Methods for Online Predictive Control of Multi-rotor Aerial Robots with Perception-driven Tasks subject to Sensing and Actuation Constraints

Martin Jacquet

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :

l'Institut National des Sciences Appliquées de Toulouse (INSA de Toulouse)

Présentée et soutenue le *23/08/2022* **par :**

Martin Jacquet

Methods for Online Predictive Control of Multi-rotor Aerial Robots with Perception-driven Tasks subject to Sensing and Actuation Constraints

---

JURY

Kostas Alexis	Full Professor	Rapporteur
Paolo Robuffo Giordano	Directeur de Recherche	Rapporteur
Margarita Chli	Assistant Professor	Examinatrice
Simon Lacroix	Directeur de Recherche	Président du Jury
Antonio Franchi	Full Professor	Directeur de thèse

École doctorale et spécialité :

EDSYS : Robotique 4200046

Unité de Recherche :

LAAS - Laboratoire d'Analyse et d'Architecture des Systèmes (UPR 8001)

Directeur de Thèse :

Antonio Franchi

Rapporteurs :

Paolo Robuffo Giordano et *Kostas* Alexis

Table of Contents

Table of Contents i

Résumé en français v

English Abstract vii

Acknowledgments ix

List of Acronyms xi

List of Figures xiv

List of Tables xvii

List of Multimedias xviii

I Preliminaries 1

1 Introduction 3
1.1 Uncrewed Aerial Vehicles . . . 4
1.2 Context and Objectives . . . 9
1.3 Contributions . . . 11
1.4 Thesis Outline . . . 12

2 Perception with Aerial Robots: A Literature Review 15
2.1 Introduction . . . 16
2.2 Multi-Rotor Designs . . . 16
2.3 Exteroceptive Perception with ARs . . . 19
2.4 Vision-Based Motion Generation . . . 23
2.5 N-MPC for Aerial Robots . . . 25
2.6 Perception Aware N-MPC . . . 31

3 Modeling 35
3.1 Introduction . . . 36
3.2 Notations . . . 36
3.3 Orientation Representations . . . 38
3.4 Generically Tilted Multi-Rotor . . . 42
3.5 Sensor Model . . . 46
3.6 Fiducial markers . . . 48
3.7 Sensor measurement filtering . . . 50

II Perception Awareness in N-MPC 53

4 Generic Problem Statement for Generically Tilted Multi-Rotor 55
4.1 Introduction . . . 56
4.2 The choice of N-MPC . . . 57
4.3 Problem Statement . . . 58
4.4 NonLinear Programming . . . 59
4.5 Conclusion . . . 62

5 Perception constrained N-MPC 65
5.1 Introduction . . . 66
5.2 Geometric Perception Criterion . . . 66
5.3 Perception Objectives . . . 67
5.4 FoV Constraints . . . 69
5.5 NonLinear Programming Formulation . . . 73
5.6 Extension to Multiple Features and Sensors . . . 74
5.7 Experimental and Simulation results . . . 75
5.8 Application to Human-Robot Handover . . . 81
5.9 Conclusion . . . 84

6 Enforced Vision-Based Localization 87
6.1 Introduction . . . 88
6.2 Related Works . . . 89
6.3 Error State Kalman Filter . . . 90
6.4 Modified N-MPC formulation . . . 94
6.5 Experimental and Simulation Results . . . 99
6.6 Conclusion . . . 104

7 Extension to Multi-Agent Systems 107
7.1 Introduction . . . 108
7.2 Centralized and Decentralized Implementations . . . 108
7.3 Collision Avoidance . . . 110
7.4 Constraints and Objectives with Several Agents . . . 112
7.5 Multi-Agent Collective Localization . . . 113
7.6 Conclusion . . . 114

8 Active Information Acquisition 115
8.1 Introduction . . . 116
8.2 Literature Review . . . 116
8.3 Intermittent Observation Modeling . . . 121
8.4 Leverage Uncertainty Minimization in N-MPC . . . 124
8.5 Experimental and Simulation Validation . . . 128
8.6 Conclusion . . . 135

III Conclusions, Perspectives and Appendices 137

9 Conclusion 139
9.1 Synthesis of Contributions . . . 140

9.2 Overall Conclusion 141
9.3 Perspectives 142
9.4 Final Thoughts 143

A Simulation and Experimental Setup **145**

B Implementation Details **151**

C Center of Mass Estimation **157**

D Collaborative Works **161**
D.1 Participation to MBZIRC 2020 162
D.2 Generic Control Scheme for Vision-Based Physical Interaction 165

Bibliography **173**

Résumé - Abstract **201**

Résumé en français

Les drones se sont d'ors et déjà rendus indispensables dans de très nombreuses applications, notamment dans les domaines de la création photographique ou vidéo, ou tout simplement dans des activités de loisir. En parallèle, l'image des robots aériens autonomes s'est elle aussi répandue dans l'imaginaire collectif, de telle sorte que de très nombreuses applications, du quotidien comme dans des domaines plus spécifiques, s'envisagent aujourd'hui fortement sous ce spectre. On peut notamment citer l'image récurrente de livraison à domicile par des drones, ou bien le déploiement de flottes de robots pour des activités d'observation en environnement difficile d'accès. La recherche en robotique aérienne est, elle, très active depuis de nombreuses années et l'état de l'art ne cesse de s'améliorer, que ce soit via la conception de robots de plus en plus performants ou le perfectionnement des algorithmes embarqués qui permettent une autonomie toujours plus grande.

Le déploiement de ces robots dans le cadre d'applications en environnement non-contrôlé pose de nombreux problèmes logistiques, notamment liés à la perception de l'environnement. L'usage de capteurs extéroceptifs est donc fondamental pour la plupart des applications en autonomie. Parmi ces capteurs, les caméras tiennent notamment une place de choix. Cette place tient d'une part de leur simplicité de conception et d'intégration, due à leurs très faibles tailles et poids, mais également de la conception même des environnements de travail humains, qui sont très fortement basés sur des marqueurs visuels (panneaux, signaux lumineux...) En revanche, les impératifs liés au maintien de la visibilité de certains objets ou phénomènes entrent régulièrement en conflit avec les autres tâches à accomplir pour le robot, ou tout simplement avec ses impératifs de mouvement. Cet effet est d'autant plus important que la plupart des robots aériens possèdent un fort couplage entre la position du robot et son orientation : le robot doit s'incliner dans une direction donnée afin de se déplacer latéralement, affectant donc l'orientation des capteurs. Partant de ce constat, la communauté scientifique travaille à la production d'algorithmes sensorimoteurs pouvant produire des mouvements tout en tenant compte de la perception de l'environnement.

Cette thèse s'inscrit dans ce contexte, en vue de proposer des méthodes de contrôle incluant des contraintes liées au maintien des phénomènes d'intérêt dans le champ de vision des capteurs. De plus, afin d'assurer la faisabilité des commandes générées, il est nécessaire de prendre en compte les différentes contraintes d'actionnement des robots.

Afin de lier ces différents aspects dans un formalisme commun, les solutions proposées s'appuient sur des méthodes de contrôle *optimal* et *prédictif*. Ces méthodes, basées sur de l'optimisation numérique, nécessitent la définition de modèles

dynamiques et perceptifs précis et complets, ce qui implique de considérer les *non-linéarités* des systèmes, qui sont parfois mises de côté à des fins de simplification.

Les contributions de cette thèse consistent dans un premier temps en l'agrégation des différents concepts dans un paradigme commun, puis à la formalisation des fonctions mathématiques permettant d'exprimer les objectifs et contraintes relatifs à la perception. Ce même paradigme est utilisé pour la résolution de plusieurs problèmes liés à des tâches courantes en robotique aérienne, notamment le suivi de phénomènes dynamiques, ou encore la localisation visuelle-inertielle. Finalement, les solutions proposées sont implémentées sur les robots et éprouvées via diverses simulations et expériences.

Les travaux effectués au cours de cette thèse ont donné lieu à diverses publications dans des conférences et journaux internationaux. L'intégralité des productions logicielles issues de ces travaux sont publiées en tant que code libre à disposition de la communauté robotique.

Mots clefs - Perception, Contrôle Prédictif, Robots Aériens, Systèmes Multi-Robots

English Abstract

Drones have an increasing place in numerous applications that already started to take advantage of those, in particular in the fields of photography and video making, or simply for leisure activities. Simultaneously, the picture of autonomous aerial robots as a mark of innovation has spread, such that many civilian or industrial applications are now envisioned through this aspect. One could cite, for instance, the persistent idea of aerial home delivery of goods, advertised by many companies. Another common use case is the deployment of fleets of aerial robots for *monitoring* activities, in hard-to-access environments, such as high mountains. The aerial robotics research community has been active for numerous years, and the state of the art keeps improving, through the conception of novel, more adaptive control algorithms, or the improvements of the hardware designs, opening new ranges of possibilities.

The deployment of such robots in uncontrolled environments comes with a lot of challenges, in particular regarding the perception of the surroundings. Exteroceptive sensors are indeed mandatory for most of autonomous applications. Among those sensors, cameras hold a peculiar position. It is due, on the one hand, to the simple onboard integration with their small size and weight, and on the other hand to the design of human-made environments, which are heavily built around visual markers (signs, illuminated signals...) However, maintaining visibility over objects or phenomena often collides with the motion requirements of the robot, or with the tasks to which it is assigned. This effect is prominent when using *underactuated* robots, which are the most widely spread types of aerial vehicles, partly because of their higher energy efficiency. This property implies a strong coupling between position and orientation: the robot needs to tilt to move, and corollary moves when it tilts, thus altering the sensor bearing. From this assessment, the robotics community works to produce sensorimotor algorithms, able to produce motions while accounting for perception.

This thesis takes place in this context, aiming at proposing such control methods to enforce the visibility over a phenomenon of interest through the onboard sensors. Moreover, to ensure the feasibility of the generated commands, it is required to account for the various actuation limitations of the robots. Finally, this thesis devotes to propose generic formulations, thus avoiding to propose *ad hoc* solutions, which would be contingent on a specific problem.

To tackle these aspects under a common formalism, the proposed solutions are based on *optimal* and *predictive* control policies. These are based on numerical optimization, implying the need for accurate models, and thus accounting for the system *nonlinearities*, which are often disregarded for simplification.

The contributions of this thesis are the aggregation of the aforementioned concepts

in a common paradigm, and the formalization of the corresponding mathematical functions transcribing the objectives and constraints related to perception. This paradigm is used in the scope of several applications related to usual perception-driven tasks in aerial robotics, namely the tracking of dynamic phenomena and its improvement, or the visual-inertial localization.

The work conducted throughout this thesis led to various publications in international peer-reviewed conferences and journals. All the related software productions are published open-source for the robotics community.

Keywords - Perception, Predictive Control, Aerial Robots, Multi-Robot Systems

Acknowledgments

Je tiens tout d'abord à exprimer ma gratitude à Antonio Franchi, mon directeur de thèse, pour m'avoir permis d'embarquer dans cette aventure au LAAS. Merci pour le support et les conseils, ainsi que les nombreuses discussions enrichissantes au fil des ans. Je lui suis profondément reconnaissant pour la motivation et l'incitation constante à viser le mieux, sans quoi cette thèse n'aurait certainement pas eu ce visage.

Je tiens également à remercier les membres de mon jury de thèse, Paolo Robuffo Giordano, Kostas Alexis, Margarita Chli et Simon Lacroix, pour avoir accepter de prendre le temps d'évaluer mes travaux, et pour les nombreux retours constructifs que j'ai pu avoir dans les rapports ainsi que lors de ma soutenance.

Ma gratitude va évidemment à toutes les personnes avec qui j'ai pu travailler, au sein de l'équipe RIS et du groupe RAM à Twente, et à qui je dois en partie le succès des expériences conduites lors de ma thèse. Je tiens particulièrement à remercier Anthony Mallet pour la maintenance des robots, et pour avoir répondu patiemment à mes très nombreuses questions. Je souhaite également remercier Gianluca Corsini, pour les nombreuses discussions et son aide continue, ainsi que pour m'avoir accompagné dans les longues soirées (ou nuits) en périodes de deadlines.

Finalement, je veux remercier du fond du coeur toutes les personnes que j'ai cotoyé pendant mes 5 ans et demi au LAAS, que ce soit dans les équipes RAP, RIS, DISCO ou Gepetto, ou même en dehors du labo. J'ai pu rencontrer des gens formidables qui ont fait de ces années une période de joie, de rire, et de partage (de gâteaux et de bières, notamment). Merci à Cyrille pour les (trop) longues pauses midi, à Sabrina pour les fous rires, à Louise pour les tutos couture, à Kévin, Jessica et Pierre pour les soirées jeux de rôles, et aux autres croisé·e·s au détour d'une partie de tarot, que je ne cite pas ici. Merci à ma famille et à mes amis, Laura et les autres, pour m'avoir aidé à décompresser en parallèle du travail. Merci à mon bureau officieux, les B-181unièmes Alexis, Aurélie, Diane et Noëlie, pour votre amour et votre soutien lors des périodes difficiles. Et enfin merci à la meilleure co-bureau, Claire, pour m'avoir accompagné sur les trajets "matinaux" à vélo et pour avoir supporté toutes mes râleries aussi longtemps.

List of Acronyms

AF Admittance Filter

AIA Active Information Acquisition

ANR French National Research Agency

AR Aerial Robot

BVP Boundary-Value Problem

CAD computer aided design

CCW Counter-Clockwise

CNN Convolutional Neural Network

CoM Center of Mass

CPU Central Processing Unit

CW Clockwise

DARPA Defense Advanced Research Projects Agency

DoF Degree of Freedom

EKF Extended Kalman Filter

ESC Electronic Speed Controller

ESKF Error State Kalman Filter

FoV Field of View

FPGA Field-Programmable Gate Array

GPS Global Positioning System

GPU Graphical Processing Unit

GTMR Generically Tilted Multi-Rotor

HVS Hybrid Visual Servoing

I2C Inter-Integrated Circuit

IBVS Image-Based Visual Servoing

IKF Intermittent Kalman Filter

IMU Inertial Measurement Unit

INDI Incremental Nonlinear Dynamic Inversion

IPPE Infinitesimal Plane-based Pose Estimation

KF Kalman Filter

KKT Karush-Kuhn-Tucker

LAAS-CNRS Laboratory for Analysis and Architecture of Systems of the French CNRS

lidar LIght Detection And Ranging

LQR Linear-Quadratic Regulator

MAP Maximum A Posteriori

MBZIRC Mohamed bin Zayed International Robotics Challenge

ML Maximum Likelyhood

MoCap Motion Capture

MPC Model Predictive Control

MRS Multi-Robots Systems

MuRoPhen MuRoPhen (Multiple Robots for observing dynamical Phenomena)

NLP NonLinear Programming

N-MPC Nonlinear MPC

OCP Optimal Control Problem

PBVS Position-Based Visual Servoing

PnP Perspective-n-Points

QP Quadratic programming

RANSAC RANdom SAmpling Consensus

RGBD RGB + Depth

RIS Robotics and InteractionS

RMSE Root Mean Square Error

RTI Real-Time Iteration

RTK-GPS Real-Time Kinematic GPS

SfM Structure from Motion

SLAM Simultaneous Localization And Mapping

SQP Sequential Quadratic programming

std standard deviation

UAV Uncrewed Aerial Vehicle

UGV Uncrewed Ground Vehicle

UKF Unscented Kalman Filter

VIO Visual-Inertial Odometry

VISLAM Visual-Inertial SLAM

VS Visual Servoing

VTOL Vertical Take-Off and Landing

WO Wrench Observer

w.r.t. with respect to

YOLO You Only Look Once

List of Figures

1.1 Two examples of early partly-autonomous vehicles. 4
1.2 Two UAVs in two international challenges on robotics. 5
1.3 Examples of civil applications of UAVs. 6
1.4 Examples of manipulation-oriented UAVs. 7
1.5 Two instances of complex robot design with rotary wings. 8
1.6 The MuRoPhen project logo. 10

2.1 The thrust directions allowed for a collinear quadrotor (left) and a tilted-propeller hexarotor, the OTHex (right) (courtesy of [Staub, 2018]). 17
2.2 Three examples of non-collinear propeller Aerial Robots. 18
2.3 Schematic view of a range-and-bearing sensor. 20
2.4 Two plausible PnP solutions that describe the detected marker are depicted as the yellow and blue frames, along with the green ground truth (courtesy of [Chng, 2020]). 23
2.5 Example images of IBVS. The task is to bring the end effector onto its goal, denoted respectively with the white square and black cross (courtesy of [Kragic, 2002]). 24

3.1 Double coverage of the rotation manifold (courtesy of [Solà, 2017]) . . 42
3.2 Schematic of the GTMR 43
3.3 Graphical depiction of the position uncertainty ellipsoid (courtesy of [Fourmy, 2022]). 50

4.1 Top-view of a typical task assigned to the AR. 56
4.2 A partial conceptual diagram of the proposed perception-aware controller paradigm. 58
4.3 Block diagram of the N-MPC framework. 63

5.1 Depiction of the visibility criterion $c\beta$. 68
5.2 Piecewise-linear function used to adapt the weight on the perception objective. 69
5.3 Visual depiction of the state-dependent α_β angle. 71
5.4 Block diagram of the proposed framework. 73
5.5 Position and attitude tracking when observing a feature moving in circle, with a quadrotor and an hexarotor. 77
5.6 (x, y) position of the GTMR tracking a trajectory while enforcing visibility constraints. 78
5.7 The measured $c\beta$ (solid) and corresponding lower bound $c\alpha_\beta$ (dashed) for the two pairs camera/marker (front-facing in yellow and down-facing in blue). 79

5.8 The measured cβ and its bound cα_β for the two pairs camera/marker (both front- and down-looking). 80
5.9 Top: Thrusts generated by the 6 propellers. Bottom: roll and pitch angles of the AR. 81
5.10 System inputs for the 6 propellers along the simulation. 82
5.11 Visibility constraint over time during the approaching phase. 84

6.1 Graph of the logistic function (Equation (6.19)) for $\lambda = 1$ (red), $\lambda = 10$ (green) and $\lambda = 100$ (blues), with $x_0 = 0$. 95
6.2 Block diagram of the proposed framework. 99
6.3 The (x, y) trajectory of the two experiments reported in Video 6.1. . 101
6.4 Simulation with a tilted-propeller hexarotor and a down-facing camera. 102
6.5 Simulations using a quadrotor and a tilted-propeller hexarotor along a 16 m path. 103

7.1 Three examples of hardware-based mutual AR localization systems. 109

8.1 Two heterogeneous ARs, equipped with sensors, observing a static feature. 123
8.2 Block diagram of the N-MPC-AIA framework. 125
8.3 Block diagram of the decentralized AIA framework for MRS. 127
8.4 The (x, y) position, the associated evolution of tr($\mathbf{P}$) and the position error over time for the 1, 2 and 3 agents cases. 129
8.5 The (x, y) position of the hexarotor and the quadrotor assigned with heterogeneous tasks. 132
8.6 Roll, pitch, and actuator thrusts of the hexarotor assigned to estimation improvement in the heterogeneous team. 133
8.7 The (x, y) path of the quadrotor tracking the mobile feature. 134

A.1 Snapshot of a quadrotor used in the various experiments. It is equipped with MoCap reflecting balls on top and a front-facing camera. 146
A.2 Quadrotors and simulated ARs used in simulations and experiments. 147
A.3 The two RealSense cameras used. 148

B.1 Box plot of the computation time (in milliseconds) of the onboard N-MPC. The red is the mean, and the black dots are the slowest 1% control cycles recorded. 154
B.2 Architecture of the AR software stack. In blue are the components developed throughout the thesis. 155

D.1 The arena of the second challenge in MBZIRC 2020. 162
D.2 The LAAS-CNRS aerial platform, the FibertHex. 162
D.3 Histogram back-projection-based detection of the colored bricks (courtesy of [Dantec, 2019]). 163
D.4 Histogram back-projection-based detection of the colored bricks (courtesy of [Dantec, 2019]). 163
D.5 RGBD ground extraction for brick detection (courtesy of [Dantec, 2019]). 164
D.6 Software architecture of the perception software stack. 165
D.7 Generic control architecture for vision-based physical interaction with fully-actuated platforms. 167

D.8 The altitude of the robot in $\mathcal{F}_W$ (top) and the estimated contact force along $\mathbf{z}_B$ (bottom). 171

List of Tables

1.1 Summary of peer-reviewed publications produced during this thesis. . 14

3.1 Usual mathematical symbols used throughout the manuscript. 37

5.1 Table of N-MPC weights for simulations and experiments in Section 5.7. 76
5.2 Reprojection error (mean and std, both in meters) with and without uncertainty estimation, for both the front and down features. 80

6.1 Table of N-MPC weights for simulations and experiments in Section 6.5. 100

8.1 Table of N-MPC weights for simulations and experiments in Section 8.5. 128
8.2 Position covariances (measurement and estimation) mean and std using 1, 2 or 3 agents 131

A.1 State estimation sensors frequencies and simulated noise. 148

C.1 Position RMSE, in meter, for two different trajectories, with and without CoM offset compensation. 159

List of Multimedias

5.1 Comparison between two types of ARs for the observation a circular motion. 76
5.2 Experiments with a quadrotor in two conditions: with and without visibility constraints. 78
5.3 Experiment with a quadrotor: tracking of a moving marker. 79
5.4 Simulation with a fully-actuated hexarotor in a complex monitoring scenario. 81
5.5 Simulation of a fully autonomous Human-Robot handover. On the left, successive positions of the simulated AR performing the handover. On the right, a frame of the robot's onboard camera. 83

6.1 Experiments with a quadrotor: illustration of the visibility constraint. 100
6.2 Simulation with a tilted-propeller hexarotor: enforcing the visibility while moving. 101
6.3 Two simulations with quadrotor and hexarotor: following a long path while performing state estimation. 103

8.1 Experiments and simulation with 1, 2, and 3 ARs in a collaborative observation task. 130
8.2 Heterogeneous system simulation: two different ARs performing two different tasks simultaneously. 132
8.3 Two successive experiments: tracking of a moving marker with 1 and 2 ARs. 134
8.4 Additional simulations: tracking of a moving feature with 1 and 2 ARs. 134

D.1 Experiment of pick and place operation with an hexarotor. 170

Part I

Preliminaries

Chapter 1

Introduction

Contents

1.1 Uncrewed Aerial Vehicles **4**

1.2 Context and Objectives **9**

1.3 Contributions **11**

1.4 Thesis Outline **12**

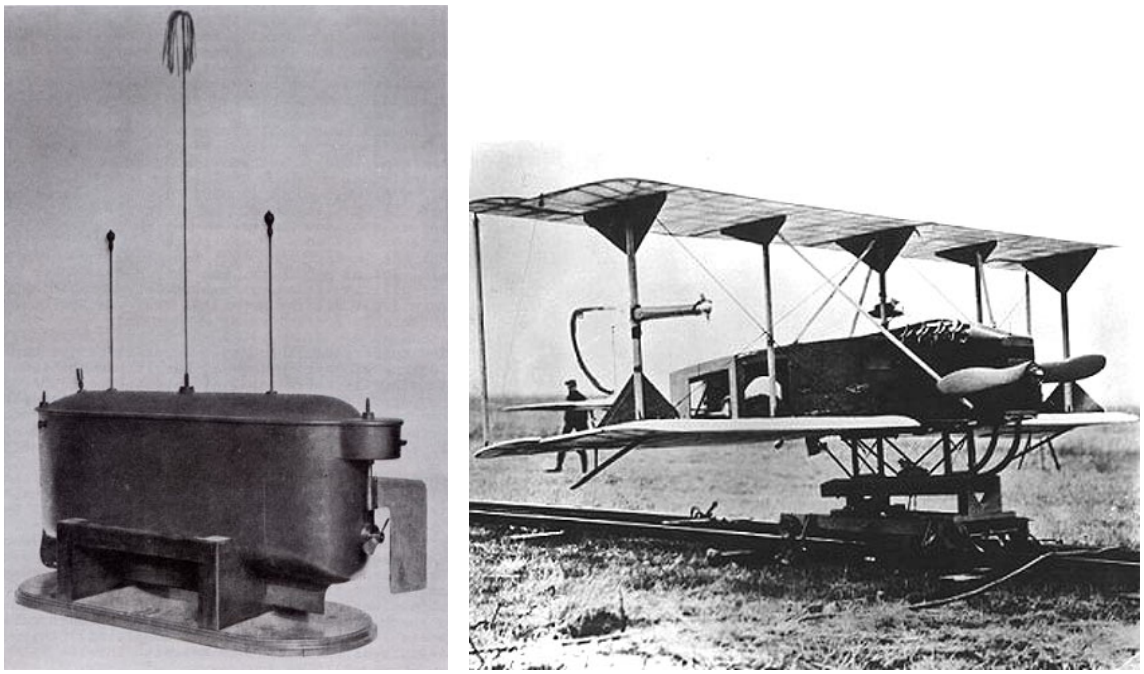
1.1 Uncrewed Aerial Vehicles

The development of Uncrewed Aerial Vehicles (UAVs) roughly followed a similar path as airplanes and manned flights. Balloon prototypes from the 19th century were followed by early powered vehicles in the first half of the 20th century [Newcome, 2004]. The second half of the last century saw large improvements for both the flight capabilities and autonomy of such vehicles. This increasing autonomy was allowed by the miniaturization of electronics, processors and sensors, and enabled the appearance of autonomous Aerial Robots (ARs).

