tilt of the vehicle only increases to about $1.3°$ on average. This shows that the Omav is able to keep a almost constant tilt angle when sliding across different surfaces, thus improve the attitude tracking performance and reducing the chance of a crash at the transition of difference surfaces.

4.4 Sliding over a surface with discontinuous geometry

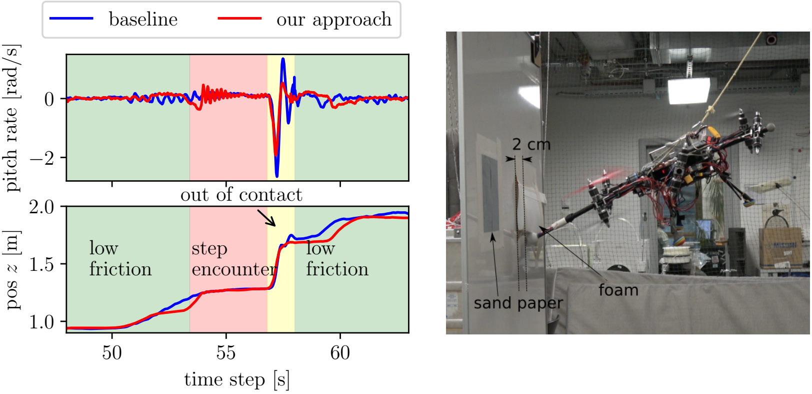


Figure 7.6: The end effector is blocked by a step of 2 cm made out of foam while sliding on a slippery whiteboard.

The experiment (Fig. 7.6) presented in this section aims to investigate the ability to reject the disturbance caused by surface discontinuity and remain stable (criteria 3 in Sec. 3.1). For the real experiment, a piece of foam is taped to the whiteboard and creates a step of about $2\,\mathrm{cm}$

along the sliding trajectory. During sliding, the step blocks the end effector, which leads to an increase of the pitch angle. The end effector then detachs from the surface and the sudden disappearance of the contact force presents as a large disturbance to the robot.

Our approach is trained as follows: Initially an even surface with randomly generated friction coefficients as described in Sec. 4.3 is used to bootstrap the policy. Then a piece of uneven surface with $1\,\mathrm{cm}$ steps is created by setting each neighbouring surface to have a height difference of $1\,\mathrm{cm}$. The policy is refined and trained for 900 epochs. The height difference is then changed to $2\,\mathrm{cm}$ with another 700 epochs of training. Training was terminated early when the reward stopped increasing for 300 epochs.

The behaviour exhibited by our approach when a step is encountered is to adapt the control gain to be more compliant. Compared to the baseline approach,this leads to less oscillations in pitch velocity when the Omav's end effector slips and is out of contact (the yellow region in Fig. 7.6).

4.5 Sliding across a challenging surface

A challenging rock-like papier-mâché surface (Fig. 7.1) is set up to compare our approach with baseline controllers. The surface of this rock is uneven and heterogeneous. Double-sided tapes (high friction) and plastic surface with lubricant (low friction) are added to the surface to emulate heterogeneous surface in extreme situations. Two trajectories (trajectory red and blue as the color in Fig. 7.7 indicates) were tested for more variety. For the training environment, we use procedurally generated terrain maps that have similar surface variations to the rock. During rollout, the fricition coefficients are randomly selected from six friction coefficients (0.1, 0.15, 0.45, 0.6, 0.75, 0.9) every four seconds. Note that the evaluation and training environments are distinct and demonstrated the generalizability of our approach.

As shown in Fig. 7.7, our approach consistently outperforms the baseline controller with nine combinations of low, middle and high angular and translational gain. These different combinations represent a tuning process that is often practiced in real-world. Each data point represents the average tracking performance over the sliding of the same trajectory for the three times with the same controller. Several interesting observations can be made: firstly, among the baseline approaches, a good tracking performance is generally achieved by high stiffness controller but the converse is not true. It is observed in the experiment that a high gain controller ($k_l = 20$ and $k_a = 100$ for trajectory blue) can lead to instability. However, we never experienced instability issues with our policy throughout the experiments. This indicates that our approach adapts the controller to be more compliant when necessary. Secondly, for each trajectory, the optimal set of constant gains are different. For example, to achieve best tracking performance in pitch, trajectory red and trajectory blue have different optimal gains (upper left in the plot). This means that in reality, the engineer have to tune the parameters for each specific trajectory and surface to gain optimal performance. However, our approach always performs well for both trajectories. The tracking data points of our approach are both located in the lower left of the plot, which means good tracking performance in both position and orientation. This implies that our policy adapts to the different surface unevenness and varying friction coefficients.

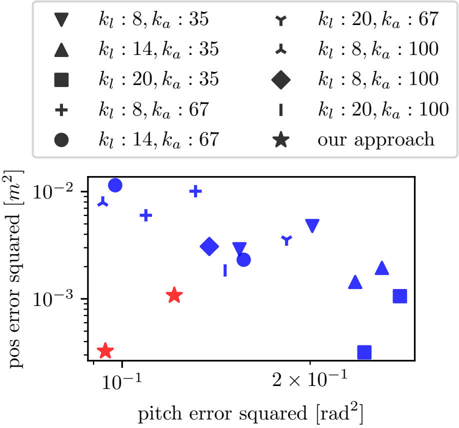


Figure 7.7: Comparison of tracking performance of the learned policy with different combinations of constant gains. The learned policy outperforms all the constant gain performance. Note that it is plotted in logarithm scale for a better visualization of data. Two different colors are used to denote different trajectories. k_a and k_l denote the angular stiffness gain and translation stiffness gain, respectively.

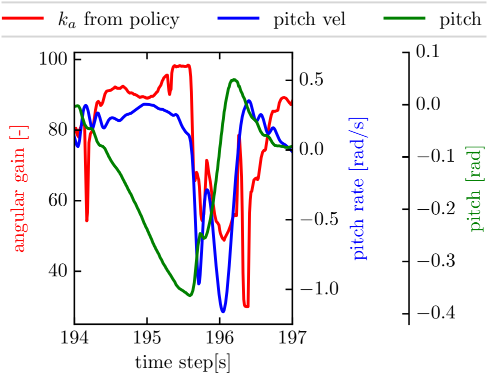


Figure 7.8: A time history plot of partial inputs and outputs of the policy when sliding on the rock-like surface

An time history plot of partial inputs and outputs of the policy in plot Fig. 7.8 provides an intuition on the adaptiveness of our policy. Note that when the pitch error increases from 194 s to 195.5 s, the angular gain k_a is also increased to maximum to reduce the tracking error. However, around 195.5, the end effector detaches from the surface and causes oscillations on pitch rate, the angular gain is decreased to be more compliant and the vehicle remains stable.

5 Conclusion

This paper presented an approach that senses, adapts and remains robust against disturbances caused by discontinuous surface variations in geometry and friction during aerial sliding tasks for fully actuated flying vehicles. When the environmental property changes, an adaptation policy adjusts the control gains of a standard impedance controller to reject these disturbances. Experimental results demonstrated that the policy learned in simulation can be directly transferred to the aerial vehicle without adaptation. The learned policy is able to slide on a challenging rock-like surface and outperform state-of-art interaction controllers.

Acknowledgement

The authors would like to thank Nicholas Lawrance for thoroughly going through the manuscript and Eugenio Cuniato for the fruitful discussions and help with experiments.

Bibliography

[1] *Bloomberg report*, (Date last accessed 16-July-2018). [Online]. Available: `https://www.bloomberg.com/press-releases/2021-02-24/global-commercial-drone-market-expected-to-exceed-8-5-billion-by-2027`.

[2] *Verity studios*, (Date last accessed 08-Dec-2021). [Online]. Available: `https://veritystudios.com/`.

[3] G. Jouvet, E. van Dongen, M. P. Lüthi, and A. Vieli, "In situ measurements of the ice flow motion at eqip sermia glacier using a remotely controlled unmanned aerial vehicle (uav)", *Geoscientific Instrumentation, Methods and Data Systems*, vol. 9, no. 1, pp. 1–10, 2020.

[4] A. Pretto, S. Aravecchia, W. Burgard, *et al.*, "Building an aerial–ground robotics system for precision farming: An adaptable solution", *IEEE Robotics & Automation Magazine*, vol. 28, no. 3, pp. 29–49, 2020.

[5] *Verity*, (Date last accessed 08-Dec-2021). [Online]. Available: `https://verity.net/`.

[6] R. W. Beard and T. W. McLain, *Small unmanned aircraft*. Princeton university press, 2012.

[7] R. Mahony, V. Kumar, and P. Corke, "Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor", *IEEE robotics & automation magazine*, 2012.

[8] A. Ollero, M. Tognon, A. Suarez, D. Lee, and A. Franchi, "Past, present, and future of aerial robotic manipulators", *IEEE Transactions on Robotics*, 2021.

[9] *Voliro airborne robotics*, (Date last accessed 08-Dec-2021).

[10] M. Hamandi, F. Usai, Q. Sablé, N. Staub, M. Tognon, and A. Franchi, "Design of multirotor aerial vehicles: A taxonomy based on input allocation", 2021.

[11] F. M. Callier and C. A. Desoer, *Linear system theory*. Springer Science & Business Media, 2012.

[12] R. Tedrake, Z. Jackowski, R. Cory, J. W. Roberts, and W. Hoburg, "Learning to fly like a bird", in *14th International symposium on robotics research. Lucerne, Switzerland*, Citeseer, 2009.