A definition for such robots is proposed in [Feron, 2008], as:

> *An Aerial Robot is a system capable of sustained flight with no direct human control and able to perform a specific task.*

Although this definition encompasses remotely controlled aircrafts, the main focus of the aerial robotics research community is oriented toward increasing the autonomy, aiming for zero – or very minimal – human intervention during the performing of the task. The pursue of autonomy in human-made systems is in fact a key aspect of robotics since its early developments. Putting aside historical examples, [Newcome, 2004] credits the first major step toward autonomous tools to the Telautomaton in 1898 (see Figure 3.3a), the first remotely operated vehicle through electromagnetic waves. The system carried enough logic to receive and execute the remote commands. This *onboard intelligence* turned out to be one of the major aspect of what would become robotics systems. The first tentative of aerial robot is the Hewitt-Sperry Automatic Airplane (Figure 3.3b). It consists of a radio-operated aircraft with an



(a) Telautomaton: The first remotely operated vehicle, invented by Nikola Tesla, publicly demonstrated in 1898.

(b) Hewitt-Sperry Automatic Airplane in 1918, the first automatic flying system, developed before and during World War I.

Figure 1.1: Two examples of early partly-autonomous vehicles.

(a) A UAV performing a fully autonomous brick pick-and-place on a wall mock up, in the MBZIRC 2020 challenge (courtesy of [Lenz, 2020]).

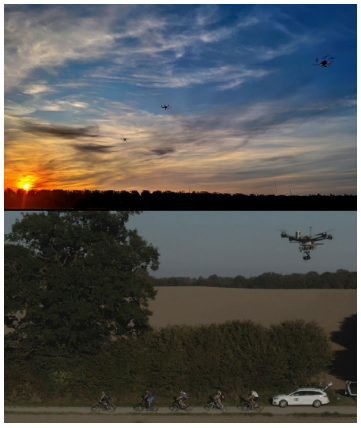
(b) A UAV entering the mine in the DARPA Subterranean Challenge (courtesy of [Rouček, 2019]).

Figure 1.2: Two UAVs in two international challenges on robotics.

onboard gyroscope-based autopilot. Over the course of the century, the technological improvements allowed the conception of more advanced robotic tools. The spur to robotic systems development is ascribed to the rise of machines for precise manufacturing in the 1960s, and in particular robotic arms. These machines were replicating the human behavior with a more rigorous – and less alienating – precision. There were no or very little appreciation of the surroundings. Therefore, such systems were confined to a defined task in a controlled environment. During the 1980s, robotics slowly came to be defined as the study of the *intelligent connection between perception and action* [Siciliano, 2008]. The paradigm consequently shifted toward the tripartite Perception-Decision-Action scheme: the robot assesses the environment, consequently plan a suited set of actions toward the accomplishment of its tasks, and execute those at best, acting on the environment. The embedding of onboard sensors and the suited *onboard intelligence* (i.e., embedded processing, mostly numerical and sometime analogical) allowed to use the robots in more complex, unknown or hazardous environments. Space rovers, for instance, drove this new paradigms, since remote control is unachievable. Nowadays, robotic systems with limited perception are still largely employed in industrial contexts [Siciliano, 2009], while the research community is mostly devoted to the conception of robust and adaptable algorithms or processes.

Over the past four decades, aerial robotics has faced an ever-increasing development [Feron, 2008], in parallel to other robotics research areas. It represents one of the main axes in robotics, with dedicated sections in major international conferences on robotics, and many specific conferences and journals. As other pieces of evidence of the interest toward aerial robotics, the *IEEE Robotics & Automation Society* holds a dedicated Technical Committee[1]. Besides, major challenges on robotics involve UAVs, sometime in collaboration with Uncrewed Ground Vehicles (UGVs). Figure 1.2 depicts a couple of UAVs in two prestigious challenges. Despite being largely employed in military domains, UAVs also got a lot of attention from the civil society. As a matter of fact, commercially available aerial drones have seen an unpredicted

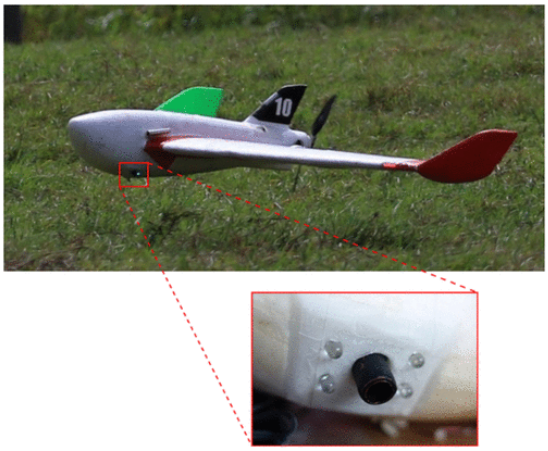
[1] https://www.ieee-ras.org/aerial-robotics-and-unmanned-aerial-vehicles



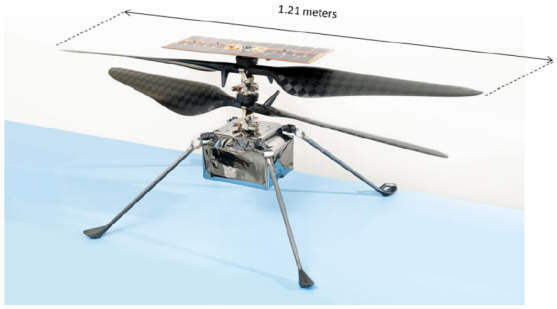
(a) Several UAVs employed for multi-view filming of a cycling event (courtesy of [Alcántara, 2020]).

(b) Transportation of medical equipment for post-disaster assistance in Japan (courtesy of [Yakushiji, 2020]).

(c) Historical monument visual monitoring with a UAV (courtesy of [Petracek, 2020]).



(d) Fixed wings UAV equipped with a sensor dedicated to cloud exploration (courtesy of [Hattenberger, 2022]).



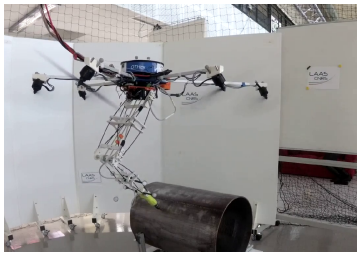
(e) NASA's Ingenuity helicopter, deployed on Mars in the scope of the Perseverance exploratory mission (courtesy of [Balaram, 2021]).

Figure 1.3: Examples of civil applications of UAVs.

outbreak over the past 10 years. The Federal Aviation Authority of the United States forecasted a rough estimate of 15,000 drones by 2020, in the USA only, as reported in The Economist[2]. This number is measured, as of May 2nd, 2022, to more than 850,000[3]. It covers both civil and commercial drones, since remotely controlled UAVs found a large audience for leisure and professional activities. In particular, those are largely employed for photography and video making, e.g. in the movie industry. The use of fully autonomous UAVs in field applications is still marginal, in part due to the tight administrative regulations regarding the deployment of such robots.

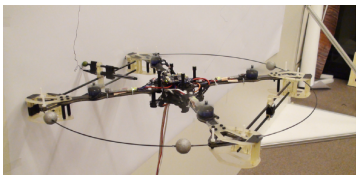
[2] https://www.economist.com/science-and-technology/2015/09/26/welcome-to-the-drone-age

[3] https://www.faa.gov/uas/resources/by_the_numbers/



(a) The OTHex ([Staub, 2018]) performing contact-based pipe default inspection (courtesy of [Tognon, 2018]).

(b) Dual-arm manipulator, developed in the scope of the Aeroarms project (courtesy of [Ollero, 2018]).



(c) A quadrotor used for contact inspection (courtesy of [Darivianakis, 2014]).

Figure 1.4: Examples of manipulation-oriented UAVs.

Many fully autonomous applications are however envisioned for UAVs, which can roughly be summed up in three domains, according to [Feron, 2008]:

- Remote sensing, including image acquisition for cinematography, field inspection, surveillance or search and rescue in hard-to-reach areas;
- Payload transportation, including cargo or person transportation, as well as goods delivery;
- Communications, e.g. as relay in disaster response, or as broadcast units.

Additionally, recent advancements paved the way for contact-based inspections and manipulations, again in hard-to-reach areas such as bridges or construction work sites, as reported in [Ollero, 2021]. Such manipulators can embed a rigid tool to perform simple tasks, or an articulated arm to perform applications that require more complex manipulation.

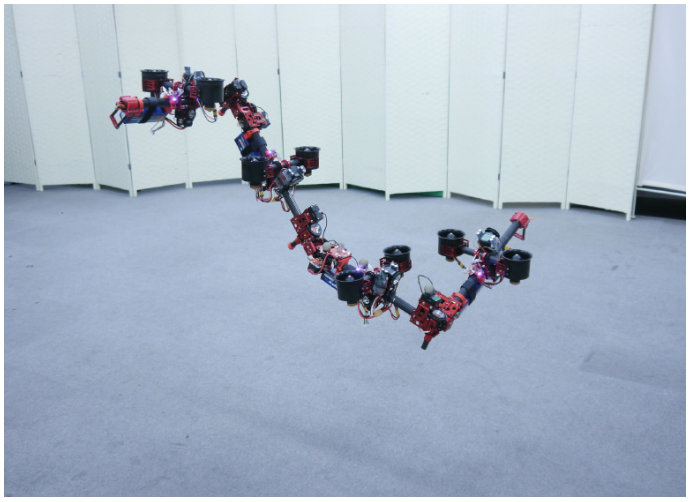
Yet, most of the currently deployed UAV-based applications are oriented toward monitoring and – to a lesser extent – cargo transportation. This stems again from the aforementioned strict regulations imposed onto UAVs. Some instances of UAVs involved in civil applications are reported in Figure 1.3. Many recent research projects are involved in the deployment of UAVs for manipulation in work environments. One could cite the French National Research Agency (ANR) The Flying Coworker[4], or European Commission H2020 Aeroarms[5] and Aerial-Core[6] projects. Figure 1.4 shows a couple of examples of manipulation-oriented applications of (rotary wings) UAVs.

In broad terms, three types of UAVs can be observed following a wing-based taxonomy:

1. *Fixed wings*,
2. *Rotary wings*,
3. and *Flapping wings*, inspired by birds or insects.

The latter type is the less commonly used, because of its complex mechanical design and very limited allowed payload [Xiao, 2021]. Fixed wings have historically been the

[4] https://www.laas.fr/projects/flying_coworker/
[5] https://aeroarms-project.eu/
[6] https://aerial-core.eu/



(a) DRAGON: A multi-body UAV with multiple actuated links, allowing a reconfigurable structure (courtesy of [Zhao, 2018]).

(b) LEONARDO: A bipedal robot equipped with two rotary wings (courtesy of [Kim, 2021]).

Figure 1.5: Two instances of complex robot design with rotary wings.

first types of powered UAVs to be developed and have got most of the attention during a large part of the 20$^{\text{th}}$ century [Newcome, 2004], partly because their similarity with classical airplanes allowed for an easier technology transfer. Those are suited for cruising flights and are appreciated for their relative energy efficiency. Rotary wings UAVs, also referred to as rotorcrafts, knew a slower spreading. Indeed, while helicopters have also been prototyped during the 19$^{\text{th}}$ century, engineers faced issues related to motor power and stability, which slowed the developments in the first half of the 20$^{\text{th}}$ century. With the technological improvements, smaller-scale helicopters and multi-rotors designs progressively appeared. They show poor energy efficiency compared to fixed wings aircrafts since most of the power-consumption is drained to compensate the gravity, whereas fixed wings UAVs can rely on air lift. However, rotorcrafts have several advantages with regard to down-scalability, flight agility, and the ability to perform Vertical Take-Off and Landing (VTOL) that provides very large usage flexibility. The high maneuverability of rotorcrafts – and their ability to hover in place – allows their use in indoor or cluttered environments, which oriented the focus of the aerial robotics community on those types of robots over the past decades. Additionally, rotorcrafts, and in particular quadrotors, have a relatively simple hardware conception and simple control policies, which have led to a massive spread of those for civil and leisure activities. The importance of this VTOL property also conducted to the production of hybrid designs, often referred to as *convertible UAVs* [Morin, 2015], which combine the fixed wings cruising flight with VTOL capabilities, such as tail-sitters. Some more original designs involving rotary wings have also been introduced to extend the locomotion capabilities of robots, such as a flying bipedal robot in [Kim, 2021], a complex multi-body AR where each link is actuated through rotors in [Zhao, 2018] (see Figure 1.5), or an aerial manipulator suspended to a cable-driven parallel robot in [Yiğit, 2021].

Apart from helicopters, which are also well studied in the literature [Ren, 2012], quadrotors and hexarotors with collinear propellers are the most common rotary wings UAVs. The collinearity of the rotors only allows for a unidirectional thrust, which simplifies both the design and the control law, but at the same time constrains their motion capabilities. In particular, the position and orientation cannot be controlled independently: the AR must tilt to move laterally, and conversely cannot tilt without inducing a lateral motion. These robots are called *underactuated*. Corollary, *fully-actuated* designs have emerged in the literature over the recent years [Hua, 2015; Rajappa, 2015; Hamandi, 2020], in which tilted positioning of the propellers allows to generate multi-directional thrusts, at the cost of a further reduced energy efficiency. These UAVs are able to decouple, to some extent, the position and orientation control, allowing, e.g., to move sideways while maintaining the same orientation (either flat or tilted). Thereby, the choice of UAV type is contingent on the task to perform, as a wide variety of designs are available, with their respective pros and cons.

In the scope of this thesis, the focus is set on small-scale, multi-rotor vehicles. The variety of available designs covered by this category allows for a large range of choices for specific applications. As previously mentioned, their capability to proceed to VTOL and their large maneuverability make them the de facto solution for cluttered environment scenarios; and this maneuverability can be exploited to comply with unexpected events occurring during the mission. Therefore, subsequent mentions of aerial systems pertain to this restrained definition. Moreover, since the stress is made on autonomous tasks, the denomination Aerial Robot is preferred over Uncrewed Aerial Vehicle in the whole manuscript.

1.2 Context and Objectives

The work conducted throughout this thesis takes place in the context of the French ANR project MuRoPhen (Multiple Robots for observing dynamical Phenomena) (MuRoPhen)[7], a research project devoted to a thorough investigation of the problem of monitoring a dynamic phenomenon with a team of sensing robots. The sensing robots are tasked to actively track the phenomenon which freely evolves in an uncontrolled environment. The robots have to autonomously control their own motion, ensuring both the stability of the system, as well as the reliable and safe accomplishment of the task in a cooperative fashion. The project aims for genericity, such that few assumptions are made on the nature of the phenomenon and its inherent motion – though it has to be reasonably bounded. The phenomenon is characterized as one or several points of interest that the robots need to detect and assess. This is in line with, but is not restricted to, several other projects conducted in the aerial research community, such as the dynamic filming of sports events [Zemas, 2017; Alcántara, 2020], the tracking or mapping of natural phenomena such as clouds or wildfire [Hattenberger, 2022; Bailon-Ruiz, 2022], and more generally the active positioning of mobile sensors for improved pose estimation [Varotto, 2022].

The sensing team composition is also generically defined as a collection of mobile robots, both aerial and ground, and possessing various motion capabilities. The sensors are of course required to be suited to the phenomenon of interest, but can

[7] https://www.murophen-project.eu/

also be heterogeneous in terms of sensing domain, size, weight, acquisition frequency, etc. The specific capabilities of each sensing agent need to be exploited at best by the system. A corollary aspect of this research project is thereby the problem of designing control strategies for such sensing teams, in order to achieve cooperation toward a common objective, by exploiting its redundancy, e.g., in terms of sensing. Indeed, this raises additional challenges related to, e.g., mutual localization, communication concerns, or collision avoidance inherent to Multi-Robots Systems (MRS).

The conception and implementation of such an observatory framework are thus challenging in many ways. Yet, the key concept that motivates the project and drives its realization is the tight entanglement of perception and control. The focus is thus oriented toward decision-making for objectives that are defined through a perception-driven semantic. It means that perception is not only a tool for the fulfillment of the task, but it defines the task itself. On the other hand, the decision-making process is required to leverage the action capabilities of the AR, that is to adapt its position and orientation in an agile fashion to account for unexpected events, while maintaining the overall stability and other motion constraints. This exploratory project proposes to deepen the knowledge on the tackled subjects. Its long-term objectives are the large-scale deployment of perception-oriented applications with collaborative ARs, in applications such as those presented in Figure 1.3.

For the scope of this thesis, the research problem is stated in more explicit terms as the conception (and implementation) of novel perception-aware control algorithms, for both single- and multi-robots systems. These control algorithms are dedicated to the monitoring of dynamic phenomena, and account for several motion-related limitations. The agility of the multi-rotor AR is exploited toward the realization of this task. These objectives are addressed through *optimal predictive control*, which are policies tailored to find optimal – or close to optimal – commands, based on a mathematical extrapolation of the current system and environment states. This is further presented and motivated in the subsequent chapters. A large place is also allocated to the practical implementation of the proposed control strategies. Therefore, all the produced software is empirically validated in simulations and on actual ARs. This work was conducted in the Robotics and InteractionS (RIS) team of the Laboratory for Analysis and Architecture of Systems of the French CNRS (LAAS-CNRS), and exploited the facilities therein to conduct the field experiments.

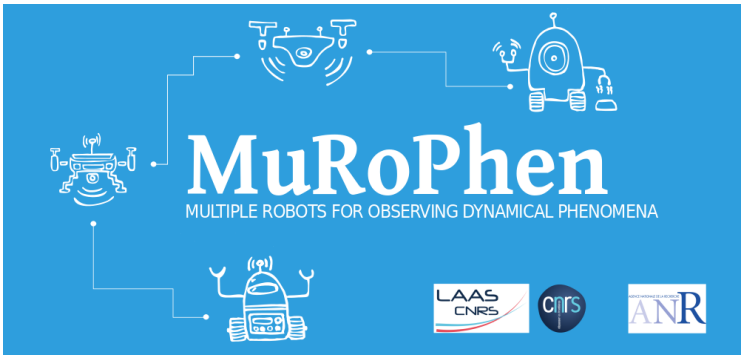


Figure 1.6: The MuRoPhen project logo.

1.3 Contributions

The previous sections corroborate the relevance of the problem of autonomously generating perception-driven motions in the field of aerial robotics research. However, as suggested by the very recent publications of works cited in Section 1.2, this field is still ongoing active developments from many research groups around the world. There are still many solutions to be explored. This thesis takes place in this context.

As previously mentioned, it addresses the problem under the spectrum of optimal predictive control. In particular, it makes use of the so-called Nonlinear Model Predictive Control (MPC) (N-MPC) approach, which will be reviewed in depth in Chapter 2. Building upon previous works, we formalize a perception-oriented N-MPC framework for the realization of perception-oriented tasks. This control framework accounts for the complete nonlinear model of the multi-rotor ARs, which have a generic design (i.e. various actuation and motion capabilities) and generic sensors, yet handled under a common paradigm. The controller produces the direct low-level inputs of the platform, namely the torque generated by each propeller. As much as possible, the controller handles both single- and multi-agent systems, with minimal changes between both formulations. In addition to the definition of this common formalism, some perception assessment criteria are proposed, and the controller is instantiated to comply with two practical applications of sensing AR:

- perception-based object pose estimation,
- visual-inertial self-localization.

Finally, the problem is also addressed under the spectrum of Active Information Acquisition (AIA), a branch of the motion planning literature that also tackles the generation of optimal sensing motions with fleets of ARs, through the use of elements of information theory and numerical optimization. This work proposes a synthesis of aspects from the AIA literature under the aforementioned formalism.

In each case, the various objectives and constraints imposed on the ARs are mathematically defined, in particular regarding the geometrical assessment of the visibility with respect to (w.r.t.) the sensor Field of View (FoV). Empirical validation is conducted through the realization of simulations and experiments with actual robots. The results are analyzed from a behavioral and a quantitative point of view. In the practical implementations, the perception is achieved using monocular cameras assigned to the detection of fiducial markers. This choice is motivated in Chapter 3, while alternatives are described in Chapter 2. More details on these simulations and experiments, and the used implementation, are reported in Appendices A and B.

The main achievements of this thesis are therefore the successful generation of perception-driven and constrained motions in various scenarios while maintaining the genericity of the framework. While previous pioneer works demonstrated the feasibility of such perception-aware N-MPC scheme, this thesis demonstrated the applicability to a broader class of problems through a more comprehensive formalism.

Additionally, collaborative works were conducted during this thesis, in the RIS team. The first two are the participation to the MBZIRC 2020 challenge with the LAAS-CNRS team, and a subsequent work on a generic vision-based control architecture for physical interaction. Thirdly, a practical application of the N-MPC

software developed along this thesis has been studied, in the scope of human-AR object handover. The first two are reported in Appendix D, while the third is presented in Section 5.8.

The work conducted throughout this thesis led to publications in international peer-reviewed journals and conferences. These are summarized in Table 1.1, along with a succinct description of the contributions.