[13] M. Piccoli, "Passive stability and actuation of micro aerial vehicles", Ph.D. dissertation, University of Pennsylvania, 2016.

[14] E. R. Ulrich, D. J. Pines, and J. S. Humbert, “From falling to flying: The path to powered flight of a robotic samara nano air vehicle”, *Bioinspiration & biomimetics*, vol. 5, no. 4, p. 045 009, 2010.

[15] J. Houghton and W. Hoburg, “Fly-by-wire control of a monocopter”, *Massachusetts Institute of Technology, Project Report*, 2008.

[16] S. Jameson, K. Fregene, M. Chang, N. Allen, H. Youngren, and J. Scroggins, “Lockheed martin’s samarai nano air vehicle: Challenges, research, and realization”, in *50th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, 2012, p. 584.

[17] A. Kellas, “The guided samara: Design and development of a controllable single-bladed autorotating vehicle”, M.S. thesis, Massachusetts Institute of Technology, 2007.

[18] M. Orsag, J. Cesic, T. Haus, and S. Bogdan, “Spincopter wing design and flight control”, *Journal of intelligent & robotic systems*, vol. 70, no. 1-4, pp. 165–179, 2013.

[19] M. Bakula, C. Hockley, R. Khatri, C. Kirby, C. Sammet, and C. Reinholtz, “A natural evolution in flight: The design and development of the samareye system, a method for searching closed quarter environments”, in *First Symp. on Indoor Flight Issues*, 2009, pp. 1–12.

[20] M. Piccoli and M. Yim, “Piccolissimo: The smallest micro aerial vehicle”, in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, IEEE, 2017, pp. 3328–3333.

[21] Z. E. Teoh, S. B. Fuller, P. Chirarattananon, N. O. Pérez-Arancibia, J. D. Greenberg, and R. J. Wood, “A hovering flapping-wing microrobot with altitude control and passive upright stability.”, in *IROS*, 2012, pp. 3209–3216.

[22] F. Van Breugel, W. Regan, and H. Lipson, “From insects to machines”, *IEEE robotics & automation magazine*, vol. 15, no. 4, 2008.

[23] S. K. H. Win, L. S. T. Win, D. Sufiyan, and S. Foong, “Design and control of the first foldable single-actuator rotary wing micro aerial vehicle”, *Bioinspiration & biomimetics*, vol. 16, no. 6, p. 066 019, 2021.

[24] K. Bodie, M. Brunner, M. Pantic, *et al.*, “An omnidirectional aerial manipulation platform for contact-based inspection”, *arXiv preprint arXiv:1905.03502*, 2019.

[25] D. Hentzen, T. Stastny, R. Siegwart, and R. Brockers, “Disturbance estimation and rejection for high-precision multirotor position control”, *arXiv preprint arXiv:1908.03166*, 2019.

[26] B. Yüksel, C. Secchi, H. H. Bülthoff, and A. Franchi, “A nonlinear force observer for quadrotors and application to physical interactive tasks”, in *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, 2014.

[27] J. Hwangbo, J. Lee, A. Dosovitskiy, *et al.*, “Learning agile and dynamic motor skills for legged robots”, *Science Robotics*, 2019.

[28] L. Ljung, *System identification, Theory for the user*. Prentice Hall, 1999.

[29] M. Á. Trujillo, J. R. Martınez-de Dios, C. Martın, A. Viguria, and A. Ollero, “Novel aerial manipulator for accurate and robust industrial ndt contact inspection: A new tool for the oil and gas inspection industry”, *Sensors*, vol. 19, no. 6, p. 1305, 2019.

[30] M. Tognon, H. A. T. Chávez, E. Gasparin, *et al.*, “A truly-redundant aerial manipulator system with application to push-and-slide inspection in industrial plants”, *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1846–1851, 2019.

[31] R. Watson, M. Kamel, D. Zhang, *et al.*, “Dry coupled ultrasonic non-destructive evaluation using an over-actuated unmanned aerial vehicle”, *IEEE Transactions on Automation Science and Engineering*, 2021.

[32] K. Bodie, M. Brunner, M. Pantic, *et al.*, “Active interaction force control for contact-based inspection with a fully actuated aerial vehicle”, *IEEE Transactions on Robotics*, vol. 37, no. 3, pp. 709–722, 2020.

[33] D. Tzoumanikas, F. Graule, Q. Yan, D. Shah, M. Popovic, and S. Leutenegger, “Aerial manipulation using hybrid force and position nmpc applied to aerial writing”, *Robotics: Science and Systems*, 2020.

[34] M. Ryll, G. Muscio, F. Pierri, *et al.*, “6d interaction control with aerial robots: The flying end-effector paradigm”, *The International Journal of Robotics Research*, vol. 38, no. 9, pp. 1045–1062, 2019.

[35] E. Shahriari, L. Johannsmeier, E. Jensen, and S. Haddadin, “Power flow regulation, adaptation, and learning for intrinsically robust virtual energy tanks”, *IEEE Robotics and Automation Letters*, vol. 5, no. 1, pp. 211–218, 2019.

[36] S. Jung, T. C. Hsia, and R. G. Bonitz, “Force tracking impedance control of robot manipulators under unknown environment”, *IEEE Transactions on Control Systems Technology*, vol. 12, no. 3, pp. 474–483, 2004.

[37] W. Amanhoud, M. Khoramshahi, and A. Billard, “A dynamical system approach to motion and force generation in contact tasks”, Robotics: Science and Systems (RSS), 2019.

[38] D. Lee, H. Seo, I. Jang, S. J. Lee, and H. J. Kim, “Aerial manipulator pushing a movable structure using a dob-based robust controller”, *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 723–730, 2020.

[39] A. Suarez, G. Heredia, and A. Ollero, “Physical-virtual impedance control in ultra-lightweight and compliant dual-arm aerial manipulators”, *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2553–2560, 2018.

[40] D. J. Hyun, S. Seok, J. Lee, and S. Kim, “High speed trot-running: Implementation of a hierarchical controller using proprioceptive impedance control on the mit cheetah”, *The International Journal of Robotics Research*, vol. 33, no. 11, pp. 1417–1445, 2014.

[41] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, “Learning quadrupedal locomotion over challenging terrain”, *Science robotics*, vol. 5, no. 47, 2020.

[42] D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl, “Learning by cheating”, in *Conference on Robot Learning*, PMLR, 2020, pp. 66–75.

[43] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, "Modelling, planning and control", *Advanced Textbooks in Control and Signal Processing. Springer,*, 2009.

[44] F. J. Abu-Dakka and M. Saveriano, "Variable impedance control and learning—a review", *Frontiers in Robotics and AI*, vol. 7, 2020.

[45] C. Yang, G. Ganesh, S. Haddadin, S. Parusel, A. Albu-Schaeffer, and E. Burdet, "Human-like adaptation of force and impedance in stable and unstable interactions", *IEEE transactions on robotics*, vol. 27, no. 5, pp. 918–930, 2011.

[46] C. Semini, V. Barasuol, T. Boaventura, *et al.*, "Towards versatile legged robots through active impedance control", *The International Journal of Robotics Research*, vol. 34, no. 7, pp. 1003–1020, 2015.

[47] C. C. Beltran-Hernandez, D. Petit, I. G. Ramirez-Alpizar, and K. Harada, "Variable compliance control for robotic peg-in-hole assembly: A deep-reinforcement-learning approach", *Applied Sciences*, vol. 10, no. 19, p. 6923, 2020.

[48] C. C. Beltran-Hernandez, D. Petit, I. G. Ramirez-Alpizar, *et al.*, "Learning force control for contact-rich manipulation tasks with rigid position-controlled robots", *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5709–5716, 2020.

[49] L. Roveda, J. Maskani, P. Franceschi, *et al.*, "Model-based reinforcement learning variable impedance control for human-robot collaboration", *Journal of Intelligent & Robotic Systems*, vol. 100, no. 2, pp. 417–433, 2020.

[50] M. Bogdanovic, M. Khadiv, and L. Righetti, "Learning variable impedance control for contact sensitive tasks", *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6129–6136, 2020.

[51] C. Li, Z. Zhang, G. Xia, X. Xie, and Q. Zhu, "Efficient force control learning system for industrial robots based on variable impedance control", *Sensors*, vol. 18, no. 8, p. 2539, 2018.

[52] W. Zhang, M. W. Mueller, and R. D'Andrea, "A controllable flying vehicle with a single moving part", in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, IEEE, 2016, pp. 3275–3281.

[53] W. Zhang, M. W. Mueller, and R. D'Andrea, "Design, modeling and control of a flying vehicle with a single moving part that can be positioned anywhere in space", *Mechatronics*, vol. 61, pp. 117–130, 2019.

[54] W. Zhang, M. Brunner, L. Ott, M. Kamel, R. Siegwart, and J. Nieto, "Learning dynamics for improving control of overactuated flying systems", *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5283–5290, 2020.

[55] W. Zhang, M. Tognon, L. Ott, R. Siegwart, and J. Nieto, "Active model learning using informative trajectories for improved closed-loop control on real robots", in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 4467–4473.

[56] W. Zhang, L. Ott, M. Tognon, and R. Siegwart, "Learning variable impedance control for aerial sliding on uneven heterogeneous surfaces by proprioceptive and tactile sensing", *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 275–11 282, 2022.