1.4 Thesis Outline

The remaining of this manuscript is organized into three main parts. First, the preliminary part (Chapters 1 to 3) introduces the concepts which are used throughout the thesis. As the focus is set on multi-rotor ARs, a brief taxonomy of existing designs is first proposed. Then, a literature review is presented to analyze the works related to perception-oriented control policies in the aerial robotics community. It motivates the use of N-MPC, and presents possible alternatives. The chapter continues with a review of numerical optimization techniques, in order to enable a good understanding of the underlying mechanisms, and a review of existing N-MPC techniques for aerial robotics. Finally, the last chapter of Part I presents the mathematical modeling which will be used in the subsequent chapters, both for the generic ARs and the generic sensors. It concludes with rigorous modeling of the detection tool used in this work, i.e. the fiducial markers, as well as the presentation of the filtering applied to the low-frequency measurements.

Part II (Chapters 4 to 8), which constitutes the core of the manuscript, proposes an in-depth presentation of the thesis contributions. First, the control problem is formally stated and the choice of N-MPC is motivated. Then a generic formulation of the perception-oriented and actuation-constrained N-MPC is introduced, which is used in all the following chapters.

Subsequently, in Chapter 4, the actual perception-constrained N-MPC is defined, through a thorough geometrical expression of the objectives and constraints, in terms of the N-MPC state vector. Related experiments and simulations are proposed to validate the proposed framework. Finally, a practical implementation of such N-MPC is presented in the context of a human-robot handover of an object.

Chapter 5 addresses the definition of a perception-aware N-MPC in another context, that is the enforcement of a visual-inertial ego-localization of an AR. The underlying concepts are recalled, and a practical implementation fulfilling the minimal criteria for interfacing with the N-MPC is proposed. Then, the previously introduced N-MPC is modified to comply with the new requirements of the task. Finally, experiments and simulations are again proposed.

The frameworks presented in these two chapters are then extended to the scope of Multi-Robots Systems (MRS) in Chapter 7. The various considerations regarding the N-MPC formulations when scaling up the number of agents are discussed, with a focus on computation decentralization. A review of the existing techniques for collision avoidance among agents in the related literature is proposed.

Finally, Chapter 8 is dedicated to the definition of an Active Information Acquisition (AIA) problem for object pose estimation, and the proposal of a N-MPC-based

solution, building upon the previously introduced solutions. In particular, this framework allows for optimal sensing motion generation without any exogenous position reference. It handles multiple agents, based on the considerations raised in the previous chapter. An extensive literature review on AIA is proposed, then validation in simulations and experiments is proposed once the N-MPC is properly defined. This validation demonstrates how the same mathematical formulation allows the emergence of collaborative behaviors when more sensing capabilities are added to the system, i.e. more sensing agents are introduced, or more actuation capabilities are allowed by the AR design.

Throughout Part II, the videos of all the presented experiments and simulations are accessible on the LAAS-CNRS PeerTube instance, through a clickable link in the images. A QR code is also available to access the link from the printed version. For convenience, the videos are also accessible from a dedicated playlist[8].

Finally, in Part III an overall conclusion is proposed. It summarizes the previously mentioned achievements and limitations. Then, taking a step back, we provide a general conclusive word on the applicability of such N-MPC frameworks for the desired tasks, including an overview of the remaining challenges to be addressed and perspectives opened by this thesis.

A handful of appendices are proposed to discuss the more practical aspects of the thesis. The first two, are dedicated respectively to the experimental setup and the practical N-MPC implementation. Then, an estimation procedure is proposed to refine the modeling of the platform. Finally, some collaborative works that are not directly linked to the core of the thesis are presented, conducted in the scope of a robotic competition and an ensuing publication on vision-based physical manipulation.

[8] https://peertube.laas.fr/videos/watch/playlist/26ed37df-bf69-42d7-b786-2775fedebc02

[Jacquet, 2020] Published in *ICRA'20*	• Preliminary work on perception-constrained N-MPC • Fictitious extension of N-MPC state for efficiency • Simulated camera with circular FoV • Numerical simulations and off-board implementation • Assessed results on exploiting the full actuation span of the AR to comply with the tasks
[Jacquet, 2021] Published in *RAL* presented at *ICRA'21*	• Use of quaternion representation of orientation to overcome singularities • Removal of extra state variables • Extension to rectangular FoV • Gazebo simulations and real experiments with on-board implementation and sensors
[Jacquet, 2022a] Published in *RAL* presented at *ICRA'22*	• Formalization of N-MPC for solving AIA problems • Generation of perception-driven motions without exogenous position references • Decentralized handling of MRS yielding emergent collaborative behavior • Experiments with 1 and 2 ARs
[Jacquet, 2022b] Published in *IROS'22*	• Exploitation of perception-constrained N-MPC to enforce visual state estimation of an AR • Definition of suited objectives and constraints • Summary of conditions on the state estimator to interface it with the N-MPC • Gazebo simulations and real experiments
[Corsini, 2021] Published in *AIRPHARO'21*	• Definition of a generic control architecture for vision-based physical interactions with UAVs • Personal participation: – Integration of the Visual Servoing (VS) scheme in the control framework – Software implementation of the VS – Co-handling of the simulations and experiments – Co-writing of the manuscript
[Corsini, 2022] Published in *IROS'22*	• Human-Aerial Robot handover of an object • Personal participation: – Implementation of the perception objectives and constraints in the N-MPC and the detection algorithm – Assistance for simulations – Co-writing of the manuscript

Table 1.1: Summary of peer-reviewed publications produced during this thesis.

Chapter 2

Perception with Aerial Robots: A Literature Review

Contents

2.1 Introduction **16**
2.2 Multi-Rotor Designs **16**
2.2.1 Collinear Designs 16
2.2.2 Tilted Designs 17
2.3 Exteroceptive Perception with ARs **19**
2.3.1 Onboard sensors 19
2.3.2 Vision-Based Localization 20
2.3.3 Onboard Computer Vision 21
2.4 Vision-Based Motion Generation **23**
2.5 N-MPC for Aerial Robots **25**
2.5.1 Optimal Control Problems 26
2.5.2 Nonlinear MPC 27
2.5.3 Real-Time Iteration 29
2.5.4 Instances of Nonlinear MPC for Aerial Robotics 29
2.6 Perception Aware N-MPC **31**

2.1 Introduction

This chapter presents an overview of the literature regarding the use of perception with UAVs. After a brief presentation of the commonly used multi-rotor AR designs, we will recall the uses of onboard perception with AR, going through the most common usage and the various limitations and challenges faced in aerial robotics. In particular, we propose a review of the classical perception-based control strategies. The last part of the chapter highlights the control strategy upon which this thesis focuses, namely N-MPC.

2.2 Multi-Rotor Designs

2.2.1 Collinear Designs

As briefly alluded to in Chapter 1, the multi-rotor ARs can be designed in numerous ways. A comprehensive allocation-based taxonomy is proposed in [Hamandi, 2021], we therefore refer to this article and references therein for further details. While unirotors and birotors exist and are studied in the literature, they are less common and face numerous stability issues. Similarly, trirotors are studied, with the emphasis put on fast maneuvering, due to the larger yaw control authority allowed by this design. However, they suffer from an unbalanced moment from the odd number of propellers, and poor hovering stability [Kataoka, 2011]. As a matter of fact, 4 rotors are the minimum to ensure the capability to perform stable hovering (i.e., counteract gravity with zero average moment) with collinear rotors. With less than 4 rotors, a tilting angle is required to nullify the moment.

The most common is a design with with 4, 6 or sometimes 8 collinear propellers, which are coplanar and evenly spaced around the geometrical center of the robot. This design is favored for several reasons. The collinearity of the propellers enhances the energy efficiency of the platform, since the propeller thrust work is fully employed toward the motion. The coplanar and evenly spaced propellers increase the simplicity of the mechanical design and control law. Quadrotors capture most of these advantages. They are nowadays very well understood, thanks to the vast literature on their study, among which one could cite [Pounds, 2010; Mahony, 2012; Powers, 2015]. The reduced number of propellers further simplifies the mechanical design, and reduces the overall weight of the platform, such that micro lightweight quadrotors emerged (e.g., a 45 g quadrotor presented in [Zhang, 2015]). The relative easiness of the control of collinear quadrotors is induced by a property called *differential flatness* [Fliess, 1995], which implies the existence of a subset of outputs (and their derivatives) that describes a full nonlinear system through an algebraic relation. It allows for the exact linearization of some nonlinear systems. Quadrotors satisfy this property [Mistler, 2001], its flat outputs being the 3D position and the yaw angle. Collinear hexarotors and octorotors share the same properties, but the addition of extra actuators allows for increased payload and redundancy which can be exploited to overcome rotor failure. Indeed, in case of defect of one rotor, an hexarotor can roughly be controlled as a quadrotor (by shutting down the rotor opposite to the defective one to ensure the moment stability of the AR). Such robustness of collinear

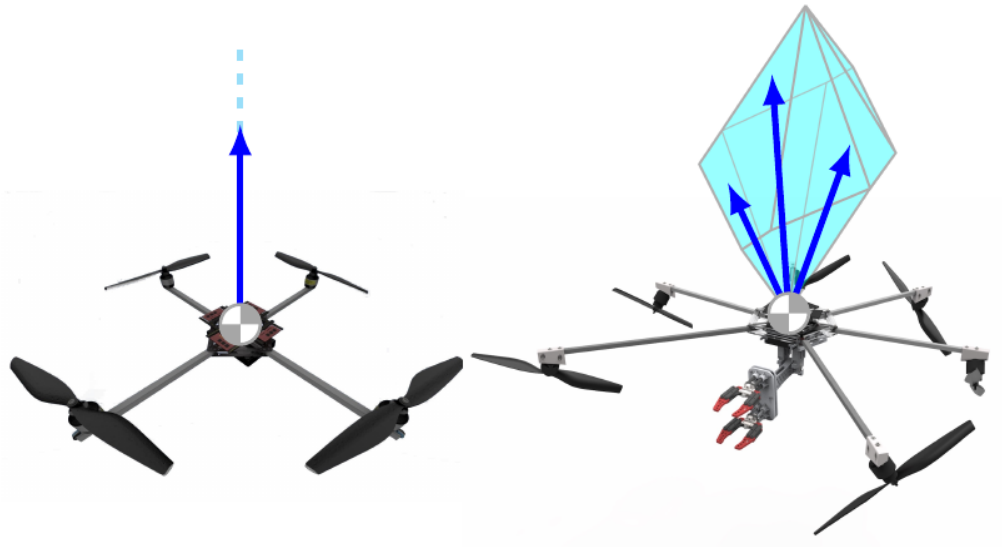


Figure 2.1: The thrust directions allowed for a collinear quadrotor (left) and a tilted-propeller hexarotor, the OTHex (right) (courtesy of [Staub, 2018]).

hexarotor to rotor failure is studied in [Baskaya, 2021], considering the spacing of the propellers around the AR center.

2.2.2 Tilted Designs

Collinear designs however face a common limitation due to the uni-directionally exerted thrust. It induces a strong coupling between the position and orientation control, as mentioned in Chapter 1. The AR needs to tilt to move laterally, and vice-versa. Corollary, the AR possesses only one stable orientation for hovering, with the propeller axis collinear to the gravity vector. This underactuation thus greatly limits the use of such platforms for precise manipulation or within cluttered environments. The yaw control authority is also limited by these designs, as this rotation is only induced by the discrepancy between Clockwise (CW) and Counter-Clockwise (CCW) propeller drag torques, which have to operate in pairs to maintain the position stability. The rotation rate is therefore limited.

To palliate these limitations, tilted-propeller designs have been developed by the aerial robotics community [Hua, 2015; Rajappa, 2015; Hamandi, 2020]. This allows to fully or partially decouple the position and orientation control of the AR. The resulting force exerted by the propellers is therefore no longer uni-directional, but rather fits in a cone or, more rarely, a cylinder. This leads to a thrust-direction-based taxonomy of the multi-rotors: collinear designs are called uni-directional, in opposition to the multi-directional ones. In particular, if the resulting force is constrained in a cone, the thrust is described as laterally-bounded, and as omni-directional if it can be exerted in any direction (i.e., in a cylinder). Figure 2.1 illustrates two cases of uni-directional and laterally ARs. For quadrotors, tilting the propellers radially, i.e. around the axes of the arms, allows heightening the yaw control authority, as in [Falanga, 2017] in which the authors claim that a 15° radial tilting multiples it by three. This is of course achieved by a loss of maximum collective vertical thrust,

(a) A V-tail quadrotor, with two tilted propellers (courtesy of [Ciarfuglia, 2014]).

(b) The Tilt-Hex, a tilted-propeller hexarotor developed in the LAAS-CNRS (courtesy of [Bicego, 2020]).

(c) The Omni7+, an omni-directional heptarotor (courtesy of [Hamandi, 2020]).

Figure 2.2: Three examples of non-collinear propeller Aerial Robots.

as the independent thrust no longer works entirely along that direction. For this 15° radial tilting, this loss is of $(1 - \cos(15°)) \approx 3\%$. Online controllable radial tilting of quadrotor propellers has also been proposed, e.g. in [Ryll, 2015], in which four servomotors are added along these arms to actively, and individually, tilt the propellers. These four extra Degree of Freedom (DoF) enable the full actuation of the robot. Yet, such highly coupled systems are challenging to control.

Tangential tilting of the quadrotor propellers (i.e. inward or outward) is also employed. To maintain the moment balance while allowing a larger control authority along the two horizontal axes, the rotation directions are not paired for opposite propellers, as done in most designs, but rather grouped on the same side of the quadrotor body. Similarly, controllable servomotors have been added on quadrotors to perform an online reconfiguration of the tangential tilting, e.g. in [Badr, 2016].

Indeed, these two tiltings have been combined. The first quadrotor of this kind has been proposed in [Şenkul, 2014], in which the propellers can be individually tilted in both directions. In [Hua, 2015], a similar approach is tackled, but the four propellers are tilted simultaneously in the same direction. In both cases, the position and orientation are fully decoupled. However, due to the complex design of the tilting mechanisms, these two approaches have only been tested in numerical simulations.

Some other "hybrid" quadrotors designs are proposed, such as the *V-tail* [Ciarfuglia, 2014], which possesses two tilted propellers on one side, in V-shape, see Figure 2.2a. Such a design is very well suited for agile maneuvering.

Tilted-propeller hexarotors are widely used for their fully-actuated capabilities at reduced mechanical complexity. This class of ARs is well studied by the recent literature [Franchi, 2018]. Figure 2.2b proposes an instance of such AR. A sufficient condition for the full actuation is stated in [Michieletto, 2018] as the fact of having non-zero tangential tilting, even without any radial tilting. Yet, it is achieved at the cost of energy efficiency, such that a trade-off is performed in the choice of the tilting angles between the maximum vertical thrust and the angle of the thrust cone. Indeed, in the recently developed platform at LAAS-CNRS (which is briefly exposed in Appendix D.1, see Figure D.2), the tangential titling is set to 20°, with no radial tilting. Again, reconfiguration designs have emerged to allow online control of the tilting angles. In [Ryll, 2016], such a platform is proposed, with a single

motor controlling simultaneously the tilting angle of all the propellers. Interestingly, in [Bicego, 2020], the choice of the tilting angle (i.e., the control of that extra motor) is delegated to the optimal controller, which is tasked, among other objectives, to optimize the energy efficiency of the platform. The propellers are thus tilted only when the full actuation is needed for the desired motion.

Tilted-propeller hexarotors, due to their full actuation and large payload, are favored in the scope of aerial physical interactions in numerous works, e.g. [Ryll, 2017; Staub, 2018; Trujillo, 2019; Tognon, 2019]. Employing fully-actuated robots in this context allowed the emergence of a new interaction paradigm in which the end-effector is rigidly attached to the AR, and its 6D pose is fully handled by maneuvering the hexarotor body. This allows to remove the heavy and bulky articulated arms from the aerial platforms, thus increasing the flight duration. This paradigm is presented slightly further in Appendix D.2.

Having more than six tilted propellers allows for omni-directionality of the robot. Indeed, as proven in [Tognon, 2017], seven actuators are necessary to gain such a property. Octorotors with non-coplanar rotors are sometimes employed as omni-directional AR, but are less convenient to exploit due to the few possible places for attaching an end-effector. Such platforms are also usually bulkier. In [Yiğit, 2021], a solution is proposed to attach the AR to a suspended cable, greatly increasing the energy efficiency. These novel platforms might be exploited for manipulation in semi-controlled environments, e.g. where a crane can be installed to suspend the cable.

This short taxonomy of existing multi-rotor designs is proposed to motivate the use, in the scope of this thesis, of a generic model that encompasses several, if not all, of the aforementioned designs. Given the large creativity of the aerial robotics community, we believe in the importance of proposing unified control strategies that are transposable among robots at a minimal effort, avoiding the pitfall of solving problems in a too-specific fashion. Henceforth, the remainder of the manuscript will focus on the genericity of the proposed methods. To this end, we will leverage a generic multi-rotor model called the Generically Tilted Multi-Rotor (GTMR), whose details are given in Section 3.4

2.3 Exteroceptive Perception with ARs

2.3.1 Onboard sensors

This section proposes to review the common applications of exteroceptive perception in aerial robotics. As mentioned in Chapter 1, the use of such sensors widely spread in robotics in order to retrieve semantic knowledge about the environment. Due to the limited payload of multi-rotors, the miniaturization of sensors was crucial for perception in aerial robotics. This is the main reason why small-scale and lightweight monocular cameras are by far the most commonly used sensors on ARs, whereas ground robots have been equipped with heavy lidars for numerous years. However, smaller lidars are nowadays installed on aerial robots, for instance in [Mohta, 2018] for environment mapping. Therein, a workaround is provided by equipping a lighter 2D lidars on an actuated gimbal, whose nodding motion allows to perform 3D mapping

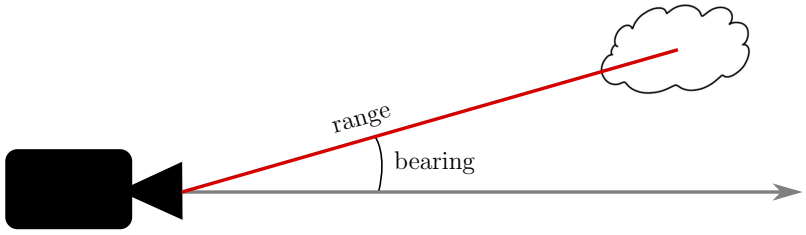


Figure 2.3: Schematic view of a range-and-bearing sensor.

of the environment. As another evidence of the increasing use of lidars for ARs, a recent comparative review of existing lidar-based Simultaneous Localization And Mapping (SLAM) algorithms is proposed in [Milijas, 2021].

Similar to lidars, stereo cameras are not commonly employed in aerial robotics. It goes the same for most depth sensors, with the notable exception of RGB + Depth (RGBD) infrared cameras, which recently became very small and lightweight, and thus started to be largely employed on ARs.

These sensors are called *range-and-bearing* sensors, since they provide both a 2D angular information and a 1D depth information (see Figure 2.3). They are in particular opposed to *bearing-only* sensors (e.g. monocular cameras), or, more rarely, *range-only* sensors (e.g. punctual lidars). This latter type is sometimes employed as altimeters, which provide useful information for onboard state estimation.

Since the monocular camera are bearing-only sensors, the literature provides a large choice of techniques to retrieve the missing depth information. In the context of object monitoring, the pose estimation can be enhanced using either some prior geometric information of the object [Thomas, 2017] or some deep learning-based algorithm [Wofk, 2019]. Mapping of the surrounding or 3D reconstruction can also be achieved with monocular cameras, using successive views or Structure from Motion (SfM) [Lundberg, 2018; Rodrigues, 2020].

Recently, event-based cameras also have been employed in robotics. Their very low latency is exploited mainly for state estimation when handling very agile maneuvering, as in [Kueng, 2016], which builds upon recently introduced event-based corner detection to introduce a dedicated Visual-Inertial Odometry (VIO) algorithm. Another usage of those is object detection and tracking [Mitrokhin, 2018], but such activity is rarely handled with agile maneuvering, hence standard monocular cameras are often preferred.

2.3.2 Vision-Based Localization

Among perception-related activities tackled in aerial robotics, one can roughly distinguish two categories:

- *localization* activities, which exploits exteroceptive sensors for estimating the AR own state;
- *monitoring* activities, which consist of, e.g., exploration, phenomenon detection, localization and tracking, or object 3D pose reconstruction.

Onboard vision-based localization has been an active field of study for decades, resulting nowadays in very robust and efficient off-the-shelf software. To cite a few examples, ROVIO [Bloesch, 2015], VINS-Mono [Qin, 2018] and ORB-SLAM (whose latest version is proposed in [Campos, 2021]) are among popular frameworks in aerial robotics. The first is based on Extended Kalman Filter (EKF), while the other two rely on nonlinear optimization to obtain a Maximum A Posteriori (MAP) estimate of the relative displacement across several frames. In addition, these software are proposed open-source and are actively maintained by their developers and communities.

Such software rely on feature detection. A computer vision algorithm is employed to retrieve points of interest in the images, which are identified and tracked through successive frames It allows to estimate the relative displacement of the camera, hence of the robot. Such features are well established visual descriptors, often targeting object corner detection. Namely, ROVIO is based on the FAST descriptors introduced in [Rosten, 2008], while VINS-Mono uses the robust descriptors from [Shi, 1994]. ORB-SLAM is, as the name states, based on the ORB descriptors [Rublee, 2011]. All the aforementioned software are exploiting inertial information in addition to visual cues. This allows to refine the estimation and to palliate the low frequency of camera devices. However, if using of inertial data is becoming the de facto standard, this is not always the case. The first version of ORB-SLAM [Mur-Artal, 2015] is a Visual-only SLAM, and the latest version in [Campos, 2021] still proposes a Visual-only mode.

A thorough evaluation of VIO software has been proposed in [Delmerico, 2018]. It concludes that there is no prominent solution among the one analyzed, and that the choice of a VIO software is therefore dependent on the available resources and task requirements. Yet, some tendencies can be observed. For instance, VINS-Mono seems to provide reliable performances, yet at a high computational cost. ORB-SLAM is claimed to outperform VINS-Mono in [Campos, 2021], yet further tests are required to assess its practicality of use and computational requirements.

2.3.3 Onboard Computer Vision

Another use of sensors is the detection of objects of interest, mostly in exploratory activities or for tracking a given dynamic phenomenon. It implies retrieving semantic information from the sensor raw data. Among exploratory applications, a famous instance is search and rescue, in which an AR or a fleet of ARs is employed to localize missing people after natural disasters or in hard-to-access environments. A recent survey on this class of applications is proposed in [Queralta, 2020], which reviews the state of the art of perception techniques for such applications. As expected, this survey shows that Convolutional Neural Network (CNN)-based detection seems to be most widely employed, as in most of image processing fields. Indeed, recent technological and algorithmic development allowed the appearance of CNN suited for small-scale computers that can be equipped on ARs. The autonomous cinematography framework alluded in Chapter 1, presented in [Alcántara, 2020], is for instance based on CNN, through the lightweight detection and tracking pipeline proposed in [Nousi, 2019].

The "miniaturization" of CNN is an ongoing trend in the computer vision community, with dedicated challenges and workshops in major conferences, for instance *ECCV'18* [Zhu, 2018]. A conclusive instance is SlimYolov3 [Zhang, 2019], based on the well-established You Only Look Once (YOLO) algorithm [Redmon, 2016], which provides very competitive performances at a reduced computational cost. While the focus is largely set on monocular vision, there also exist similar techniques for multi-modal detection, involving for instance lidar or RGBD cameras, as discussed in [Queralta, 2020]. Aerial applications are enabled by recent small-scale onboard GPUs, such as the Nvidia Jetson computers[1].

In the scope of object detection and tracking with bearing-only sensors, image-based tracking strategies exist, but do not cover the majority of the literature. Detecting the object in the 2D data is often only the first step, and depth estimation is conducted. As stated in Section 2.3.1, this can be achieved either via prior geometrical knowledge, neural network approaches, or through temporal processing of data flow. In [Wofk, 2019], an efficient CNN is used to extrapolate depth information from single images. The algorithm is computationally light, reaching up to 178 fps on a Jeston TX2 GPU, which is 3 time faster than the data acquisition of most cameras. A different solution is proposed in [Rodrigues, 2020], based on successive frames analysis to converge toward an accurate depth estimation. The model-based approach is proven to converge under some conditions on the sensing robot actuation. In [Thomas, 2017], a geometric algorithm is used, based on the known size of the tracked ball. This extra information allows to assess the 3D pose of the ball in the camera frame, therefore to proceed to active tracking.

Another well known geometric approach is the use of Perspective-n-Points (PnP) algorithms, such as [Lepetit, 2009; Collins, 2014], which reconstructs the 6D transform between two frames, from n points whose coordinates are known. Prior knowledge of the object model is required, to assess the object frame coordinates of the n detected feature points (typically the object corners). Thereby, the 6D pose of the object in camera frame can be obtained by computing the rigid body transform that minimizes the reprojection errors between the pixel measurements in the object coordinates (through an analytical pseudo-inversion or numerical optimization). The minimum value of n depends on the algorithm, but is usually set to $n = 4$ points to avoid singularities related to coplanarity [Lepetit, 2009], thus enabling the common use of square markers. Some simplifications allow to decrease this minimum down to $n = 3$ [Gao, 2003] or $n = 2$ [DAlfonso, 2013]. More details on the use of such markers are given in Section 3.6.

While the detection of such markers is very efficient, there exists a known theoretical limitation for its use in tracking activities. There is an ambiguity in the orientation detection of these markers, i.e. there can be more than one plausible pose for a given observation. This problem is prominent when the camera in frontoparallel to the marker, i.e. that $\mathbf{z}_M$ is normal to the marker plane. This is characterized by the fact that two solutions to the PnP problem provide similar reprojection errors. An example of such a scenario is depicted in Figure 2.4, in which the two poses on the left (yellow and blue) are almost as likely. They have respectively reprojection errors of 1.1e−4 and 1.3e−4. In successive detections, the most relevant pose returned by PnP might "switch" between the two solutions, thus preventing an accurate tracking

[1] https://en.wikipedia.org/wiki/Nvidia_Jetson

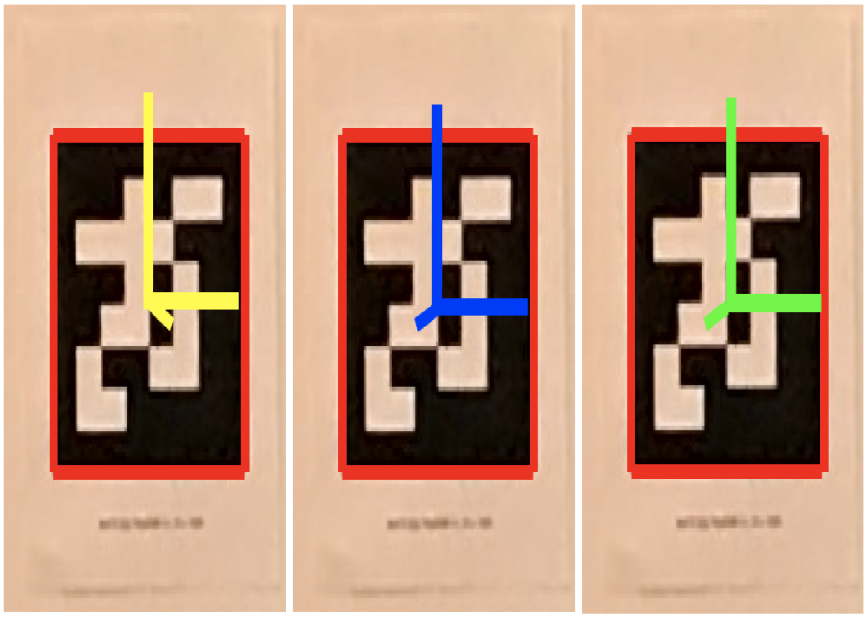


Figure 2.4: Two plausible PnP solutions that describe the detected marker are depicted as the yellow and blue frames, along with the green ground truth (courtesy of [Chng, 2020]).

of the object orientation. An algorithm such as IPPE [Collins, 2014], one of the recent standards for PnP, generally outputs both plausible solutions for posterior filtering, as in [Chng, 2020].

PnP algorithms are useful since there exist some techniques to propagate a prior measurement uncertainty, thus providing an accurate estimate of the object 6D pose uncertainty, useful to perform filtering (e.g. with a KF). Suited propagation schemes are presented in Section 3.6. It can be used either with a heuristic uncertainty assumption (i.e. an arbitrary isotropic pixel noise), but some recent CNN algorithms provide an uncertainty estimation along with the pixel detections. In [Tremblay, 2018], a 6D pose estimator is proposed that combines feature extraction and PnP. The belief map of the 2D pixels being estimated, a suited propagation would allow to provide an estimate of the overall pose estimation uncertainty.

All of the aforementioned methods require a prior calibration of the camera (both intrinsic and extrinsic). There exist several methods to proceed to this calibration, some of which are implemented in the well-known OpenCV library [Bradski, 2000]. A unified camera/Inertial Measurement Unit (IMU) calibration algorithm is proposed in the *Kalibr* library, introduced in [Furgale, 2013].

2.4 Vision-Based Motion Generation

Once semantic knowledge of the environment is retrieved from the sensors, a suited control strategy can generate the motion toward the fulfillment of the task. Indeed, incorporating visual information directly into the control loop has become a popular approach in robotics, in order to design reactive control for tasks that are visually guided (positioning, grasping, physically interacting, avoiding obstacles...). Over the years, it has gradually become a paradigm called *vision-based control*.

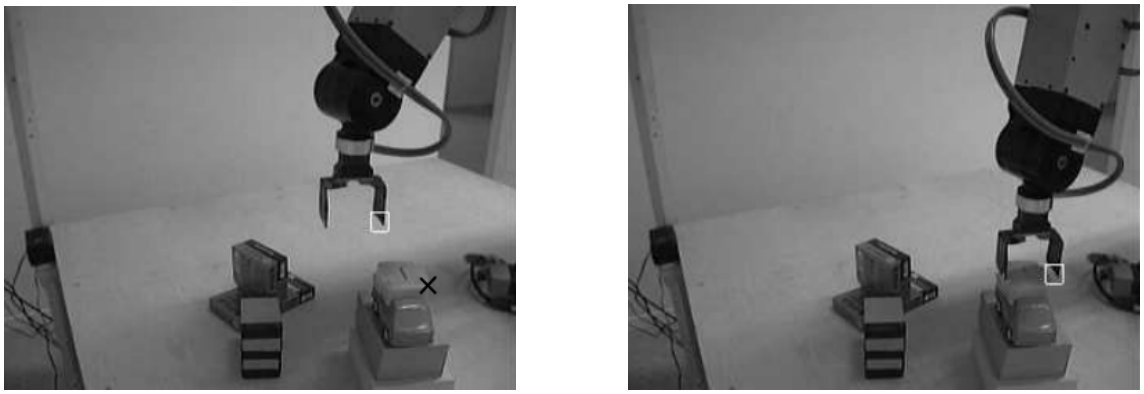


Figure 2.5: Example images of IBVS. The task is to bring the end effector onto its goal, denoted respectively with the white square and black cross (courtesy of [Kragic, 2002]).

The most famous and well-studied technique in this field is Visual Servoing (VS). It consists of designing a control law (usually controlling at velocity-level) aiming at nullifying a vision-defined error. A first proposal for defining this error is to state it directly in the image plane of the camera, and is thus called Image-Based Visual Servoing (IBVS). Roughly speaking, IBVS proposes to position the camera w.r.t. a reference image. It allows to solve the problem without relying on a 3D pose estimation, which is not de facto given with a monocular camera, as previously stated. However, it is considered a more difficult technique due to the complexity of the kinematic relationship between the image features and the motion of the camera. It is indeed subject to local minima and singularities, as explained in [Chaumette, 1998]. Another criticism that is addressed in [Corke, 2001] is the sensitivity to large rotations, which lead to wide "retreat" motion of the camera, which moves backward before converging in front of its goal. Despite these drawbacks, IBVS has been widely employed in industrial contexts, as recalled in [Kragic, 2002], a survey on the topic.

Conversely, Position-Based Visual Servoing (PBVS) is an approach that estimates the 3D pose of the features in the Cartesian camera frame and minimizes the tracking error w.r.t. a reference position. It requires to retrieve the depth information, usually through a model-based geometric approach. PBVS is also sensitive to image measurement and kinematic model errors, and [Kragic, 2002] claims that it requires a precise extrinsic camera calibration, while IBVS are more robust to such imprecisions. However, the stability analysis for PBVS schemes is generally simpler and has a larger domain of attraction. Finally, PBVS is however more convenient to use from a user point of view, since the reference pose is typically easier to define than in IBVS, and the produced trajectory is more legible (e.g., a straight line in 3D).

An intermediate solution is offered by Hybrid Visual Servoing (HVS), sometimes referred to as 2.5D VS [Malis, 1999; Conticelli, 1999], which use features both from the image space and the Cartesian space. These are less sensitive to calibration uncertainty than PBVS and can rely on partial pose estimation [Chaumette, 2007], and are more robust to singularities than IBVS schemes. The definition of such a scheme in a pick and place activity is detailed in Appendix D.2.

These approaches are suited for industrial contexts where the robot dynamics can be abstracted, and where a velocity-level planning is applicable. Yet, these approaches are not suited for systems with fast dynamics such as ARs, in which

the coupling between robot and camera motions is more complex. Indeed, and in particular with underactuated ARs, the coupling between orientation and position cannot be easily put aside in the scope of vision-based control. The sensing AR is de facto an *eye-in-hand* device, i.e. the camera moves along with the robot, and this needs to be considered in the control to prevent any loss of visibility. This can be done by arbitrarily limiting the rotational motion, but this is not desirable since it prevents the AR to exploit its full actuation. Then, the computed trajectory needs to account for these limitations. Henceforth, it is desirable to use constrained controllers. Indeed, a recent field of study emerged for vision-based control strategies based on optimal rather than geometrical control.

An optimization-based VS scheme is introduced in [Sheckells, 2016]. The first step consists of computing the desired 6D pose w.r.t. to the object using a PnP algorithm. This allows to initialize the problem as in IBVS, with desired image coordinates, but expresses the destination in the Cartesian space, avoiding the aforementioned limitations of IBVS. Then, an optimal trajectory is generated by exploiting differential-flatness and accounting for image feature tracking. The PnP exploits the depth information through a SfM scheme.

A vision-based trajectory generation for flight through narrow gaps is proposed in [Falanga, 2017]. Therein, the approaching phase of the trajectory is performed in a two-step optimization problem. First, the desired orientation of the camera throughout the motion is computed using constrained optimization. From there, a second optimization is conducted to select a minimum-length trajectory that brings the AR in front of the gap to cross, while satisfying the visibility during the trajectory.

In [Spasojevic, 2020], a minimum-time trajectory generation is proposed, which handles visibility constraints. This is achieved by a suited path parameterization, allowing to formalize a convex optimization problem. Recently, [Mao, 2022] also proposes a Quadratic programming (QP) optimization-based scheme to generate visibility-aware trajectories, under visibility and actuation constraints. An online replanning strategy is proposed to palliate the errors accumulated while moving. Another similar optimal planner is introduced in [Tordesillas, 2022] in the scope of vision-based obstacle avoidance. The approach is built in two steps: first, a graph-based search algorithm is used to provide an initial guess of the trajectory under collision constraints. Then, an optimization is performed to modify this prior guess accounting for visibility constraints.

These recent works have as common trait to generate motions that account for visibility constraints and objectives. Additionally, recent works focus on proposing optimal control strategies that account for similar constraints, often using N-MPC. These works will be the focus of the upcoming sections. Such controllers – or trajectory planners – are often qualified as “perception-aware”, after an establishing work in this field conducted in [Falanga, 2018].

2.5 N-MPC for Aerial Robots

Before presenting the aforementioned perception-aware N-MPC controllers, we propose a review of the use of N-MPC in the scope of aerial robotics. This section first recalls the underlying concepts of optimal control, then proposes a presentation of

the (N)-MPC approach, including an overview of the RTI concept. Finally, various instances of N-MPC applications in aerial robotics are presented. A survey of available N-MPC software suites is proposed in Section B.

2.5.1 Optimal Control Problems

Optimization is a branch of applied mathematics which focuses on the minimization (or maximization) of a given function on a given domain. In robotics, optimal control has been studied for decades to solve problems which cannot be analytically resolved. Indeed, if linear systems often admit an analytical solution, general systems subject to nonlinear dynamics or constraints do not. Therefore, it is common to compute a so-called "optimal" solution which minimizes a given performance criterion, through an optimization process. This criterion is called an *objective* or *cost* function, a function of the system state and inputs with scalar value. Moreover, some constraints on the allowed values for the state and input variables can be included in the scheme. The constrained optimization problem can then be solved under the Karush-Kuhn-Tucker (KKT) conditions [Boyd, 2004].

Optimal control and optimal trajectory generation are based on such techniques. The former consist of generating the sequence of system inputs that minimizes the objective function while complying with a certain control law; while the latter is rather the generation of successive control points that yield the minimum cost. It is also common to minimize the time taken to complete this trajectory. They are two similar instances of optimization problems, which comply with the same optimality criterion, namely Pontryagin's maximum principle, which states some conditions for the optimality of the solution. We refer to, e.g., [Bertsekas, 2012] for further details on this principle.

Optimal Control Problems (OCPs) are often formulated in a non-myopic (or non-greedy) fashion, meaning that it does not only consider the current instant, but rather a future time window, called the time horizon or receding horizon, in which the system evolution is extrapolated based on a model of the system dynamics.

There exist two classes of methods to solve optimization problems, the *direct* and *indirect* ones. Indirect methods consist of rephrasing the minimization problem as the integration of a differential equation which is subject to the same set of constraints. It reflects the classical approach to optimization having been employed before the use of computers for numerical resolutions. Such methods are also known as *first optimize, then discretize*. The reformulated problem (called a Boundary-Value Problem (BVP)) is often nonlinear and has several local optima. Finding the global minimum is in general hard, since it would require performing an extensive search amongst all local minima. Moreover, the differential equation is often complex to integrate as it is. When the optimal problem can be formulated in an indirect fashion, a compact global optimal solution to the problem can be retrieved This is the case in some specific problems which satisfy enough hypotheses to simplify the BVP.

The indirect approach is partly disregarded in recent solvers. On the contrary, the direct approach, also known as *first discretize, then optimize*, consist of solving the minimization problem directly, i.e. numerically computing the minimum of the objective function that satisfies the constraints. The (continuous) minimization

problem is discretized in terms of all the state and input variables, thus transforming the infinite-dimensional problem into a finite-dimensional one. The discretized problem takes the form of a NonLinear Programming (NLP), which is a constrained discrete minimization problem of a given objective function. The latter is solved using well-known optimization techniques, e.g. Sequential Quadratic programming (SQP). If direct methods are also prone to local minima, in particular in nonlinear optimization where the convexity of the function cannot be proven, they are generally more robust to the initial guesses, as reported in [Rao, 2009].

One of the most widely utilized direct methods for recasting the OCP into the NLP is a method called *multiple shooting* [Bock, 1984] It divides the receding horizon into multiple sub-intervals, on which the value of the system state and the cost function are evaluated. Despite the addition of several continuity constraints at the junction of each sub-interval, the reduction of the integration interval makes the formulation much more accurate.

2.5.2 Nonlinear MPC

With this being said, we can now present the Model Predictive Control approach. MPC is a control strategy that emerged in oil industry in the 1980s, and which has been developed over the past decades in various contexts. It is nowadays vastly employed in many robotics research areas. MPC is an optimization-based control policy. The optimization problem is often subject to a set of equality and inequality constraints. It is a receding horizon-based technique

The length of the receding horizon depends on the capability to extrapolate the system behavior over a long period of time, hence on the precision of the model and numerical tools. Once the minimization is solved and the sequence of inputs is computed, the first input of this sequence is applied to the system, and another optimization cycle is performed, accounting for the new system configuration.

A generic discrete formulation of minimization problem is

$$\min_{\substack{\mathbf{x}_0 \dots \mathbf{x}_N \\ \mathbf{u}_0 \dots \mathbf{u}_{N-1}}} \quad \sum_{k=0}^{N-1} h_r(\mathbf{x}_k, \mathbf{u}_k) + h_t(\mathbf{x}_N) \tag{2.1a}$$

$$s.t. \quad \mathbf{x}_0 = \mathbf{x}(t) \tag{2.1b}$$

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k), \qquad k \in \{0, N-1\} \tag{2.1c}$$

$$\underline{\mathbf{u}} \leq \mathbf{u}_k \leq \overline{\mathbf{u}}, \qquad k \in \{0, N-1\} \tag{2.1d}$$

$$\underline{\mathbf{x}} \leq \mathbf{x}_k \leq \overline{\mathbf{x}}, \qquad k \in \{0, N\} \tag{2.1e}$$

where $\mathbf{x}$ and $\mathbf{u}$ are the system state and input vectors, $\mathbf{f}$ summarizes the discrete system dynamics, and h_r and h_t are two continuous semi-definite positive linear forms respectively defining the *running* and *terminal* cost functions. Equation (2.1b) is the initialization equation of the problem. Indeed, MPC is a closed-loop formulation that requires state feedback at each optimization cycle. Equations (2.1d) and (2.1e) are respectively the input and state constraints inequality to which the system is subject. Finally, N is the number of samples considered over the receding horizon. We remark

that if the state constraints are optional, the input constraints are generally not, as they are required to prove that the infimum of the cost function for a given state vector is reachable by an admissible sequence of control inputs [Grimm, 2005].

The MPC technique consists of iteratively optimizing for successive sampling time instants over the receding horizon, which makes it differ from the Linear-Quadratic Regulator (LQR), another optimal control policy in which a single optimal input is computed for the whole horizon. While LQR has better stability properties than MPC, it displays losses of performance when operating far from the linearized operating point. Another major advantage of the MPC over LQR is the ability to handle nonlinear systems, through a variant summarily called Nonlinear MPC (N-MPC). For N-MPC, a large scale NLP which encompasses the system dynamics and the various constraints and objectives is solved at every sampling time over the receding horizon. This is achieved through local linearizations of the system. N-MPC problems are not necessarily convex, which challenges the stability analysis and the convergence to a solution. Closed-loop stability is indeed proven in [Mayne, 2000] for constrained linear MPC, but proofs in the nonlinear case are more difficult to obtain. In [Grimm, 2005], a theoretical proof is given that there exists a sufficiently long horizon length that ensures stability for unconstrained N-MPC, under some boundedness assumption on the cost function. Further proof in this direction is provided by [Grüne, 2010], in which stability criteria for N-MPC without terminal constraints are analyzed. Also, additional assumptions on the terminal cost not being of Lyapunov type allows for shorter time horizons. This proof is however formulated for unconstrained N-MPC, which are very restricting, as the main perk of optimal control is to handle constrained systems. The stability analysis of N-MPC remains an open problem, and further studies are required.

Nonetheless, N-MPC has empirically proven to be a very efficient and robust control policy for nonlinear systems. This makes it well suited for ARs, and many recent works are employing this technique. Fast maneuverings in particular are more and more tackled with N-MPC. A recent study in [Foehn, 2021] demonstrates successful tracking of agile trajectories up to 20 m/s. A thorough comparative study is conducted in [Sun, 2022], in which N-MPC shows superior performances w.r.t. a state of the art differential-flatness-based feedback linearization controller. In particular N-MPC demonstrates more robustness to most of model uncertainties than the differential-flatness-based controller. However, both controllers are evaluated in presence of a Incremental Nonlinear Dynamic Inversion (INDI) inner-loop regulator [Smeur, 2018]. Another perk of N-MPC is to consider several, sometimes conflicting, objective functions. For instance, the previously mentioned N-MPC in [Bicego, 2020] for the control of the tiltable-propeller hexarotor uses an objective function which encompasses the tracking of a reference trajectory as well as the maximization of the energy efficiency through the active control of the propeller tilting angle.

In aerial robotics, (N-)MPC is mostly used as a real-time controller for trajectory tracking. More rarely, it can be utilized as a trajectory planner, often providing minimum-time optimal trajectories. The loop can be closed, so that the MPC performs an online replanning of the trajectory, whose tracking might not be accurate. Indeed, recent technological advancements allowed to perform online nonlinear optimization. A thorough comparison between linear and nonlinear MPC for ARs is proposed in [Kamel, 2017]. It concludes that N-MPC presents several advantages

on linear MPC, in particular for handling disturbances and improving the tracking accuracy. Moreover, N-MPC is shown to outperform linear MPC in terms of computational speed using Real-Time Iteration (RTI).

2.5.3 Real-Time Iteration

RTI is a solving scheme introduced in [Diehl, 2002], in which the NLP is built using a multiple shooting point strategy, and solved using SQP. The latter is performed only once at each sampling time of the receding horizon. The solution computed in this way is therefore suboptimal, since the global convergence is prevented. However, the computation cost is substantially diminished, enabling real-time onboard computing. Furthermore, in a recent study, [Gros, 2020] shows pieces of evidence that RTI mostly yields a small error w.r.t. a fully-converged solution. It concludes that RTI strategy can be deployed as a substitute to normal N-MPC solving, with a reduced computational cost, even compared to linear MPC.

To further improve the solving speed, a solution is proposed in [Chen, 2017], called fixed-time block update RTI. It is based on the assessment that a computationally critical step in RTI is the system linearization at each sampling point. Hence, the proposed solution consists of using a local measure of the nonlinearity of the system (e.g., based on the curvature of the model function manifold [Bates, 1980]) in order to assess whether the system is locally behaving close to its linear approximation. The linearization is re-computed only when the previously computed one is not suited, thus reducing the overall computational cost.

Since a single optimization step is performed, the solution is necessarily not converged. To palliate this issue, the choice of the initial guess is crucial. In [Diehl, 2005], it is proposed to use the solution previously computed from past iterations. This heuristic is called *warm start* usually provides good results, but is not robust to disturbances, sudden changes in the environment, or unstable configurations. This problem is addressed in [Mansard, 2018] by learning a control policy in an offline training phase. This control policy is leveraged to provide an accurate warn initial guess for the MPC solver. This approach is tested, among other systems, on ARs. It has been further validated in [Dantec, 2021], in which it enabled a N-MPC to perform the whole-body torque control humanoid robot, partly thanks to the consequent gain in computational resources. This technique is however rarely employed in off-the-shelf numerical solvers, mainly because of the burden of performing a training phase.