[57] W. Zhang, M. W. Mueller, and R. D'Andrea, "A controllable flying vehicle with a single moving part", in *IEEE International Conference on Robotics and Automation*, 2016.

[58] M. Brunner, K. Bodie, M. Kamel, W. Zhang, J. Nieto, and R. Siegwart, "Trajectory tracking nonlinear model predictive control for an overactuated mav", in *IEEE International Conference on Robotics and Automation*, 2020.

[59] H. Liu, Y.-S. Ong, X. Shen, and J. Cai, "When gaussian process meets big data: A review of scalable gps", *IEEE transactions on neural networks and learning systems*, vol. 31, no. 11, pp. 4405–4423, 2020.

[60] S. Vijayakumar and S. Schaal, "Locally weighted projection regression: An o (n) algorithm for incremental real time learning in high dimensional space", in *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000)*, Morgan Kaufmann, vol. 1, 2000, pp. 288–293.

[61] D. Nguyen-Tuong and J. Peters, "Local gaussian process regression for real-time model-based robot control", in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2008, pp. 380–385.

[62] C. McCutchen, "Flying machines", *Aeromodeller Magazine, July 1954*, 1954.

[63] R. Å. Norberg, "Autorotation, self-stability, and structure of single-winged fruits and seeds (samaras) with comparative remarks on animal flight", *Biological Reviews*, vol. 48, no. 4, pp. 561–596, 1973.

[64] G. Matič, M. Topič, and M. Jankovec, "Mathematical model of a monocopter based on unsteady blade-element momentum theory", *Journal of Aircraft*, vol. 52, no. 6, pp. 1905–1913, 2015.

[65] B. Obradovic, G. Ho, R. Barto, K. Fregene, and D. Sharp, "A multi-scale simulation methodology for the samarai monocopter μuav", in *AIAA Modeling and Simulation Technologies Conference*, 2012, p. 5012.

[66] G. Richards, "Christmas under control!", *Engineering & Technology*, vol. 5, no. 18, pp. 42–43, 2010.

[67] *Flower Flutterbye Fairy, Spin Masters*, (Date last accessed 18-July-2018). [Online]. Available: `https://www.spinmaster.com/product_detail.php?pid=p10171`.

[68] C. De Wagter, S. Tijmons, B. D. Remes, and G. C. de Croon, "Autonomous flight of a 20-gram flapping wing mav with a 4-gram onboard stereo vision system", in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, IEEE, 2014, pp. 4982–4987.

[69] K. Y. Ma, P. Chirarattananon, S. B. Fuller, and R. J. Wood, "Controlled flight of a biologically inspired, insect-scale robot", *Science*, vol. 340, no. 6132, pp. 603–607, 2013.

[70] M. Keennon, K. Klingebiel, and H. Won, "Development of the nano hummingbird: A tailless flapping wing micro air vehicle", in *50th AIAA aerospace sciences meeting including the new horizons forum and aerospace exposition*, 2012, p. 588.

[71] J. Paulos and M. Yim, “Flight performance of a swashplateless micro air vehicle”, in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, IEEE, 2015, pp. 5284–5289.

[72] *B-IONIC Airfish, FESTO*, (Date last accessed 16-July-2018). [Online]. Available: `https://www.festo.com/net/SupportPortal/Files/344798/b_IONIC_Airfish_en.pdf`.

[73] D. S. Drew, N. O. Lambert, C. B. Schindler, and K. S. Pister, “Toward controlled flight of the ionocraft: A flying microrobot using electrohydrodynamic thrust with onboard sensing and no moving parts”, *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 2807–2813, 2018.

[74] M. W. Mueller and R. D’Andrea, “Stability and control of a quadrocopter despite the complete loss of one, two, or three propellers”, in *IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2014, pp. 45–52.

[75] M. W. Mueller and R. D’Andrea, “Relaxed hover solutions for multicopters: Application to algorithmic redundancy and novel vehicles”, *The International Journal of Robotics Research*, vol. 35, no. 8, pp. 873–889, 2016.

[76] G.-X. Du, Q. Quan, B. Yang, and K.-Y. Cai, “Controllability analysis for multirotor helicopter rotor degradation and failure”, *Journal of Guidance, Control, and Dynamics*, vol. 38, no. 5, pp. 978–985, 2015.

[77] P. Martin and E. Salaün, “The true role of accelerometer feedback in quadrotor control”, in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, IEEE, 2010, pp. 1623–1629.

[78] P. Pounds, R. Mahony, P. Hynes, and J. M. Roberts, “Design of a four-rotor aerial robot”, in *Proceedings of the 2002 Australasian Conference on Robotics and Automation (ACRA 2002)*, Australian Robotics & Automation Association, 2002, pp. 145–150.

[79] G. J. Leishman, *Principles of helicopter aerodynamics with CD extra*. Cambridge university press, 2006.