2.5.4 Instances of Nonlinear MPC for Aerial Robotics

The recent literature shows several instances of N-MPC with an AR, as collected in recent surveys, e.g. [Nguyen, 2021]. In [Kamel, 2015], a N-MPC is used to control the inner-loop attitude on $SO(3)$ of a quadrotor, while the outer-loop is delegated to a 2^{nd} order LQR with integral regulation. Several works leverage multi-objective cost functions to handle obstacle avoidance with agile maneuvering, e.g. in [Pereira, 2021] which considers non-convex obstacles. The $SE(3)$ state parameterization allows to perform agile maneuvering without orientation singularities, and the work addresses the consequent challenges, such as the proper definition of the orientation error and

the $SO(3)$ integration stability issues. The framework was however simply validated through numerical simulations. On a similar topic, [Lu, 2021] tackles the rigorous definition of an on-manifold MPC through the local Lie Algebra-based linearization of the nonlinear dynamics.

An interesting method for combined trajectory optimization and tracking is proposed in [Neunert, 2016]. It relies on an inner-loop trajectory tracker, whose gains are computed online by the outer-loop N-MPC, which acts as a local trajectory planner. This combined approach allows to maintain a hierarchical structure which is more robust to disturbances while leveraging the replanning capabilities of the N-MPC to compute the feedback gains of the inner-loop.

Similar to the learning-based warm start in [Mansard, 2018], some solutions have been proposed to combine N-MPC and machine learning. In [Torrente, 2021], a learning approach based on Gaussian Processes is used to learn the dynamic model of the aerodynamic drag. The model is used in a N-MPC, leading to increased accuracy of the trajectory tracking. A Learning MPC in proposed in [Li, 2022], which consists of using previous iterations of the same trajectory as initial guesses for the controller, leading to incremental refining. From a roughly constant and low speed initial trajectory, the proposed framework is able to achieve a very efficient minimum-time trajectory after a small number of iterations, e.g. 6 in their proposed experiments.

In the scope of aerial manipulation, a hierarchical approach based on nested optimization is proposed in [Lunni, 2017]. Nested optimization consists of computing a cascade of optimizations, each minimizing a different cost function, often referred to as *task function* [Khatib, 1987]. The task functions are organized hierarchically, such that the higher priority task are solved first, and included in constraints in the subsequent ones. Therefore, all the subsidiary tasks are solved in the nullspaces of the previous ones, ensuring that those are fulfilled in priority. Such priority management is meaningful in redundant systems such as aerial manipulators, since it allows to avoid the conflicting multiplication of parallel objectives. Another controller introduced in [Martí-Saumell, 2022] proposes to import techniques from legged robotics to account for contact interaction in the N-MPC.

Rotor fault tolerance is an active field of research in aerial control, and N-MPC schemes are also studied in this direction. The rotor failure of a symmetric hexarotor is analyzed in [Tzoumanikas, 2020], which shows that N-MPC is able to maintain the system stability. The problem is tackled with a quadrotor in [Sun, 2021; Nan, 2022]. The proposed N-MPC with INDI regulation is demonstrated to stabilize the quadrotor subject to propeller failure, and to follow a recovery trajectory.

Lastly, one could cite an example of mixed-initiative N-MPC proposed in [Barros Carlos, 2021], in which a N-MPC controller is designed for the training of UAV human pilot, to ensure both human and hardware safety. The N-MPC complies with the inputs of the pilot to some extend, but is able to recover control if the remote commands would yield poor performances and jeopardize the system stability. This is done through an assessment of the quality of the human piloting w.r.t. the trajectory to track, which is reflected in the cost function to either neglect or acknowledge the human inputs related terms.

These last three works illustrate another aspect of the use of N-MPC as a full-state controller, similar to [Bicego, 2020]. Indeed, the aforementioned works have a cascaded structure, and rely on an inner-loop to compute the rotor input, while the N-MPC is tasked to generate an intermediate tracked variable, e.g. angular rates. Exception is made of [Kamel, 2015] in which the N-MPC acts as the attitude regulation loop. Yet, recent works tackled the definition of a full-state N-MPC which accounts for the actuation limits of the AR. The motivation for such a controller is discussed in Chapter 3, along with in depth modeling of the associated system, based upon [Bicego, 2020].

2.6 Perception Aware N-MPC

This section presents the main corpus of works upon which this thesis is built. As previously mentioned, optimization-based techniques, and in particular N-MPC, have been recently favored for the definition of so-called perception-aware controllers; that is controllers including perceptions objectives or constraints. Indeed, such optimization is well suited to express the nonlinear visibility constraints.

A first non-MPC approach has been proposed in [Thomas, 2017], in which the active tracking of a mobile object with a quadrotor is tackled through a velocity error minimization planning problem. Visibility is expressed with the bearing vector, that is constrained to belong to a pyramidal approximation of the camera FoV. The minimization is solved as a SQP problem in a myopic fashion The planner relies on differential-flatness, and the dynamic feasibility of the trajectory is enforced by constraining continuity on the position derivatives. The work is validated through experiments with a mobile object moving in straight line, in which the upholding of the visibility is assessed.

A similar problem is addressed in [Penin, 2017], with a N-MPC approach that generates minimum-time trajectories with a quadrotor equipped with a camera, which is tasked to maintain visibility over a set of features. First, a geometric angular criterion is defined to assess the visibility as a closed-form function of the system state. This criterion is constrained in the proposed NLP formulation for each observed point. The N-MPC relies on differential-flatness to simplify the dynamics of the system, allowing efficient computation. The work is validated through simulations where the quadrotor underactuation is clearly handled by the controller, as the AR goes up to enlarge the ground-level FoV of the camera before tilting to reach its destination.

Going further, [Penin, 2018] uses a similar approach to perform maneuvering under perception, collision and occlusions constraints. The minimum-time planning is traded for a constant horizon problem, which produces constrained trajectories. The visibility criterion introduced in [Penin, 2017] is reused, while generalized position coordinates constraints are added for collision avoidance; and additional angular constraints are stated for preventing complete occlusion of the tracked feature. This last constraint is relaxed using a slack variable, in order to allow partial occlusion if the AR stability is a stake. However, the newly proposed framework is handling a single tracked object. Another drawback of this technique is the restriction to spherical obstacles, which positions are perfectly known at all times. The N-MPC is

validated in simulations, demonstrating its capability to produce the desired behavior. An experimental obstacle-free test is also performed to assess the tracking capabilities of the framework.

Later, [Li, 2021] proposed a vision-constrained N-MPC for suspended load cargo. A quadrotor is tasked to carry a load, whose position is visually assessed, rather than resorting to an inertial estimator. Thereby, it employs a perception-constrained N-MPC to ensure visibility through the generated motion. Contrary to the approach in [Penin, 2018], the object is not fixed, but its motion depends on the AR own dynamics, thus requiring a more complex model. Real experiments are conducted to validate the approach.

These works propose interesting use of the N-MPC toward vision-constrained motions. However, these strategies focus on trajectory planning using a constrained solver, whose output is fed to an unconstrained controller. Furthermore, despite the use of constant replanning of the solution, the computation time of the N-MPC is not discussed. If the replanning is not fast enough, these solutions might not be suited to handle mobile objects of unknown motion.

In [Falanga, 2018], the newly proposed course of action is rather to leverage the N-MPC for the control of the AR. Indeed, the N-MPC is used to compute optimal angular rates and global propeller thrust. In addition, the problem is tackled following a different angle, that is including a visibility objective instead of hard constraints. This is motivated by the fact that the framework is oriented toward vision-based state estimation rather than toward the tracking of a given object. Thereby, the features of interest are the feature points detected by the VIO software, all of these being equivalent. It is therefore meaningless to constrain some to belong to the FoV. From there, the visibility objective is defined w.r.t. the barycenter of the detected feature points. Contrary to the aforementioned works which expressed the visibility in terms of 3D bearing vectors, [Falanga, 2018] proposes to define it directly onto the image plane, through a suited projection via the pinhole camera model. It is claimed that the convexity of the problem is maintained by the positivity of the projectors. The N-MPC is validated through an extensive set of experiments, demonstrating that the controller handles the platform orientation suitably to maintain visibility during, e.g, circular fast maneuvers. However, this solution disregards the constraints imposed on the visibility, implicitly assuming that enough features will be detectable in the vicinity of the barycenter.

Using similar techniques, [Paneque, 2022] proposes perception-driven power line perching maneuvers based on N-MPC. The formulation is based on prior works conducted throughout this thesis and on the work in [Falanga, 2018]. The controller formulates geometric visibility constraints to ensure that the goal (i.e., a power line) is always visible. Moreover, a rephrasing of the aforementioned objective cost is proposed to handle line segments rather than punctual features. Interestingly, the authors validate the N-MPC for trajectory tracking, then propose to use the same formulation for minimum-time trajectory generation. Actual perching experiments are performed at high speed, showing upside-down perching.

Another image-based N-MPC for AR is proposed in [Lee, 2020]. Therein, the pixel state of the observed feature is integrated into the system state vector. Rather than resorting to geometrical approaches, the pixel dynamics are estimated through

a CNN version of the classical Optical Flow technique, named Deep Optical Flow. Optical flow consists of an estimation of an object motion from a sequence of images. Hence, the CNN infers the pixel dynamics, which is integrated into the N-MPC. The N-MPC strategy is close to [Falanga, 2018], which is an objective-based approach and the computation of angular rates and total thrust. The framework is tested in drone racing simulation. However, the cascade of computationally heavy processing (CNN-based object detection using YOLO, depth estimation using Deep Optical Flow, and N-MPC control) makes this solution hard to implement on an actual AR.

Again based on some learning policy, [Greeff, 2020] proposes a controller which enforces localization. A data-driven perception model is used to estimate the "chance" that a valid observation will be yielded in a given configuration. This chance is constrained to be greater than an arbitrary threshold, theoretically ensuring that the AR will recover its state at any instant. Again, the N-MPC is used for the attitude control of the AR. This work is further discussed in Chapter 6.

Finally, another approach is proposed in [Mueller, 2020]. Therein, a N-MPC-controlled quadrotor is tasked to autonomously approach the window of a building. A multi-objective cost function is used, to generate the frontal approaching motion while maintaining the window at the center of the FoV. Similar to [Thomas, 2017], this work proposes an application-specific computation of the distance to the object of interest, rather than resorting to specific Again, the work is validated in simulation.

Building upon these rich works, this thesis focuses on the definition of a generic perception-aware framework, able to handle localization and tracking activities. Furthermore, two drawbacks are common in all the cited works. First, they only focus on collinear quadrotors, e.g. by employing differential-flatness-based relaxation of the dynamics. Section 2.2 assessed the large variety of existing designs in the literature and therefore the need for generic methods. Second, the low-level actuation is never considered in the N-MPC, which furthermore relies on an unconstrained inner-loop. An approach similar to [Bicego, 2020] would allow to ensure that the computed trajectory is dynamically feasible, and constrained down to the motor-level inputs with as few intermediate steps as possible. This approach is further motivated in Chapter 4.

Chapter 3

Modeling

Contents

3.1 Introduction **36**
3.2 Notations **36**
3.3 Orientation Representations **38**
3.3.1 Rotation Matrices, Euler Angles and Angle-Axis 38
3.3.2 Orientation as Unit Quaternions 40
3.4 Generically Tilted Multi-Rotor **42**
3.4.1 GTMR Modeling 42
3.4.2 GTMR Actuation 44
3.5 Sensor Model **46**
3.5.1 Generic Sensor 46
3.5.2 Specific Case of a Camera 47
3.6 Fiducial markers **48**
3.7 Sensor measurement filtering **50**

3.1 Introduction

Because of the tight entanglement between the onboard perception and the AR control, it is of paramount importance for a perception-aware controller to capture at best the complexity of the AR motion. It is thus impossible to use a linear approximation, which inherent imprecision might induce unexpected loss of visibility, leading to possible undesired consequences.

Similarly, it is important for the controller to consider the limited actuation capabilities of the AR. In particular, exploiting fictitious limits on the angular rates, as it is typically done in classical N-MPC approaches, does not leverage the full motion capabilities of the AR. As presented in Section 2.5, considering the rotor velocities and accelerations provides a more accurate approximation of the actuation limits.

Moreover, because of the large variety of existing AR designs, which was briefly introduced in Section 2.2, it is desirable to model the AR in a generic fashion, such that the resulting control policy is versatile and can be easily transposed amongst platforms.

This chapter introduces the mathematical models used throughout the thesis. After a succinct overview of the quaternion representation of the 3D rotations, we present in detail the model of the GTMR, as well as the nonlinear dynamics governing its motion.

Secondly, we present the perceptive sensor model. For the sake of genericity, we propose to use an abstracted range-bearing sensor. The choice of observation model associated with the sensor is later discussed, as well as a filtering policy for the measurement.

Finally, we present the fiducial markers used as static or mobile features in the various experiments presented in Part II.

3.2 Notations

This section presents the mathematical writing convention used throughout the manuscript. Variables are written using Greek or Latin letters, with the following rules:

- normal font for scalars,
- small bold for vectors,
- capital bold for matrices.

The null and identity matrices of size $n \times m$ are respectively denoted $\mathbf{O}_{n\times m}$ and $\mathbf{I}_{n\times m}$ (simplified $\mathbf{O}_n$ and $\mathbf{I}_n$ for square matrices).

Sets are written using dedicated letters, e.g. $\mathbb{R}$ for real numbers, $\mathbb{Q}$ for quaternion numbers, or $\mathbb{S}$ for the unit sphere of $\mathbb{Q}$. Additionally, continuous intervals are written using straight brackets $[a,\ b]$, while discrete sets of integers are written using curly ones $\{a,\ b\}$.

Frames and Frame Transformations

The world inertial frame is denoted with $\mathcal{F}_W$, with its origin O_W and canonical base $(\mathbf{x}_W, \mathbf{y}_W, \mathbf{z}_W)$. Similarly, we define the body frame $\mathcal{F}_B$ attached on the AR geometrical center of the robot O_B, and $\mathcal{F}_S$ the frame of the sensor (or $\mathcal{F}_C$ in the case of a camera). Lastly, the frame attached to a mobile punctual object M is written $\mathcal{F}_M$.

The translation from a frame $\mathcal{F}_1$ to a frame $\mathcal{F}_2$ is denoted ${}^2\mathbf{p}_1 = [{}^2x_1, {}^2y_1, {}^2z_1]^\top$. Similarly, the rotation from $\mathcal{F}_1$ to $\mathcal{F}_2$ is denoted, in its matrix form, with ${}^2\mathbf{R}_1 \in \mathbb{R}^{3\times3}$. We remark that ${}^2\mathbf{p}_1$ simultaneously denotes the position of O_1 w.r.t. $\mathcal{F}_2$, and ${}^2\mathbf{R}_1$ the attitude of $\mathcal{F}_1$ w.r.t. $\mathcal{F}_2$. Finally, the 6D transform from $\mathcal{F}_1$ to $\mathcal{F}_2$ is ${}^2\mathbf{T}_1 \in SE(3)$.

More generally, the reference frames of vectors is written with a left topscript, while the point described is denoted with a right subscript.

Definition	Symbol
State vector	$\mathbf{x}$
System input vector	$\mathbf{u}$
Position vector	$\mathbf{p}$
Velocity vector	$\mathbf{v}$
Acceleration vector	$\mathbf{a}$
Angular velocity vector	$\boldsymbol{\omega}$
Orientation matrix	$\mathbf{R}$
Orientation quaternion	$\mathbf{q}$
Euler angles vector	$\boldsymbol{\eta}$
roll	ϕ
pitch	θ
yaw	ψ
Euler vectors	$\mathbf{r}$
rotation angles	θ
rotation angle-axis	$\mathbf{u}$
Gravity acceleration	g
mass	m
Inertia tensor	$\boldsymbol{I}$
Actuator thrusts vector	$\boldsymbol{\gamma}$
Rotor velocities vector	$\boldsymbol{\Omega}$

Table 3.1: Usual mathematical symbols used throughout the manuscript.

Operators

The Hamilton product of two quaternions, which details are recalled in Section 3.3.2, is denoted $\otimes$. The skew operator of the vector product is denoted $[\bullet]_\times$, and defined

as

$$\forall \mathbf{v} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \in \mathbb{R}^3, \ [\mathbf{v}]_\times = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix} \tag{3.1}$$

The Moore-Penrose pseudo-inversion operator is denoted with $\bullet^\dagger$. Finally, lower and upper bounds associated with bounded variables are denoted using respectively the $\underline{\bullet}$ and $\overline{\bullet}$ operators.

3.3 Orientation Representations

This section recalls the various representation of the rigid body orientation, and motivates the choice of quaternions for the upcoming modeling.

3.3.1 Rotation Matrices, Euler Angles and Angle-Axis

The 3D rotation between two frames $\mathcal{F}_1$ and $\mathcal{F}_2$ is described by the rotation matrix $\mathbf{R} = {}^2\mathbf{R}_1 \in \mathbb{R}^{3\times3}$, where the column of $\mathbf{R}$ are the coordinates of the unit vectors of $\mathcal{F}_1$ expressed in $\mathcal{F}_2$. Each rotation matrix is therefore full-ranked and orthogonal. Those matrices form a group under the operation of composition – i.e. matrix multiplication – which is denoted $SO(3)$, the Special Orthogonal group of dimension 3. As rotations preserve distances and relative angles, they are good candidates to describe the orientation of rigid bodies in an inertial frame. Hence, they are of major importance in robotics.

The time derivative of such matrices can easily be obtained from the angular rates $\boldsymbol{\omega}$ around each axis, though the relation

$$\dot{\mathbf{R}} = \mathbf{R}[\boldsymbol{\omega}]_\times. \tag{3.2}$$

Remark. Equation (3.2) hold for angular rates $\boldsymbol{\omega}$ defined locally, i.e., in $\mathcal{F}_B$. For globally defined angular rates (i.e., ${}^W\boldsymbol{\omega}$), the formula from [Siciliano, 2009] holds, that is

$$\dot{\mathbf{R}} = [{}^W\boldsymbol{\omega}]_\times \mathbf{R}. \tag{3.3}$$

A comprehensive proof of the related derivation can be found in [Solà, 2017]. All the derivation formula presented hereafter consider locally defined $\boldsymbol{\omega}$.

However, using matrices to represent the orientation state of a system is not convenient. First, this representation is not minimal, since each matrix $\mathbf{R}$ is characterized by 9 coordinates for a transformation with 3 degrees of freedom. Second, while $SO(3)$ is a group w.r.t. the matrix multiplication operation, it is not stable w.r.t. summation. This is very restricting for control and filtering processes, which are most often based on the assumption of linearly additive quantities.

Therefore, some reduced vector representations of rotations have been proposed. The most widely spread one is the Euler angles representation, which consists of describing the angle of rotation about each axis of the reference frame. The most common one in aerial robotics is the roll-pitch-yaw convention, in which ϕ, θ and ψ are the angles respectively about the x, y, z axes. Some other roll-pitch-yaw

convention invert the x and z names [Siciliano, 2009]. Their signs are determined using the right-hand rule. They are defined as

$$\boldsymbol{\eta} = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} \in]-\pi,\ \pi] \ \times\ \left]-\frac{\pi}{2},\ \frac{\pi}{2}\right[\ \times\ ]-\pi,\ \pi]\,. \tag{3.4}$$

Therefore, elementary rotations matrices about the 3 axes can be defined, and combined to retrieve the corresponding element of $SO(3)$. This representation has several advantages. First, the angles are additive quantities, thus the usual control and filtering algorithms can be employed. Second, the representation is minimal, since $\boldsymbol{\eta} \in \mathbb{R}^3$. Finally, there exist a direct mapping – although nonlinear in $\boldsymbol{\eta}$ – between the angular rates $\boldsymbol{\omega}$ and the Euler angles derivatives $\dot{\boldsymbol{\eta}}$:

$$\dot{\boldsymbol{\eta}} = \begin{bmatrix} 1 & \cos\phi\tan\theta & \sin\theta\cos\phi \\ 0 & \cos\phi & -\sin\theta \\ 0 & \frac{\sin\phi}{\cos\theta} & \frac{\cos\phi}{\cos\theta} \end{bmatrix} \boldsymbol{\omega}. \tag{3.5}$$

The main limitation of Euler angles is the existence of several singularities. In particular, the angles are defined on segments, hence the transitions at π for roll and yaw angles are discontinuous. Moreover, the Euler Angles are subject to the so-called *Gimbal Lock*, e.g. that a pitch angle θ of $\frac{\pi}{2}$ introduces mathematical singularities in the equations. Such angle is therefore not properly defined, which is indeed reflected in Equation (3.5).

Another common representation for orientation is to decouple two quantities: the 3D unit vector $\mathbf{u}$ around which the rotation is performed, and the angle $\theta \in [0,\ \pi[$ of the rotation. The rotation angle is defined positive and minimal, by choosing $\mathbf{u}$ accordingly. From those, the rotation matrix is retrieved using the Rodrigues formula:

$$\mathbf{R} = \mathbf{I}_3 + \sin\theta[\mathbf{u}]_\times + (1-\cos\theta)[\mathbf{u}]_\times^2. \tag{3.6}$$

This representation can be condensed in a more concise *rotation vector*, often called *Euler vector*, defined as the 3D vector

$$\mathbf{r} = \theta\mathbf{u}, \tag{3.7}$$

which is collinear to $\mathbf{u}$ and whose norm is equal to θ. This minimal representation appears in the Lie Algebra study of the rotation group, which is presented in [Solà, 2018], in a restricted use for robotics. In particular, the Exponential and Logarithm mappings

$$\mathbf{R} = \mathrm{Exp}(\mathbf{r}), \tag{3.8a}$$
$$\mathbf{r} = \mathrm{Log}(\mathbf{R}), \tag{3.8b}$$

allow to retrieve Equation (3.6) from an algebraic derivation. We refer to [Solà, 2018] for further details on those, or to, e.g., [Chirikjian, 2011] for an in-depth book on the matter. The angle-axis representation is also subject to singularities, notably for rotation of π rad, since two representations exist.

3.3.2 Orientation as Unit Quaternions

This section is intended as a brief overview of the use of unit quaternion as a representation of orientations. For a comprehensive reference on the matter, we refer to, e.g., [Kuipers, 1999; Diebel, 2006].

The quaternion space is denoted $\mathbb{Q}$. We recall that quaternions are described by 4 real numbers, hence $\mathbb{Q}$ is homeomorphic to $\mathbb{R}^4$ just as the complex plane $\mathbb{C}$ is homeomorphic to $\mathbb{R}^2$. A quaternion $\mathbf{q}$, defined as

$$\mathbf{q} = q_w + iq_x + jq_y + kq_z \in \mathbb{Q}, \tag{3.9}$$

is then conveniently written as

$$\mathbf{q} = \begin{bmatrix} q_w \\ q_x \\ q_y \\ q_z \end{bmatrix} \in \mathbb{R}^4. \tag{3.10}$$

Remark. Alternative quaternion conventions exist, which are roughly equivalent, for instance by placing the scalar component q_w last in $\mathbf{q}$. Formulas presented hereafter follow the convention from Equation (3.10).

A multiplicative operator is defined on $\mathbb{Q}$, namely the Hamilton product, defined as

$$\mathbf{q}_1 \otimes \mathbf{q}_2 = \begin{bmatrix} q_{w1}q_{w2} + q_{w1}q_{x2} + q_{w1}q_{y2} + q_{w1}q_{z2} \\ q_{x1}q_{w2} + q_{x1}q_{x2} + q_{x1}q_{y2} + q_{x1}q_{z2} \\ q_{y1}q_{w2} + q_{y1}q_{x2} + q_{y1}q_{y2} + q_{y1}q_{z2} \\ q_{z1}q_{w2} + q_{z1}q_{x2} + q_{z1}q_{y2} + q_{z1}q_{z2} \end{bmatrix}. \tag{3.11}$$

From Equation (3.11), it can be derived that

$$\|\mathbf{q}_1 \otimes \mathbf{q}_2\| = \|\mathbf{q}_1\| \, \|\mathbf{q}_2\| \, . \tag{3.12}$$

It is proposed to represent the rotations as unit quaternions, i.e. whose norm is 1. Starting from the angle-axis representation $\mathbf{r} = \theta\mathbf{u}$ of a rotation, we define a rotation quaternion as

$$\mathbf{q} = \begin{bmatrix} \cos\frac{\theta}{2} \\ \mathbf{u}\sin\frac{\theta}{2} \end{bmatrix}, \tag{3.13}$$

which satisfies $\|\mathbf{q}\| = \sin^2\frac{\theta}{2} + \cos^2\frac{\theta}{2} = 1$. It can be shown, combining Equations (3.11), (3.13) and (3.6) that

$$\forall \mathbf{v} \in \mathbb{R}^3, \quad \begin{bmatrix} 0 \\ \mathbf{R}\mathbf{v} \end{bmatrix} = \mathbf{q} \otimes \begin{bmatrix} 0 \\ \mathbf{v} \end{bmatrix} \otimes \mathbf{q}^*, \tag{3.14}$$

where $\mathbf{R}$ and $\mathbf{q}$ are the matrix and quaternion representation of the same rotation, and $\mathbf{q}^*$ is the conjugate quaternion of $\mathbf{q}$, defined as

$$\mathbf{q}^* = \begin{bmatrix} q_w \\ -q_x \\ -q_y \\ -q_z \end{bmatrix}. \tag{3.15}$$

Equation (3.14) therefore defines the rotation action of a unit quaternion on a given vector.

Remark. For unit quaternions, the conjugate and inverse and equivalent, since the inverse is defined as $\mathbf{q}^{-1} = \frac{\mathbf{q}^*}{\|\mathbf{q}\|^2}$. In the literature, Equation (3.14) is thus sometimes written as $\mathbf{q} \otimes \begin{bmatrix} 0 \\ \mathbf{v} \end{bmatrix} \otimes \mathbf{q}^{-1}$.

Using this mapping, it can be seen that the set of all rotations is entirely encoded onto the unit sphere of $\mathbb{Q}$, denoted $\mathbb{S} = \{\mathbf{q} \in \mathbb{Q} \mid \|\mathbf{q}\| = 1\}$, which is a 3-dimensional manifold. Moreover, Equation (3.12) shows that $\mathbb{S}$ is stable through the Hamilton product, which acts as a composition for quaternion rotations. This can be proven starting from Equation (3.14), assessing that two rotations respectively represented by matrices $\mathbf{R}_1$, $\mathbf{R}_2$ and quaternions $\mathbf{q}_1$, $\mathbf{q}_2$, verify

$$\forall \mathbf{v} \in \mathbb{R}^3, \quad \begin{bmatrix} 0 \\ \mathbf{R}_1\mathbf{R}_2\mathbf{v} \end{bmatrix} = \mathbf{q}_1 \otimes \mathbf{q}_2 \otimes \begin{bmatrix} 0 \\ \mathbf{v} \end{bmatrix} \otimes \mathbf{q}_2^* \otimes \mathbf{q}_1^*, \tag{3.16}$$

Therefore, quaternions offer a minimal singularity-free representation of orientations. It is also convenient since this representation does not require computing the matrix form to perform the rotation action, as opposed to Euler angles. Quaternions are commonly used in robotics, as well as in many domains, such as compute vision, 3D graphics or flight dynamics for planes and spacecrafts.

An analysis of the existing metrics for rotations, including quaternions, is performed in [Huynh, 2009]. More details in this regard are provided in 4.4.2.

The time derivation formula of unit quaternions (proven, e.g., in [Solà, 2017]) is

$$\dot{\mathbf{q}} = \frac{1}{2}\, \mathbf{q} \otimes \begin{bmatrix} 0 \\ \boldsymbol{\omega} \end{bmatrix}. \tag{3.17}$$

Again, this holds for locally defined angular rates. The alternate form can sometimes be found in the literature, where the two terms in Equation (3.17) are inverted, which gives

$$\dot{\mathbf{q}} = \frac{1}{2} \begin{bmatrix} 0 \\ {}^{W}\boldsymbol{\omega} \end{bmatrix} \otimes \mathbf{q}. \tag{3.18}$$

The unit quaternions sphere $\mathbb{S}$ is not stable over summation, which brings back the same issues raised for rotation matrices. However, the compactness of quaternions w.r.t. matrices (4 components instead of 9) makes them easier to handle in controllers. For instance, two geometrical control strategies for ARs involving quaternions are introduced in [Fresk, 2013; Carino, 2015]. Yet, an important known issue regarding quaternion for predictive controllers is that their numerical integration, e.g. using the Runge-Kutta integration scheme, tends to not necessarily adhere to the manifold. This issue is addressed in [Rucker, 2018], which proposes to implement regularized non-unit quaternions. From there, using a redundant mapping from $\mathbb{Q}$ to $\mathbb{S}$, the non-unit quaternions are used in a numerically stable fashion. The software implementation of the controller proposed throughout the subsequent chapter, detailed in Appendix B, makes use of this scheme.

Quaternion filtering is also affected by these issues. A solution to proceed with quaternion filtering is to leverage Lie algebra to propagate the errors in a vector space tangent to the manifold, then projected back onto the manifold, using the Exp and Log from Equation (3.8). More details are provided in Section 6.3.

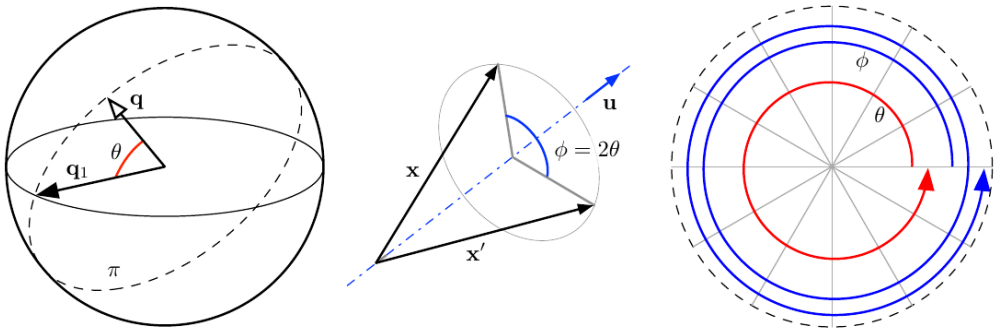


Figure 3.1: Double coverage of the rotation manifold (courtesy of [Solà, 2017]). Left: the angle θ between a quaternion $\mathbf{q}$ and the identity quaternion $\mathbf{q}_1$. Center: the resulting 3D rotation of angle ψ. Right: superposing the evolution of ϕ as θ performs covers a 2π course.

Remark. From Equation (3.13), it can be observed that the unit quaternions perform a double coverage of the rotation set, see Figure 3.1. It implies that two opposite unit quaternions $\mathbf{q}$ and $-\mathbf{q}$ represent the same 3D rotations. This has an impact, e.g., on the proper definitions of quaternion metrics.

3.4 Generically Tilted Multi-Rotor

3.4.1 GTMR Modeling

The multi-rotor Aerial Robot, which can take various designs (see Section 2.2), can be modeled using a common, generic formalism. An instance of such formalism is the so-called GTMR model (presented, e.g., in [Michieletto, 2018]). The GTMR is modeled as rigid body of mass m and inertia tensor $\boldsymbol{I}$ with n actuators (motor and propeller pairs), arbitrarily placed and oriented around O_B. Contrary to the assumption made in [Michieletto, 2018; Bicego, 2020], the Center of Mass (CoM) in which the gravity is applied is in general not coincident with O_B. We denote the offset between this CoM and O_B by ${}^B\mathbf{p}_{CoM}$. The standard collinear configuration is the most widely spread, and is more energy efficient, while tilted configurations allow for larger actuation spans. Those are defined by two angles $\alpha_{a,i}$ and $\beta_{a,i}$ for each propeller i, respectively in the radial and tangential directions, as depicted in Figure 3.2.

Following the nomenclature from Section 3.2, the GTMR state vector is denoted $\mathbf{x}$ and is then defined as

$$\mathbf{x} = [{}^W\mathbf{p}_B^\top \; {}^W\mathbf{q}_B^\top \; {}^W\mathbf{v}_B^\top \; {}^B\boldsymbol{\omega}_B^\top \; \boldsymbol{\gamma}^\top]^\top \in \mathbb{R}^{13+n}, \tag{3.19}$$

where $\boldsymbol{\gamma} = \begin{bmatrix}\gamma_1 & \cdots & \gamma_n\end{bmatrix}^\top \in \mathbb{R}^n$ is a vector containing the n forces produced by the n actuators. Accordingly, the system input vector $\mathbf{u}$ is defined as the time derivative of $\boldsymbol{\gamma}$:

$$\mathbf{u} = \dot{\boldsymbol{\gamma}}. \tag{3.20}$$

The choice of state and input, regarding the GTMR actuation, is further discussed in Section 3.4.2.

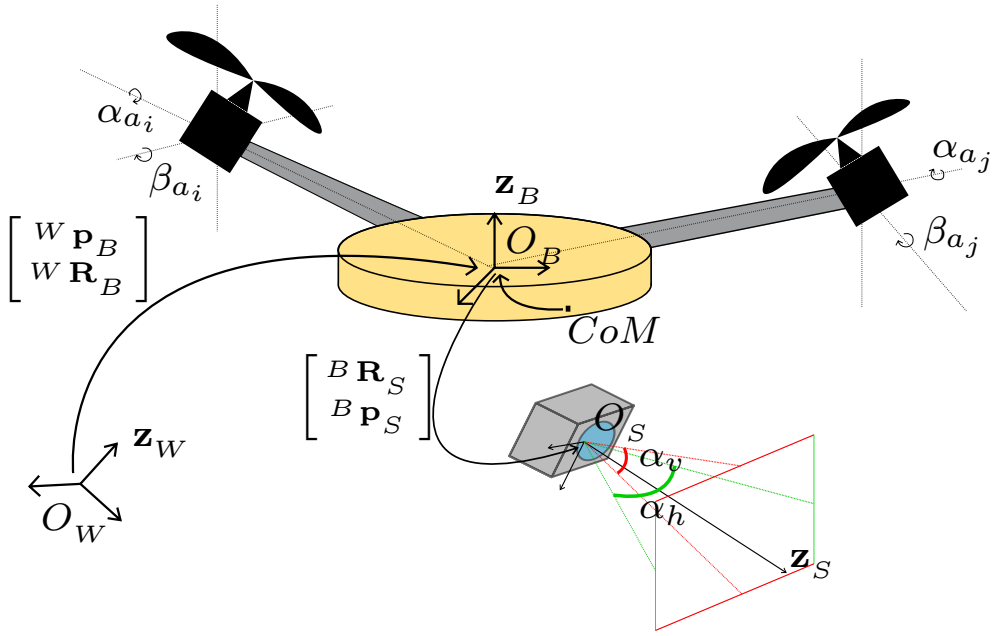


Figure 3.2: Schematic of the GTMR, depicting the frames and angles presented in Section 3.4.1, illustrated with two of the n propellers, i and j, with their respective tilting angles. A sensor (camera), as described in Section 3.5, is attached to the GTMR.

Finally, the dynamic equations of a GTMR are then given, dropping the reference frames for legibility, by

$$\dot{\mathbf{p}} = \mathbf{v}, \tag{3.21a}$$

$$\dot{\mathbf{q}} = \frac{1}{2}\,\mathbf{q} \otimes \begin{bmatrix} 0 \\ \boldsymbol{\omega} \end{bmatrix}, \tag{3.21b}$$

$$m\dot{\mathbf{v}} = -mg\mathbf{z}_W + {}^W\mathbf{R}_B\mathbf{G}_f\boldsymbol{\gamma}, \tag{3.21c}$$

$$\boldsymbol{I}\dot{\boldsymbol{\omega}} = -\boldsymbol{\omega} \times \boldsymbol{I}\boldsymbol{\omega} + \mathbf{G}_\tau\boldsymbol{\gamma} - {}^B\mathbf{p}_{CoM} \times {}^W\mathbf{R}_B^\top mg\mathbf{z}_W, \tag{3.21d}$$

$$\dot{\boldsymbol{\gamma}} = \mathbf{u}, \tag{3.21e}$$

where $\mathbf{G}_f$ and $\mathbf{G}_\tau \in \mathbb{R}^{3\times n}$ are the *force* and *torque control allocation matrices* [Michieletto, 2018] whose definitions are given in Section 3.4.2.

The offset between the geometrical center and the CoM affects the dynamics in two ways. First, the lever distance ${}^B\mathbf{p}_{CoM}$ induces an extra torque generated by the gravity force, which needs to be compensated in Equation (3.21d). Second, the inertia tensor is usually known w.r.t. the CoM, while $\boldsymbol{I}$ in Equation (3.21d) needs to be written w.r.t. $\mathcal{F}_B$. Such $\boldsymbol{I}$ can sometimes be obtained using a computer aided design (CAD) software. Otherwise, it can be computed from the tensor expressed w.r.t. the CoM, denoted $\boldsymbol{I}_{CoM}$, using Steiner's theorem [Siciliano, 2009]:

$$\boldsymbol{I} = \boldsymbol{I}_{CoM} - m[{}^B\mathbf{p}_{CoM}]_\times^2. \tag{3.22}$$

This offset ${}^B\mathbf{p}_{CoM}$ can be set to 0 as a first approximation. However, MPC are sensitive to such model approximations. Therefore, limiting those is important to improve the precision of the trajectory tracking. This offset is in general hard to obtain, except for a precise CAD modeling (including the data of proprietary parts, such as onboard computers). The design needs to be definitive, since adding a sensor

or moving cables would necessarily affect ${}^{B}\mathbf{p}_{CoM}$. Consequently, it is important to be able to estimate this quantity. A dedicated procedure is proposed in Appendix C.

Remark. Whilst the translational parts in Equation (3.21) are expressed in the inertial frame $\mathcal{F}_W$, the rotational dynamics (3.21d) are expressed in $\mathcal{F}_B$. This aims at simplifying the overall equation, in particular concerning the inertia tensor $\boldsymbol{I}$ which would be configuration-dependent otherwise. Consequently, the angular rates are expressed in $\mathcal{F}_B$ in the state vector, setting the quaternion derivative formula according to Equation (3.17).

3.4.2 GTMR Actuation

In order to generate the GTMR actuator commands, it is required to explicit the relation between the thrusts exerted by the n actuators and the resulting wrench applied to the body. Indeed, the actuator thrusts are the true controllable cause of the GTMR motion, yet the overall behavior is described at the level of the body.

Firstly, the resulting force applied to the body, ${}^{B}\mathbf{f}$, is the sum of the individual actuator forces γ_i, properly rotated in $\mathcal{F}_B$. On the other hand, the resulting torque ${}^{B}\boldsymbol{\tau}$ is the consequence of the torques created by the lever between the CoM and the actuators, plus the drag torques induced by the reaction of the air against the rotation of the propellers. For the *i-th* actuator, the lever torque ${}^{B}\boldsymbol{\tau}_i^{\gamma}$, can easily be expressed as a function of γ_i using a vector product:

$$ {}^{B}\boldsymbol{\tau}_i^{\gamma} = \left[{}^{B}\mathbf{p}_{A_i}\right]_{\times} {}^{B}\mathbf{R}_{A_i}\mathbf{z}_{A_i}\ \gamma_i, \tag{3.23} $$

where $\mathcal{F}_{A_i}$ is the frame attached with the *i-th* actuator.

The drag torque, denoted ${}^{B}\boldsymbol{\tau}_i^{d}$, is collinear with $\mathbf{z}_{Ai}$. Its magnitude is function of a constant parameter $c_{\tau,i}$, depending on the type of propeller. The subscript i recalls that the robot might have several types of propellers, each with specific coefficients. They are usually chosen to be the same, but this is not a requirement. This coefficient maps the rotor velocity with the generated rotor drag torque:

$$ {}^{B}\boldsymbol{\tau}_i^{d} = \pm \left\| {}^{B}\boldsymbol{\tau}_i^{d} \right\| {}^{B}\mathbf{R}_{A_i}\mathbf{z}_{A_i}, \tag{3.24a} $$

$$ \left\| {}^{B}\boldsymbol{\tau}_i^{d} \right\| = c_{\tau,i}\ \gamma_i. \tag{3.24b} $$

The direction of ${}^{B}\boldsymbol{\tau}_i^{d}$ is defined by the rotation direction of the propeller (positive for CCW rotation, negative for CW), which are typically chosen in an alternate pattern.

Then, the resulting torque is given by

$$ {}^{B}\boldsymbol{\tau} = \sum_i ({}^{B}\boldsymbol{\tau}_i^{d} + {}^{B}\boldsymbol{\tau}_i^{\gamma}). \tag{3.25} $$

Thus, there exists an algebraic relation between the actuator-generated body wrench, and the individual actuator thrusts. This mapping is only determined by the geometry of the GTMR (i.e., the number and positions of the actuators), and the type of propellers (through $c_{\tau,i}$). It is referred to as the *control allocation mapping*. More generically, this mapping is defined as a matrix $\mathbf{G} \in \mathbb{R}^{6\times n}$, for which the following relation holds:

$$ \begin{bmatrix} {}^{B}\mathbf{f} \\ {}^{B}\boldsymbol{\tau} \end{bmatrix} = \mathbf{G}\boldsymbol{\gamma} = \begin{bmatrix} \mathbf{G}_f \\ \mathbf{G}_\tau \end{bmatrix} \boldsymbol{\gamma}. \tag{3.26} $$

Combining Equations (3.25), (3.24) and (3.23), $\mathbf{G}$ can be expressed, column-wise, as

$$\mathbf{G}_j = \begin{bmatrix} {}^{B}\mathbf{R}_{A_j}\mathbf{z}_{A_i} \\ ([{}^{B}\mathbf{p}_{A_i}]_\times \pm c_{\tau,j}\mathbf{I}_3){}^{B}\mathbf{R}_{A_j}\mathbf{z}_{A_i} \end{bmatrix} \in \mathbb{R}^6, \tag{3.27a}$$

$$\mathbf{G} = \begin{bmatrix} \dots & \mathbf{G}_j & \dots \end{bmatrix}, \quad j \in \{1,\ n\}\,. \tag{3.27b}$$

This formalism allows for a control strategy called the inverse dynamics approach [Rajappa, 2015; Brescianini, 2016]. It consists of computing the desired control wrench, then deducing the desired individual propeller thrust to apply through the inversion – or pseudo-inversion – of the allocation matrix $\mathbf{G}$.

Remark. This approach links the generated body wrench with the propeller thrusts, which are in turn algebraically related to the propeller rotational speeds [Ryll, 2015]:

$$\gamma_i = c_{\gamma,i}\ \Omega_i^2, \tag{3.28}$$

where $\mathbf{\Omega}$ is the vector containing the n rotor angular speeds, and $c_{\gamma,i} \in \mathbb{R}$ is another fixed coefficient, characterizing the type of propeller used. Consequently, the propeller rotational accelerations $\dot{\mathbf{\Omega}}$ are directly related to $\dot{\boldsymbol{\gamma}}$ through the component-wise differentiation of Equation (3.28):

$$\dot{\gamma}_i = 2c_{\gamma,i}\ \Omega_i\dot{\Omega}_i, \tag{3.29}$$

Thus, an alternative allocation matrix $\mathbf{G}'$ could be written to map $\mathbf{\Omega}$ to the resulting body wrench. Both formulations are equivalent, even more so since the hardware implementations, which most often employ Electronic Speed Controllers (ESCs), require the conversion to rotor velocities at some point.

This approach is interesting since, as previously mentioned, the actuator thrusts γ_i are the true controllable causes of the GTMR motion. Considering $\boldsymbol{\gamma}$ in the scope of constrained optimal control allows to account for the motor-level limitations, which is physically meaningful. Indeed, [Franchi, 2018] shows the importance of keeping into account the rotor velocity constraints in the GTMR control, in order to preserve the system stability. Additionally, [Bemporad, 2009] claims that including the actuator dynamics in the GTMR modeling allows for improved performances. Indeed, the motor torques are the lowest-level inputs of the multi-rotor robots. However, the modeling of the actuator down to the motor torque – or its input current – complexifies the controller implementation. In fact, most hardware implementations make use of ESCs to control the motor velocity (through, e.g., a PID). The brushless motor current control is thus abstracted through the use of a (fast but not instantaneous) velocity control.

Thereby, a trade-off solution is proposed in [Geisert, 2016]. The authors propose to consider the propeller accelerations as system inputs, while their velocities are included in the system state. Indeed, the motor acceleration is directly linked, through a change of coordinates, to the motor torques. Doing so holds two main advantages:

- the simplistic assumption that the motor velocities can be changed instantaneously is dropped;

- constraints on the motor acceleration can be written to limit the motor torques.

Remark. A more precise approximation can be made by considering a first order response for the propeller acceleration control. Equation (3.20) then becomes

$$\dot{\gamma} = \frac{1}{t_i}(u_i - \gamma_i), \quad i \in \{1,\ n\}\,, \tag{3.30}$$

where t_i is a time constant of the actuator to be identified. This strategy is implemented in, e.g., [Nan, 2022], but the precision increase compared to a direct control assumption still needs to be evaluated. It is not used in the scope of this thesis.

Therefore, the state and input choice in Section 3.4.1 is motivated by these considerations. Further discussion on the actuation limits (including their estimation or identification) is provided in Chapter 4, when the constrained NLP formalism is addressed.

3.5 Sensor Model

3.5.1 Generic Sensor

We propose to model an onboard range-and-bearing sensor, as described in Section 2.3.1. Indeed, the range information is required to extrapolate the relative robot-object pose over the receding horizon. Thus, all of the subsequent development based on this assumption. Possible solutions to circumvent this limitation and exploit some (simpler to retrieve) bearing-only measurements are left out of the scope of this thesis.

It is modeled as a punctual device S, to which is attached a frame $\mathcal{F}_S$. This frame is defined with the z axis, $\mathbf{z}_S$, aligned with the sensor principal axis, i.e. the one describing its bearing. The $\mathbf{x}_S$ and $\mathbf{y}_S$ axes are set such that $\mathcal{F}_S$ is right-handed and they respectively define the horizontal and vertical directions of the sensor. The sensor is rigidly attached to the GTMR body such that the pose transformation between $\mathcal{F}_S$ and $\mathcal{F}_B$ is constant and known. The sensor FoV has a pyramidal shape centered around $\mathbf{z}_S$, defined by two halved FoV angles α_v and α_h along the vertical and horizontal axes. Moreover, the pyramidal shape is most often truncated by two planes ($z = \underline{d}$) and ($z = \overline{d}$), describing the minimum and maximum sensing distances.