[80] B. Etkin and L. D. Reid, *Dynamics of flight: stability and control*. Wiley New York, 1996, vol. 3.

[81] M. D. Shuster, “A survey of attitude representations”, *Navigation*, vol. 8, no. 9, pp. 439–517, 1993.

[82] F. M. Callier and C. A. Desoer, *Linear system theory*. Springer Science & Business Media, 2012.

[83] D. S. Bernstein, *Matrix Mathematics: Theory, Facts, and Formulas . Princeton reference*, 2nd ed. Princeton University Press, 2009.

[84] S. Lupashin, M. Hehn, M. W. Mueller, A. P. Schoellig, M. Sherback, and R. D’Andrea, “A platform for aerial robotics research and demonstration: The flying machine arena”, *Mechatronics*, vol. 24, no. 1, pp. 41–54, 2014.

[85] D. P. Bertsekas, *Dynamic programming and optimal control*. Athena scientific Belmont, MA, 2005, vol. 1.

[86] P. J. Antsaklis and A. N. Michel, *Linear systems*. Springer Science & Business Media, 2006.

[87] M. Ryll, H. H. Bülthoff, and P. R. Giordano, "Modeling and control of a quadrotor uav with tilting propellers", in *IEEE International Conference on Robotics and Automation*, 2012.

[88] D. Brescianini and R. D'Andrea, "An omni-directional multirotor vehicle", *Mechatronics*, 2018.

[89] S. Park, J. Lee, J. Ahn, *et al.*, "Odar: Aerial manipulation platform enabling omnidirectional wrench generation", *IEEE/ASME Transactions on mechatronics*, 2018.

[90] N. Staub, D. Bicego, Q. Sablé, V. Arellano, S. Mishra, and A. Franchi, "Towards a flying assistant paradigm: The othex", in *IEEE International Conference on Robotics and Automation*, 2018.

[91] M. Kamel, S. Verling, O. Elkhatib, *et al.*, "Voliro: An omnidirectional hexacopter with tiltable rotors", *arXiv preprint arXiv:1801.04581*, 2018.

[92] M. Kamel, M. Burri, and R. Siegwart, "Linear vs nonlinear mpc for trajectory tracking applied to rotary wing micro aerial vehicles", *IFAC-PapersOnLine*, 2017.

[93] D. Nguyen-Tuong and J. Peters, "Model learning for robot control: A survey", *Cognitive processing*, 2011.

[94] G. Chowdhary, H. A. Kingravi, J. P. How, and P. A. Vela, "Bayesian nonparametric adaptive control using gaussian processes", *IEEE transactions on neural networks and learning systems*, 2014.

[95] D. Nguyen-Tuong, J. R. Peters, and M. Seeger, "Local gaussian process regression for real time online model learning", in *Advances in Neural Information Processing Systems*, 2009.

[96] F. Meier, D. Kappler, N. Ratliff, and S. Schaal, "Towards robust online inverse dynamics learning", in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2016.

[97] T. Beckers, J. Umlauft, D. Kulic, and S. Hirche, "Stable gaussian process based tracking control of lagrangian systems", in *IEEE Conference on Decision and Control*, 2017.

[98] M. K. Helwa, A. Heins, and A. P. Schoellig, "Provably robust learning-based approach for high-accuracy tracking control of lagrangian systems", *IEEE Robotics and Automation Letters*, 2019.

[99] D. Mitrovic, S. Klanke, and S. Vijayakumar, "Adaptive optimal feedback control with learned internal dynamics models", in *From Motor Learning to Interaction Learning in Robots*, 2010.

[100] J. Kabzan, L. Hewing, A. Liniger, and M. N. Zeilinger, "Learning-based model predictive control for autonomous racing", *IEEE Robotics and Automation Letters*, 2019.

[101] C. J. Ostafew, A. P. Schoellig, and T. D. Barfoot, “Learning-based nonlinear model predictive control to improve vision-based mobile robot path-tracking in challenging outdoor environments”, in *IEEE International Conference on Robotics and Automation*, 2014.

[102] G. Cao, E. M.-K. Lai, and F. Alam, “Gaussian process model predictive control of an unmanned quadrotor”, *Journal of Intelligent & Robotic Systems*, 2017.

[103] M. Deisenroth and C. E. Rasmussen, “Pilco: A model-based and data-efficient approach to policy search”, in *International Conference on machine learning*, 2011.

[104] M. Saveriano, Y. Yin, P. Falco, and D. Lee, “Data-efficient control policy search using residual dynamics learning”, in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017.

[105] J. Ko, D. J. Klein, D. Fox, and D. Haehnel, “Gaussian processes and reinforcement learning for identification and control of an autonomous blimp”, in *IEEE International Conference on Robotics and Automation*, 2007.