Remark. For some sensors, the sensor FoV is a cone instead of a pyramid. In some cases (e.g., when $\alpha_v \approx \alpha_h$), it can also be approximated as conic. The cone is then defined by a single angle α, which simplifies the resulting equations, in particular when it comes to visibility constraints. Such approximation does not allow the system to exploit its full margin of action along the longest FoV axis and in the corners. Still, it can be motivated by remarking that some specific sensors typically have a squared FoV (e.g. some small onboard cameras). In such case, the corners of the pyramidal FoV would not be exploited, but the conic approximation would cover the larger part of the angular sectors defining the FoV.

The sensor is able to retrieve, in $\mathcal{F}_S$, the $3D$ pose of an object that falls into its FoV. This is achieved through some software processing, e.g., segmentation or CNN-based object detection techniques, which extract semantic information from the sensor row

data. The resulting *sensor + software* block provides the desired measurements. A Gaussian uncertainty is associated with those observations. Denoting the observed object M and the measurement vector $\mathbf{z}$, the observation model is written as

$$\mathbf{z} = {}^{S}\mathbf{p}_{M} + \boldsymbol{\eta}_{M}, \qquad \boldsymbol{\eta}_{M} \sim \mathcal{N}(0, \boldsymbol{\Sigma}_{M}), \tag{3.31}$$

where $\boldsymbol{\eta}_{M}$ is a 0-mean Gaussian noise of covariance matrix $\boldsymbol{\Sigma}_{M}$. The latter is defined as

$$\boldsymbol{\Sigma}_{M} = \begin{bmatrix} \sigma_x^2 & \sigma_{xy} & \sigma_{xz} \\ \sigma_{xy} & \sigma_y^2 & \sigma_{yz} \\ \sigma_{xy} & \sigma_{yz} & \sigma_z^2 \end{bmatrix}, \tag{3.32}$$

where $\sigma_\bullet \in \mathbb{R}$ are the cofactors along the x, y and z axes. Those are either fixed or measurement-dependent. In any case, $\boldsymbol{\Sigma}_{M}$ needs to be retrieved from the sensor datasheet, estimated through proper calibration, or estimated online [Tremblay, 2018]. It is indeed dependent on the software processing chosen to retrieve the measurement. The covariance is used to filter the sensor measurements, to palliate their low frequency (typically 10 to 60Hz).

The measurements are expressed either in the local frame $\mathcal{F}_{S}$ or in the inertial one $\mathcal{F}_{W}$. Yet, for practical reasons related to the measurement filtering, $\mathcal{F}_{W}$ is chosen since it allows to express the motion of the object using a simple linear model. Therefore, in the remaining of this manuscript, the localization of $\mathcal{F}_{S}$ in $\mathcal{F}_{W}$ is thus assumed to be provided along with its covariance matrix, used to propagate the measurement uncertainty to $\mathcal{F}_{W}$, using a first order approximation scheme.

3.5.2 Specific Case of a Camera

Although the sensor model aims at genericity, and encompasses various types of usual onboard exteroceptive sensors, a specific focus is made on monocular cameras, which hold a peculiar place in aerial robotics. As previously stated, these sensors are easiest to embark on a GTMR because of their lightweight, and thus are the most widely used in the research community, as well as for commercial drones. Monocular cameras are bearing only sensors, hence a suited software processing is required to fit the aforementioned model.

For convenience and coherence w.r.t. the literature, the camera – which is still considered a punctual device – is denoted C, with $\mathcal{F}_{C}$ its attached frame. The vertical and horizontal directions are chosen such that $\mathbf{x}_{C}$ and $\mathbf{y}_{C}$ are respectively collinear to $\mathbf{u}_{I}$ and $\mathbf{v}_{I}$, the horizontal and vertical unit vectors defining the image plane I.

The camera complies with the pinhole model, and its intrinsic calibration matrix $\mathbf{K} \in \mathbb{R}^{3\times3}$ is assumed known. Moreover, the images are assumed undistorted, avoiding reprojection errors in the peripheral FoV. The camera thus provides measurements $\mathbf{c} \in \mathbb{R}^2$ of 3D points ${}^{C}\mathbf{p} = \begin{bmatrix} p_x & p_y & p_z \end{bmatrix}^\top$ that fall into its FoV, following

$$p_z \begin{bmatrix} \mathbf{c} \\ 1 \end{bmatrix} = \mathbf{K}\ {}^{C}\mathbf{p}. \tag{3.33}$$

Remark. Although a simple 3D Gaussian noise can be applied on the final $3D$ measurement (i.e. obtained through software processing), the actual measurements of the camera are the pixels, and thus the Gaussian noise is applied on those. These

pixel measurements are subject to an isotropic Gaussian noise of standard deviation σ_C, such that the associated covariance matrix is

$$\mathbf{\Sigma_c} = \sigma_\mathbf{c}\mathbf{I}_2. \tag{3.34}$$

The 3D covariance matrix described in Equation (3.32) is to be deduced from $\mathbf{\Sigma_c}$ using a proper propagation scheme, which depends on the software processing. An actual example is provided in the upcoming Section 3.6.

3.6 Fiducial markers

Throughout this thesis, all the mathematical formulations will comply with the generic sensor model described in Section 3.5.1. The detection processes are left outside of the scope of this work, such that the sensors are considered as "black boxes", providing their measurements and covariances as defined in Equations (3.31) and (3.32). However, practical implementations are achieved through the use of monocular cameras. Hence, proper software processing is needed to retrieve the 3D pose estimates of the observed object.

CNN algorithms are nowadays the de facto standard for objects or features detection and recognition, as described in Section 2.3. CNN pose estimation algorithms that yield an uncertainty ellipsoid along with the estimate allows to fit the model from Section 3.5.1, as mentioned in Section 2.3.3. Otherwise, an arbitrary covariance should be designed. Yet, in order to avoid the use of such complex algorithms, whose implementation is heavy both in terms of computational power on the onboard PC and of training and tuning time, we operate a trade-off solution by using fiducial markers, such as AprilTags [Olson, 2011], AruCo [Garrido-Jurado, 2014] or WhyCon [Krajník, 2014]. It allows to abstract the detection at the small cost of additional hardware burden (i.e. adding such tags of objects to detect). These markers are both reliable and practical to use from a software point of view, with open libraries available. The OpenCV [Bradski, 2000] library offers for instance an implementation of the AruCo detection.

In the remaining of this thesis, we will refer to the fiducial markers indifferently as *features*, *landmarks* and *markers*. The word *feature* is here employed as a generic term to refer to an item of interest, and is not to be confused with the 2D image point of interest used, e.g., in VIO, and often referred to using the same word.

The pose estimation for square markers is based on the localization of the corners, followed by an algorithmic computation of ${}^C\mathbf{T}_M$, $\mathcal{F}_M$ being the frame attached to the marker. This assumes that the poses of the marker corners are known in $\mathcal{F}_M$, i.e. that the marker size is known. Such an algorithm is usually a PnP algorithm, as the ones mentioned in Section 2.3.3.

In this section, we propose a covariance propagation scheme based on [Fourmy, 2019], that describes accurately the pose estimate uncertainty of a square marker, such as AruCo. A visual representation of this can be found in Figure 3.3. A Maximum Likelyhood (ML) PnP is proposed in [Urban, 2016], which uses a careful parameterization of the homogeneous coordinates of the 2D pixel features to propagate the isotropic pixel noise through the PnP algorithm, using the Hessian matrix of the

problem. A simpler solution is brushed in [Fourmy, 2019] which uses a first order propagation scheme through the computation of the PnP projection Jacobian, i.e. the Jacobian of the mapping function from the 4 corners to the 6D pose:

$$f\colon \quad \begin{aligned} \mathbb{R}^8 &\to SE(3) \\ \begin{bmatrix}\mathbf{c}_1\\ \mathbf{c}_2\\ \mathbf{c}_3\\ \mathbf{c}_4\end{bmatrix} &\mapsto {}^{C}\mathbf{T}_{M} \end{aligned}, \tag{3.35}$$

However, such a mapping function is hard to differentiate. Instead, we rather consider its inverse $f^{-1}\colon\ SE(3) \to \mathbb{R}^8$, which is the perspective projection of the four corners. We note that f is the mapping from the 6D pose to the pixel vector, since the rotational part of the transformation is needed for the computation of the Jacobian, even if only the translational part is of interest to compute $\boldsymbol{\Sigma}_{\boldsymbol{M}}$.

We denote $\mathbf{J}_{f^{-1}} \in \mathbb{R}^{8\times 6}$ the Jacobian of f^{-1}, and $\mathbf{J}_{M} \in \mathbb{R}^{8\times 3}$ the Jacobian of the translational part, corresponding to the first three columns of $\mathbf{J}_{f^{-1}}$. Using Equation (3.34), a first order propagation scheme yields the relation

$$\sigma_{\mathbf{c}}^2 \mathbf{I}_8 = \mathbf{J}_{M} \boldsymbol{\Sigma}_{\boldsymbol{M}} \mathbf{J}_{M}^{\top}. \tag{3.36}$$

The Jacobian $\mathbf{J}_{M}$ is non-square but has full column rank, f being bijective. Therefore we have the following properties:

$$\mathbf{J}_{M}^{\dagger}\mathbf{J}_{M} = \mathbf{I}_8, \tag{3.37a}$$
$$\mathbf{J}_{M}^{\top}(\mathbf{J}_{M}^{\top})^{\dagger} = \mathbf{I}_3, \tag{3.37b}$$
$$\mathbf{J}_{M}^{\dagger}(\mathbf{J}_{M}^{\top})^{\dagger} = (\mathbf{J}_{M}^{\top}\mathbf{J}_{M})^{\dagger} = (\mathbf{J}_{M}^{\top}\mathbf{J}_{M})^{-1}. \tag{3.37c}$$

Using all the above, Equation (3.36) is inverted and further simplified as

$$\cancel{\mathbf{J}_{M}^{\dagger}}\cancel{\mathbf{J}_{M}}\,\boldsymbol{\Sigma}_{\boldsymbol{M}}\,\cancel{\mathbf{J}_{M}^{\top}}\cancel{(\mathbf{J}_{M}^{\top})^{\dagger}} = \sigma_{\mathbf{c}}^2 \mathbf{J}_{M}^{\dagger}(\mathbf{J}_{M}^{\top})^{\dagger} \tag{3.38a}$$
$$\boldsymbol{\Sigma}_{\boldsymbol{M}} = \sigma_{\mathbf{c}}^2 (\mathbf{J}_{M}^{\top}\mathbf{J}_{M})^{-1}. \tag{3.38b}$$

To compute $\mathbf{J}_{f^{-1}}$, and thus $\mathbf{J}_{M}$, we use the known four corners coordinates in the marker frame $\mathcal{F}_{M}$, denoted $\mathbf{x}_i,\ \ i \in \{1,\ 4\}$, defined as

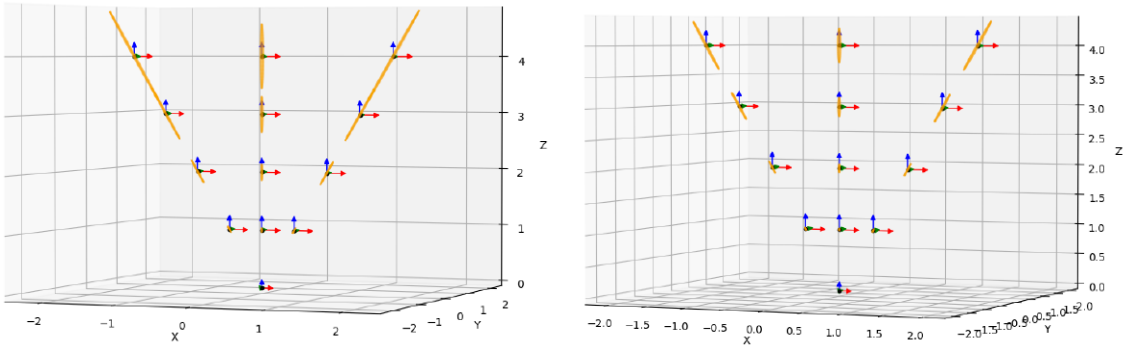
$$\mathbf{x}_i = \begin{bmatrix}\pm l/2\\ \pm l/2\\ 0\end{bmatrix}, \tag{3.39}$$

where l is the marker size. For each corner, we have the relation

$$\mathbf{c}_i = pix(\mathbf{h}_i) = pix(\mathbf{K}({}^{C}\mathbf{R}_{M}\mathbf{x}_i + {}^{C}\mathbf{p}_{M})), \tag{3.40}$$

where $\mathbf{h}_i \in \mathbb{R}^3$ are the homogeneous coordinates obtained from the perspective projection of the camera, and $pix\colon \mathbb{R}^3 \to \mathbb{R}^2$ is the *pixelization* of $\mathbf{h}_i$ into the pixel coordinates, defined as

$$pix\colon \quad \begin{bmatrix} x & y & z\end{bmatrix}^{\top} \mapsto \begin{bmatrix} x/z \\ y/z\end{bmatrix}. \tag{3.41}$$



(a) Uncertainty ellipsoids for 15 cm tags.

(b) Uncertainty ellipsoids for 30 cm tags.

Figure 3.3: Graphical depiction of the position uncertainty ellipsoid corresponding to $\mathbf{\Sigma}_M$ (in orange) (courtesy of [Fourmy, 2022]). The camera is centered in $(0, 0, 0)$, and the ellipsoids are centered on the corresponding tag positions. Two different sizes are used, illustrating how it impacts the resulting uncertainty. It is noticeable that the largest uncertainty is associated with the camera range.

The derivation of (3.40) gives, using the chain rule, the Jacobian:

$$\mathbf{J}_{\mathbf{T}}^{\mathbf{c}_i}(\mathbf{x}_i) = \mathbf{J}_{\mathbf{h}_i}^{\mathbf{c}_i}\, \mathbf{J}_{\mathbf{T}}^{\mathbf{h}_i}(\mathbf{x}_i) \in \mathbb{R}^{2\times 6}, \tag{3.42a}$$

with:

$$\mathbf{J}_{\mathbf{T}}^{\mathbf{h}_i}(\mathbf{x}_i) = \mathbf{K}\begin{bmatrix}\mathbf{I}_3 & -{}^{C}\mathbf{R}_{M}[\mathbf{x}_i]_{\times}\end{bmatrix} \in \mathbb{R}^{3\times 6}, \tag{3.42b}$$

$$\mathbf{J}_{\mathbf{h}_i}^{\mathbf{c}_i} = \begin{bmatrix}1/z & 0 & -x/z^2 \\ 0 & 1/z & -y/z^2\end{bmatrix}. \tag{3.42c}$$

Finally, the four Jacobian matrices $\mathbf{J}_{\mathbf{T}}^{\mathbf{c}_i}$ computed for the four corners are stacked into the full Jacobian $\mathbf{J}_{f^{-1}}$.

The measurement covariance in the desired frame $\mathcal{F}_W$ is obtained by further propagating $\mathbf{\Sigma}_M$ using the Jacobians of the rototranslation ${}^{W}\mathbf{T}_{B}$.

Remark. The last three columns of $\mathbf{J}_{f^{-1}}$ contain the uncertainty of the orientation ${}^{B}\mathbf{q}_{M_i}$ expressed as an "orientation element" of dimension 3, and not as quaternion $\in \mathbb{Q}$. In order to retrieve the covariances of the 4 individual quaternion elements, these 3 columns need to be transformed using the Jacobian of the Exponential map [Solà, 2018]. We remark that this propagation model captures well the local maximum arising when the marker is frontoparallel to the camera, justifying its pertinence.

3.7 Sensor measurement filtering

After the detection is issued, a filtering step is applied to the measurements. This serves three objectives: first, it allows to increase the precision by reducing the impact of poor, false or missing detections; second, it allows to exploit estimations in-between the measurements, which frequency are typically quite low ($\sim$ 50 Hz) versus the control frequency ($\sim$ 500 Hz); and third, it allows to extrapolate the current feature state for the near future. The latter is indeed of use to improve the accuracy of predictive controllers.

Such filtering requires a proper modeling of the landmark motion. The feature evolves freely in the inertial frame $\mathcal{F}_W$. A linear Gaussian model is typically well suited to describe the motion of 3D point. Its motion is also assumed to be governed by an unknown underpinning process, since the robot is agnostic of its nature. Therefore, the system inputs are not considered in the filtering.

Remark. There exist, in the literature, some strategies to predict the motion of humans [Martinez, 2017] or human-driven vehicles [Anderson, 2020; Qin, 2021]. These are either model-based or deep learning-based. Such policies could be used to extend the proposed filtering process in specific scenarios, introducing an estimation of the human inputs based on its past motion.

Finally, in order to describe accurately the motion and allow meaningful extrapolation over a short period of time, a constant acceleration assumption is made. It provides a good compromise between being accurately descriptive, and reasonably observable through position-only measurements.

Hence, the feature M is described as a system whose discrete motion model is

$$\mathbf{x}_M = \begin{bmatrix} {}^W\mathbf{p}_M \\ {}^W\mathbf{v}_M \\ {}^W\mathbf{a}_M \end{bmatrix}, \tag{3.43}$$

$$\mathbf{x}_{M,k+1} = \mathbf{A}\mathbf{x}_{M,k} + \boldsymbol{\eta}_\mathbf{Q}, \quad \boldsymbol{\eta}_\mathbf{Q} \sim \mathcal{N}(0, \mathbf{Q}), \tag{3.44}$$

where $\mathbf{x}_M$ is the feature state, and $\boldsymbol{\eta}_\mathbf{Q}$ is the 0-mean Gaussian noise of covariance $\mathbf{Q}$, and $\mathbf{A}$ is the transition matrix. In case of a constant acceleration model, we have

$$\mathbf{A} = \begin{bmatrix} \mathbf{I}_3 & \delta t\mathbf{I}_3 & \frac{1}{2}\delta t^2\mathbf{I}_3 \\ \mathbf{O}_3 & \mathbf{I}_3 & \delta t\mathbf{I}_3 \\ \mathbf{O}_3 & \mathbf{O}_3 & \mathbf{I}_3 \end{bmatrix}, \tag{3.45}$$

δt being the discrete time step.

To filter $\mathbf{x}_M$, a linear KF is used, to which position measurements are provided. As described in Equation (3.31), the observation model is

$$\mathbf{z}_k = \begin{bmatrix} \mathbf{I}_3 & \mathbf{O}_3 & \mathbf{O}_3 \end{bmatrix} \mathbf{x}_{M,k} + \boldsymbol{\eta}_\mathbf{R}, \quad \boldsymbol{\eta}_\mathbf{R} \sim \mathcal{N}(0, \mathbf{R}), \tag{3.46}$$

where $\mathbf{z}$ is the measurement, and $\boldsymbol{\eta}_\mathbf{R}$ is a 0-mean Gaussian noise of covariance $\mathbf{R}$.

Remark. When exploiting the KF to extrapolate the feature motion over a long receding horizon (e.g., 1 s), it might be desirable to prevent the estimation from diverging far from its current state, in particular since the instantaneous acceleration estimated from the position measurements might be inaccurately high. This is motivated by the idea that an object will not continuously accelerate or brake in the future. In that case, a similar model might be used by applying a damping on the acceleration through a scalar $\lambda \in]0,\ 1[$, which will reduce the acceleration over successive predictions. Thus, the transition model becomes

$$\mathbf{A} = \begin{bmatrix} \mathbf{I}_3 & \delta t\mathbf{I}_3 & \frac{1}{2}\delta t^2\mathbf{I}_3 \\ \mathbf{O}_3 & \mathbf{I}_3 & \delta t\mathbf{I}_3 \\ \mathbf{O}_3 & \mathbf{O}_3 & \lambda\mathbf{I}_3 \end{bmatrix}. \tag{3.47}$$

Part II

Perception Awareness in N-MPC

Chapter 4

Generic Problem Statement for Generically Tilted Multi-Rotor

Contents

4.1 Introduction **56**
4.2 The choice of N-MPC **57**
4.3 Problem Statement **58**
4.4 NonLinear Programming **59**
4.4.1 Actuation Constraints 59
4.4.2 Motion Objective 60
4.4.3 NLP Formulation 61
4.5 Conclusion **62**

4.1 Introduction

In the various applications where GTMR are employed, some of which are presented in Section 1.1, the task assigned to the AR is usually to follow a trajectory (or a set of waypoints), while performing a given activity of interest (monitoring an object or an area, interacting with the environment...).

In the scope of this thesis, such activity is perception-oriented. In particular, we focus on the visibility coverage of a set of objects or phenomena.

The task assigned to the GTMR can thus be divided in two parts:

1. the *motion task*, defining the trajectory to follow, or the position to stay at. It is usually defined as a reference value for the positions, orientations, and their derivatives. Such a task also encloses the objective of maintaining the stability of the system over time;
2. the *perception task*, depicting the perception activity that the AR has to achieve.

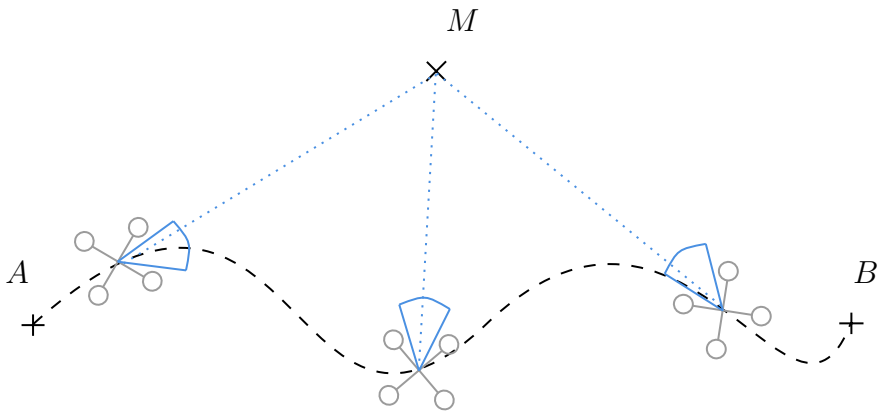


Figure 4.1: Top-view of a typical task assigned to the AR. It is required to go from A to B following the black dashed line, while keeping visibility over the feature of interest M. Three position samples along this path are pictured in gray, while the blue triangle denotes the sensor FoV.

We remark that these tasks can be multiple. The perception task might be to maintain coverage over several features, or the motion task might be to follow a trajectory while maintaining a given orientation. In addition to those, some additional tasks might be expressed, in order to, e.g., maintain a safety distance w.r.t. the environment or the other agents in the workspace.

Finally, the GTMR has to perform these tasks while being subject to several constraints. The most immediate constraints to be formulated are the motion constraints, prohibiting the AR to leave a given workspace or to collide with other agents. Other constraints can be stated, e.g., regarding the actuation limits of the GTMR.

This chapter proposes to

1. discuss and motivate the choice of N-MPC controllers in the specific context of such motion/perception tasks,

2. propose a unified framework to express the NLP for GTMR under generic perception-related constraints and objectives,
3. and include therein the formalization of the actuation limitations, exploiting previous related works.