[106] A. Spitzer and N. Michael, “Inverting learned dynamics models for aggressive multirotor control”, *arXiv preprint arXiv:1905.13441*, 2019.

[107] C. E. Rasmussen, “Gaussian processes in machine learning”, Springer, 2003.

[108] V. R. Desaraju, A. Spitzer, and N. Michael, “Experience-driven predictive control with robust constraint satisfaction under time-varying state uncertainty.”, in *Robotics: Science and Systems*, 2017.

[109] F. Furrer, M. Burri, M. Achtelik, and R. Siegwart, “Robot operating system (ros): The complete reference (volume 1)”, in A. Koubaa, Ed. Cham: Springer International Publishing, 2016, ch. RotorS—A Modular Gazebo MAV Simulator Framework, pp. 595–625, ISBN: 978-3-319-26054-9. DOI: `10.1007/978-3-319-26054-9_23`. [Online]. Available: `http://dx.doi.org/10.1007/978-3-319-26054-9_23`.

[110] *Adis16448 fact sheet*, (Date last accessed 5-May-2020). [Online]. Available: `https://www.analog.com/media/en/technical-documentation/data-sheets/ADIS16448.pdf`.

[111] GPy, *GPy: A gaussian process framework in python*, `http://github.com/SheffieldML/GPy`, since 2012.

[112] S. G. Johnson, *The nlopt nonlinear-optimization package*, (Date last accessed 10-Sept-2019). [Online]. Available: `http://github.com/stevengj/nlopt`.

[113] A. Novikov, “PyClustering: Data mining library”, *Journal of Open Source Software*, 2019.

[114] P. Abbeel, A. Coates, and A. Y. Ng, “Autonomous helicopter aerobatics through apprenticeship learning”, *The International Journal of Robotics Research*, vol. 29, no. 13, pp. 1608–1639, 2010.

[115] M. Kamel, T. Stastny, K. Alexis, and R. Siegwart, “Model predictive control for trajectory tracking of unmanned aerial vehicles using robot operating system”, in *Robot operating system (ROS)*, Springer, 2017, pp. 3–39.

[116] S. Kuindersma, R. Deits, M. Fallon, *et al.*, “Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot”, *Autonomous robots*, vol. 40, no. 3, pp. 429–455, 2016.

[117] M. T. Gillespie, C. M. Best, E. C. Townsend, D. Wingate, and M. D. Killpack, “Learning nonlinear dynamic models of soft robots for model predictive control with neural networks”, in *2018 IEEE International Conference on Soft Robotics (RoboSoft)*, IEEE, 2018, pp. 39–45.

[118] D. Nguyen-Tuong and J. Peters, “Using model knowledge for learning inverse dynamics”, in *IEEE International Conference on Robotics and Automation*, 2010.

[119] B. Settles, “Active learning literature survey”, University of Wisconsin-Madison Department of Computer Sciences, Tech. Rep., 2009.

[120] P. Schrangl, P. Tkachenko, and L. del Re, “Iterative model identification of nonlinear systems of unknown structure: Systematic data-based modeling utilizing design of experiments”, *IEEE Control Systems Magazine*, vol. 40, no. 3, pp. 26–48, 2020.

[121] A. D. Wilson, J. A. Schultz, A. R. Ansari, and T. D. Murphey, “Dynamic task execution using active parameter identification with the baxter research robot”, *IEEE Transactions on Automation Science and Engineering*, vol. 14, no. 1, pp. 391–397, 2016.

[122] T. Koller, F. Berkenkamp, M. Turchetta, and A. Krause, “Learning-based model predictive control for safe exploration”, in *2018 IEEE Conference on Decision and Control (CDC)*, IEEE, 2018, pp. 6059–6066.

[123] M. Buisson-Fenet, F. Solowjow, and S. Trimpe, “Actively learning gaussian process dynamics”, *arXiv preprint arXiv:1911.09946*, 2019.

[124] C. Zimmer, M. Meister, and D. Nguyen-Tuong, “Safe active learning for time-series modeling with gaussian processes”, in *Advances in Neural Information Processing Systems*, 2018, pp. 2730–2739.

[125] Y. K. Nakka, A. Liu, G. Shi, A. Anandkumar, Y. Yue, and S.-J. Chung, “Chance-constrained trajectory optimization for safe exploration and learning of nonlinear systems”, *arXiv preprint arXiv:2005.04374*, 2020.

[126] F. Borrelli, A. Bemporad, and M. Morari, *Predictive control for linear and hybrid systems*. Cambridge University Press, 2017.

[127] A. Capone, G. Noske, J. Umlauft, T. Beckers, A. Lederer, and S. Hirche, “Localized active learning of gaussian process state space models”, in *Learning for Dynamics and Control*, PMLR, 2020, pp. 490–499.

[128] P. Congdon, *Bayesian statistical modelling*. John Wiley & Sons, 2007, vol. 704.

[129] T. Homem-de-Mello and G. Bayraksan, “Monte carlo sampling-based methods for stochastic optimization”, *Surveys in Operations Research and Management Science*, vol. 19, no. 1, pp. 56–85, 2014.

[130] J. MacQueen *et al.*, “Some methods for classification and analysis of multivariate observations”, in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, Oakland, CA, USA, vol. 1, 1967, pp. 281–297.

[131] W. Amanhoud, M. Khoramshahi, M. Bonnesoeur, and A. Billard, “Force adaptation in contact tasks with dynamical systems”, in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 6841–6847.

[132] N. Hogan, “Impedance control: An approach to manipulation”, in *1984 American control conference*, IEEE, 1984, pp. 304–313.

[133] D. S. Walker, J. K. Salisbury, and G. Niemeyer, “Demonstrating the benefits of variable impedance to telerobotic task execution”, in *2011 IEEE International Conference on Robotics and Automation*, IEEE, 2011, pp. 1348–1353.

[134] R. Ikeura and H. Inooka, “Variable impedance control of a robot for cooperation with a human”, in *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, IEEE, vol. 3, 1995, pp. 3097–3102.

[135] A. Y. Mersha, S. Stramigioli, and R. Carloni, “Variable impedance control for aerial interaction”, in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2014, pp. 3435–3440.

[136] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[137] J. Buchli, F. Stulp, E. Theodorou, and S. Schaal, “Learning variable impedance control”, *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 820–833, 2011.

[138] A. Garofano-Soldado, G. Heredia, and A. Ollero, “Aerodynamic interference in confined environments with tilted propellers: Wall effect and corner effect”, in *2021 Aerial Robotic Systems Physically Interacting with the Environment (AIRPHARO)*, IEEE, 2021, pp. 1–8.

[139] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms”, *arXiv preprint arXiv:1707.06347*, 2017.

[140] R. Martın-Martın, M. A. Lee, R. Gardner, S. Savarese, J. Bohg, and A. Garg, “Variable impedance control in end-effector space: An action space for reinforcement learning in contact-rich tasks”, in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2019, pp. 1010–1017.

[141] E. Spyrakos-Papastavridis, P. R. Childs, and J. S. Dai, “Passivity preservation for variable impedance control of compliant robots”, *IEEE/ASME Transactions on Mechatronics*, vol. 25, no. 5, pp. 2342–2353, 2019.

[142] S. A. Khader, H. Yin, P. Falco, and D. Kragic, “Stability-guaranteed reinforcement learning for contact-rich manipulation”, *IEEE Robotics and Automation Letters*, vol. 6, no. 1, pp. 1–8, 2020.

[143] J. Hwangbo, J. Lee, and M. Hutter, “Per-contact iteration method for solving contact dynamics”, *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 895–902, 2018. [Online]. Available: `www.raisim.com`.

List of acronyms

GP Gaussian Process

ROS Robot Operating System

Curriculum Vitae

Weixuan Zhang
Mail: zhangweixuanpeter@gmail.com
Date of Birth: 26.08.1989
Nationality: China

Education

2019 - 2022	**Autonomous Systems Lab, ETH Zurich, Switzerland** Ph.D. Student, Supervisor: Prof. Roland Siegwart
2015 - 2019	**Institute for Dynamics and Control, ETH Zurich, Switzerland** Ph.D. Student, Supervisor: Prof. Raffaello D'Andrea
10.2013 - 04.2015	**ETH Zurich, Switzerland** Master of Science in Mechanical Engineering
04.2013 - 09.2013	**Bosch GmbH, Germany** 5 month industry internship
09.2010 - 09.2013	**Karlsruhe Institute of Technology, Germany** Bachelor of Science in Mechanical Engineering
09.2001 - 06.2007	**Nanjing Foreign Language School, China** High school diploma

Publications Included in this Thesis

[P1] W. Zhang, M. W. Mueller, and R. D'Andrea, "Design, modeling and control of a flying vehicle with a single moving part that can be positioned anywhere in space", *Mechatronics*, vol. 61, pp. 117–130, 2019

[P2] W. Zhang, M. Brunner, L. Ott, M. Kamel, R. Siegwart, and J. Nieto, "Learning dynamics for improving control of overactuated flying systems", *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5283–5290, 2020

[P3] W. Zhang, M. Tognon, L. Ott, R. Siegwart, and J. Nieto, "Active model learning using informative trajectories for improved closed-loop control on real robots", in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 4467–4473

[P4] W. Zhang, L. Ott, M. Tognon, and R. Siegwart, "Learning variable impedance control for aerial sliding on uneven heterogeneous surfaces by proprioceptive and tactile sensing", *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 275–11 282, 2022