4.2 The choice of N-MPC

Optimal control policies are powerful tools to consider constraints and objectives from various semantics, e.g., perception- and motion-related. These are mathematically formalized and jointly included in the OCP. From there, several policies can be employed to retrieve the optimal solution, as discussed in Chapter 2. Among these, MPC has proven to be a very efficient technique for constrained systems with fast dynamics. Section 2.6 purveys an overview of recent works which exploit MPC in the scope of perception-aware controllers.

In addition to the various advantages of (N-)MPC, our interest in this control policy is that it allows to consider the system evolution over the near future, which yields non-myopic decision-making. This is of prime importance in the scope of fulfilling the perception task with the AR. For instance, when observing a moving object with an underactuated quadrotor, the position-orientation coupling implies that the quadrotor must counter-tilt to catch up with the object motion, which leads to visibility loss. This implies overcoming a local minimum, since the counter-tilting increases the cost function value for the first sampling points. Thus, a greedy approach is not suited for fulfilling this task,

In the literature, MPC is most often used either to locally plan trajectories [Penin, 2017] that tracked with another cascaded controller; or to produce a desired orientation as virtual input which regulation is also left for a low-level regulator [Darivianakis, 2014; Baca, 2016]. The resulting motion is as close as possible to the computed one, but the inner-loop rarely encapsulates the full complexity of the AR dynamics. Moreover, the previously cited works made use of linear MPC, in which the linearization of the AR dynamics (e.g., based on differential flatness in [Penin, 2017]) pushes even further the discrepancy between the planned trajectory and the actual motion. As opposed to the two-step approach used in [Penin, 2017; Penin, 2018], which decouples planning and control, the N-MPC in [Falanga, 2018] is used to compute the attitude and thrust inputs of the platform. In these two approaches, despite the use of N-MPC, the low-level control remained unconstrained. Henceforth, the resulting motion is not fully satisfactory, as it does not provide the best insurance regarding the constrained fulfillment of the task.

On the other hand, in the vein of [Bicego, 2020], full-state N-MPC allows to compute directly the constrained low-level inputs of the actuators. Such a scheme allows to account for the various requirements of the tasks, and the produced behavior is guaranteed to fulfill the constraints – up to the accuracy allowed by the model and the state estimation. We claim that this approach can be combined with the aforementioned perception-aware control techniques such that the constraints are ensured throughout the entire control scheme.

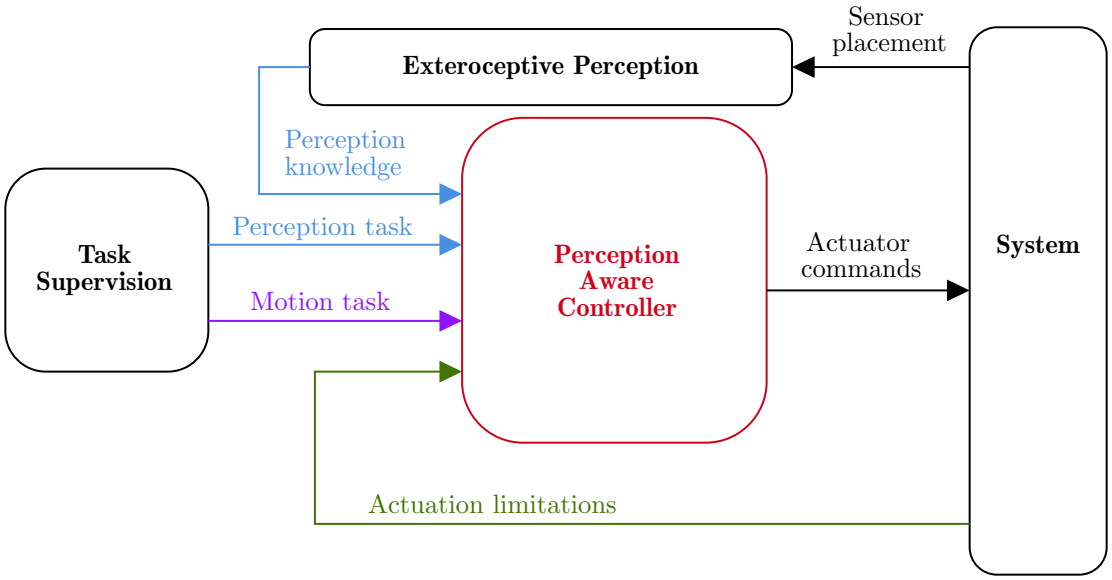


Figure 4.2: A partial conceptual diagram of the proposed perception-aware controller paradigm. The quantities from various domains (namely, perception, motion and actuation) are depicted with different colors, respectively blue, purple and green.

4.3 Problem Statement

In the scope of this thesis, we present a perception-aware N-MPC that performs the low-level (i.e. the motor-torque-level) control of a GTMR. The GTMR is assigned to a trajectory tracking activity, and is subject to actuation constraints, expressed in terms of minimum and maximum propeller thrusts and thrust derivatives. To this end, we exploit the nonlinear model of the GTMR presented in Chapter 3, following the paradigm introduced in [Bicego, 2020].

Additionally, the GTMR is tasked to achieve perception-related activities, and the N-MPC is consequently subject to objectives and constraints arising from the perception domain, i.e. that are deduced through appropriate processing of sensory data. The perception model is described in Chapter 3, including the associated filtering. The latter is used to provide a model-based extrapolation of sensory data over the receding horizon, which are combined with the predictive aspect of the controller. The diverse formulations taken by the perception constraints and objectives, the main contribution of this thesis, are presented in depth in the subsequent chapters.

A schematic overview of the problem definition is depicted in Figure 4.2. Therein, the input quantities of the proposed controller (red block) are coming from various semantics, depicted with various colors. The knowledge assessed from exteroceptive perception is used to comply with the pre-defined perception task. Similarly, the motion task and actuation limitations are defining the set of motion- and actuation-related constraints and objectives. This schematic representation is obviously simplified and thus incomplete, as many blocks or connections are missing. One would e.g. add a proprioceptive perception block, enabling the controller to close the loop, and which is also mandatory in order to assess the actuation limits of the platform. Additionally, the tunable controller weights could be added as an input

of the system, which would act as the direct means of control for the supervisor. Such figures roughly depict the main concepts that are aggregated into our proposed perception-aware control scheme. A more rigorous block diagram representation is provided in Figure 4.3.

In the remaining of the manuscript, we distinguish between the motion objectives, denoted C^{motion}, and the perception objectives, denoted $C^{\text{perception}}$.

4.4 NonLinear Programming

4.4.1 Actuation Constraints

Following the actuation parameterization proposed in Section 3.4.2, this section recalls a definition of the resulting constraints imposed on the system.

The propellers of the GTMR are subject to the physical limitations of any moving (more specifically, rotating) body, and are therefore subject to inertia. Additionally, the motors can only receive a limited amount of electrical current, hence produce a (both lower and upper) bounded torque. These considerations translate into a finite acceleration limit for the propellers, thus in a finite limit for $\dot{\boldsymbol{\gamma}}$. Finally, the rotational speed of the propeller is of course bounded due to the always present dissipative effects (friction, propeller air drag, etc).

In order to account for the complex relation between these limitations and their effect of the GTMR body, it is not sufficient to apply constraints on the body state, e.g. limiting angular rates and total propeller thrust. Such strategy is often employed in aerial robotics when using cascaded control strategies [Darivianakis, 2014; Alexis, 2016; Foehn, 2018]. In particular, this implies the necessity of limiting some state variables (namely, those interfacing the MPC and the inner regulator) with some fictitious bounds, i.e. not motivated by a physical meaning, but rather by a *ad hoc* heuristic. Such limitations are conservative w.r.t. the AR motion and its real dynamics are therefore not exploited. Moreover, this strategy is not suited for full-state controllers, since the absence of regulator might result in actuator saturation – hence in an unfeasible motion – and also exposes to the risk of burning the motor by applying a very large current. Thereby, it is mandatory to account for actuator limitations.

As discussed in Section 3.4, accounting the complete dynamic model of the motor-actuator pair has severe implementability drawbacks. In particular, this renders difficult to define accurate limitations in terms of minimum and maximum torques, which depends on multiple parameters (friction, the air drag, the ESCs software and hardware, etc). Following the trade-off solution introduced in [Geisert, 2016],a simplified dynamic model for the propellers was presented in Section 3.4.

Using this model, the physical limitations in terms of motor torques and rotational speed are equivalently recast as constraints on the GTMR actuator state and control inputs, i.e. on $\boldsymbol{\gamma}$ and $\dot{\boldsymbol{\gamma}}$:

$$\underline{\boldsymbol{\gamma}} \leq \boldsymbol{\gamma} \leq \overline{\boldsymbol{\gamma}}, \tag{4.1a}$$

$$\underline{\dot{\boldsymbol{\gamma}}}(\boldsymbol{\gamma}) \leq \dot{\boldsymbol{\gamma}} \leq \overline{\dot{\boldsymbol{\gamma}}}(\boldsymbol{\gamma}). \tag{4.1b}$$

Remark. Vector inequalities are intended component-wise throughout the manuscript.

Using the algebraic relation between the propeller thrust and rotational speed (see Equation (3.28)), we obtain the bounds on the individual propeller thrust. The lower bounds, $\underline{\boldsymbol{\gamma}}$, is directly related to the minimum rotational speed of the motors (typically close to 0, according to the design of the motor and its electrical controller). The upper bound $\overline{\boldsymbol{\gamma}}$ is related to the rotational speed achieved at steady state when applying the maximum torque to the motor.

Similarly, the bounds on the time derivative of the thrusts $\dot{\boldsymbol{\gamma}}$ are algebraically related to the minimum and maximum accelerations of the propellers, achieved respectively when applying the minimum and maximum motor torques. Such limits also depend on the inertia of the propeller, the friction and the air drag, which in turn depend on the propeller speed. This justifies the dependency on $\boldsymbol{\gamma}$ in Equation (4.1b). Such bounds are obtained through an identification campaign, as detailed in [Bicego, 2020].

We note that these are the only real physical constraints applied to the system. Other potential constraints on the system state, such as limitations on the linear or angular velocities, can only artificially limit the range of capabilities of the platform, and would be contingent to a specific task or context.

4.4.2 Motion Objective

Following the problem defined in Section 4.3, the motion task assigned to the GTMR can be expressed as a reference trajectory to follow. Such a trajectory is computed using an external motion planner, whose design is left out of the scope of this thesis. To exploit the predictive aspect of the N-MPC, this trajectory is usually provided through the complete receding horizon. It is expressed in terms of reference position and orientation $(\mathbf{p}_r, \mathbf{q}_r)$, together with their first and second order time derivatives.

Then the motion objective is defined as the minimization of the distance to this reference. To this end, we define an output map $\mathbf{y}$ and its reference $\mathbf{y}_r$, as

$$\mathbf{y} = [\mathbf{p}^\top \ \mathbf{q}^\top \ \dot{\mathbf{p}}^\top \ \boldsymbol{\omega}^\top \ \ddot{\mathbf{p}}^\top \ \dot{\boldsymbol{\omega}}^\top]^\top, \tag{4.2a}$$

$$\mathbf{y}_r = [\mathbf{p}_r^\top \ \mathbf{q}_r^\top \ \dot{\mathbf{p}}_r^\top \ \boldsymbol{\omega}_r^\top \ \ddot{\mathbf{p}}_r^\top \ \dot{\boldsymbol{\omega}}_r^\top]^\top, \tag{4.2b}$$

where the subscript $\bullet_r$ indicates a reference value.

The error w.r.t. the reference is defined as a weighted squared Euclidean norm of the difference:

$$\|\mathbf{y} - \mathbf{y}_r\|^2_{\mathbf{W}_\mathrm{m}} = (\mathbf{y} - \mathbf{y}_r)^\top \mathbf{W}_\mathrm{m} \, (\mathbf{y} - \mathbf{y}_r), \tag{4.3}$$

where $\mathbf{W}_\mathrm{m}$ is the diagonal weight matrix, which acts as the tunable controller gains.

Nevertheless, the Euclidean distance between two unit quaternions is not suitable to represent the dissimilarity between two orientations, mainly because $\mathbf{q}$ and $-\mathbf{q}$ represent the same orientation in $SO(3)$. Following [Huynh, 2009], there are at least 6 norms that can be defined on the orientation space $SO(3)$. The most immediate one is the geodesic distance on the manifold $\mathbb{S}$ ([Park, 1995; Park, 1997]):

$$d_1(\mathbf{q}, \mathbf{q}_r) = \left\| \mathrm{Log}(\mathbf{q}^{-1} \otimes \mathbf{q}_r) \right\| \ \in [0, \pi[. \tag{4.4}$$

which is in fact equal to the absolute value of half the angular distance between two quaternions. This function is however computationally complex due to the quaternion logarithm. Alternatively, in some robotics algorithmic frameworks [Carpentier, 2019], a lighter, more computationally efficient, form of the orientation distance can be expressed as the dot product of unit quaternions:

$$d_2(\mathbf{q}, \mathbf{q}_r) = \arccos(|\mathbf{q} \cdot \mathbf{q}_r|) \ \in [0, \frac{\pi}{2}], \tag{4.5}$$

or in a substitute form that does not employ any trigonometric function [Kuffner, 2004]:

$$d_3(\mathbf{q}, \mathbf{q}_r) = 1 - |\mathbf{q} \cdot \mathbf{q}_r| \ \in [0, 1]. \tag{4.6}$$

Such a function provides a pseudometric on the unit quaternion space $\mathbb{S}$, but is a properly defined metric on $SO(3)$. In the following, for the sake of readability, we will keep the notation $\|\mathbf{q} - \mathbf{q}_r\|^2_{\mathbf{W}_\mathrm{m}}$ to refer to the weighted attitude error associated with any of these distances.

Using the aforementioned definitions, we can write

$$C^{\,\mathrm{motion}} = \|\mathbf{y} - \mathbf{y}_r\|^2_{\mathbf{W}_\mathrm{m}}\,. \tag{4.7}$$

4.4.3 NLP Formulation

Finally, the NLP is written as a minimization problem over the state and input variables $\mathbf{x}$ and $\mathbf{u}$.

The cost function of this problem is the weighted summation, over the receding horizon, of the two objectives $C^{\,\mathrm{motion}}$ and $C^{\,\mathrm{perception}}$.

An extra term can be added to this cost function in order to provide a reference value for the system inputs $\mathbf{u}$. Given the model presented in Section 3.4, the reference value for the system inputs (Equation (3.20)) is logically 0, since the optimal steady hovering state is reached at steady hovering thrusts. The minimization of $\mathbf{u}$ in the N-MPC implies the minimization of the motor torques, hence the reduction of the energy consumption of the system. However, it is common, as mentioned in [Bicego, 2020], to have a small (or even zero) weight on the input. This allows to exploit at best the actuation capabilities of the robot. We denote this objective $C^{\,\mathrm{inputs}}$, which is defined as

$$C^{\,\mathrm{inputs}} = \|\mathbf{u}\|^2_{\mathbf{W}_\mathrm{i}}\,, \tag{4.8}$$

where $\mathbf{W}_\mathrm{i}$ is the corresponding tunable weight matrix.

The minimization problem is subject to a set of equality constraints, defining

1. the state initialization (see Equation (4.11b)),
2. the system dynamics (Equation (4.11c)),
3. the system output map (Equation (4.11d)),
4. the equality mapping for the perception objective (Equation (5.19e)).

Additionally, multiple inequality constraints are expressed to account for the limitations applied to the body motion, on the actuation, or induced by the perception. The constraints are expressed as functions of the N-MPC state $\mathbf{x}$, as well as external

parameters $\mathbf{p}$. The actuation constraints are expressed following Inequation (4.1). The motion constraints are contingent on each specific task, and are generically denoted using the mapping

$$\underline{\boldsymbol{\mu}} \;\leq\; \boldsymbol{\mu}(\mathbf{x},\mathbf{p}) \;\leq\; \overline{\boldsymbol{\mu}}. \tag{4.9}$$

Similarly, the perception bounds, which will be detailed in the subsequent chapters, are generically denoted using the mapping

$$\underline{\boldsymbol{\psi}} \;\leq\; \boldsymbol{\psi}(\mathbf{x},\mathbf{p}) \;\leq\; \overline{\boldsymbol{\psi}}. \tag{4.10}$$

The NLP, expressed over the receding horizon T, is discretized in N shooting points.

Consequently, at given instant t, the NLP is finally written as

$$\min_{\substack{\mathbf{x}_0\dots\mathbf{x}_N\\ \mathbf{u}_0\dots\mathbf{u}_{N-1}}} \quad \sum_{k=0}^{N} \|\mathbf{y}_k - \mathbf{y}_{r,k}\|^2_{\mathbf{W}_\mathrm{m}} + \sum_{k=0}^{N} C_k^{\text{perception}} + \sum_{k=0}^{N-1} \|\mathbf{u}_k\|^2_{\mathbf{W}_\mathrm{i}} \tag{4.11a}$$

$$\begin{aligned}
s.t.\quad & \mathbf{x}_0 = \mathbf{x}(t) && && (4.11b)\\
& \mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k,\mathbf{u}_k,\mathbf{p}_k), && k\in\{0,\,N-1\} && (4.11c)\\
& \mathbf{y}_k = \mathbf{h}(\mathbf{x}_k), && k\in\{0,\,N\} && (4.11d)\\
& C_k^{\text{perception}} = \mathbf{g}(\mathbf{x}_k,\mathbf{p}_k), && k\in\{0,\,N\} && (4.11e)\\
& \underline{\boldsymbol{\gamma}} \leq \boldsymbol{\gamma}_k \leq \overline{\boldsymbol{\gamma}}, && k\in\{0,\,N\} && (4.11f)\\
& \underline{\dot{\boldsymbol{\gamma}}}(\boldsymbol{\gamma}_k) \leq \mathbf{u}_k \leq \overline{\dot{\boldsymbol{\gamma}}}(\boldsymbol{\gamma}_k), && k\in\{0,\,N-1\} && (4.11g)\\
& \underline{\boldsymbol{\mu}_k} \;\leq\; \boldsymbol{\mu}(\mathbf{x}_k,\mathbf{p}_k) \;\leq\; \overline{\boldsymbol{\mu}_k}, && k\in\{0,\,N\} && (4.11h)\\
& \underline{\boldsymbol{\psi}_k} \;\leq\; \boldsymbol{\psi}(\mathbf{x}_k,\mathbf{p}_k) \;\leq\; \overline{\boldsymbol{\psi}_k}, && k\in\{0,\,N\} && (4.11i)
\end{aligned}$$

where $\mathbf{x}(t)$ is the measurement of the state at time t, $\mathbf{f}$ synthetically denotes the dynamics of the GTMR expressed in Equation (3.21), which are discretized using any integration scheme. Finally, $\mathbf{h}$ denotes the system output map defined in (4.2), and $\mathbf{g}$ abstracts the perception objective mapping.

A block diagram for the proposed framework is presented in Figure 4.3.

4.5 Conclusion

This section proposes a N-MPC paradigm for perception-constrained control of a GTMR. Based on the idea that an AR tasks are decoupled into motion and perception, the proposed formulation is stated generically and can be used in numerous applications. Accounting for the low-level actuation constraints and computing the direct motor commands ensures that the computed motion is feasible by the AR, as opposed to the most widespread usage of N-MPC, e.g. in cascaded control.

Remark. This relies on the assumption that the flight controller, and in particular the onboard ESCs, allow for a sufficiently fast velocity control of the rotors. In practice, the bounds on $\dot{\boldsymbol{\gamma}}$ are obtained through an hardware identification campaign [Bicego,

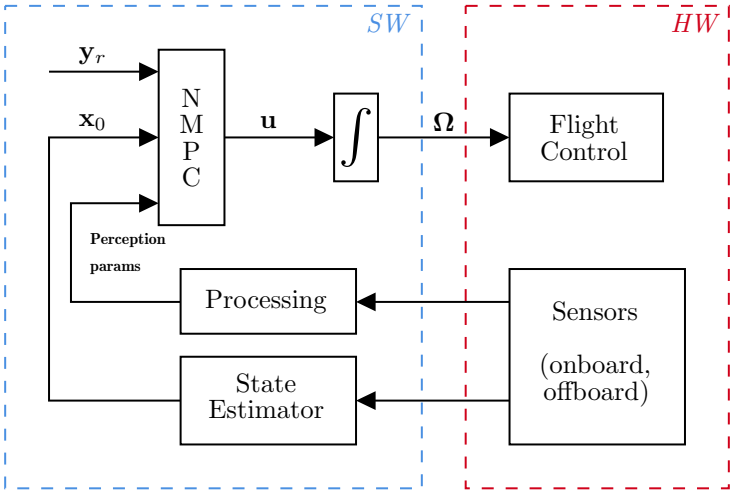


Figure 4.3: Block diagram of the N-MPC framework. The software (SW) and hardware (HW) domains are depicted by the color blocks. The system input $\mathbf{u}$ are integrated and converted to rotor velocities $\mathbf{\Omega}$.

2020], and are evaluated with the onboard ESCs. Hence, the convergence time of the embedded velocity control should be, at least partially, considered therein.

In the following chapters, this formulation will be instantiated to solve a couple of common applications for ARs: static or dynamic object tracking, and visual state estimation.

Chapter 5

Perception constrained N-MPC

Contents

5.1 Introduction **66**
5.2 Geometric Perception Criterion **66**
5.3 Perception Objectives **67**
5.4 FoV Constraints **69**
5.4.1 Range Constraint 70
5.4.2 Conic FoV 70
5.4.3 Pyramidal FoV 71
5.4.4 Relaxation of Constraint 72
5.5 NonLinear Programming Formulation **73**
5.6 Extension to Multiple Features and Sensors **74**
5.7 Experimental and Simulation results **75**
5.7.1 Near Hovering while Observing a Circular Motion 76
5.7.2 Hover-to-hover Under Visibility Constraints 77
5.7.3 Mobile Feature Tracking 78
5.7.4 Simulation with a Tilted-Propeller Hexarotor 80
5.8 Application to Human-Robot Handover **81**
5.8.1 Presentation 81
5.8.2 Simulation 83
5.9 Conclusion **84**
5.9.1 Synthesis 84
5.9.2 Limitations 84

5.1 Introduction

Following the paradigm introduced in Chapter 4, the perception-related aspects of the tasks can be integrated into the framework as constraints and objectives in the N-MPC. Such aspects are dependent on multiple factors, e.g. the task to fulfill, or the amount and types of the various perceptive sensors equipped on the AR.

This chapter introduces the formulation of the perception objectives $C^{\text{perception}}$, as well as the sensing constraints $\boldsymbol{\psi}$ to which the system is subject. We validate the proposed approach in simulations and experiments both with standard collinear quadrotors and tilted-propeller hexarotors, underlying the capability of the proposed controller to exploit its full actuation span to perform at best the various objectives.

First, a geometrical expression of the visibility is proposed. Second, the perception constraints and objectives are expressed, and the complete NLP is formalized. Then, the system behavior in various simulations and experiments is reported. Finally, an additional practical application of this framework is presented.

For the sake of readability, the mathematical formulation is first introduced for the simplest case, i.e. the tracking of a single feature with a single sensor. Later, the notations are extended to the scope of multiple features and sensors, and the associated considerations are discussed.

The contributions of this chapter are:

- A geometrical condition for the visibility in a pyramidal-shaped sensor,
- A motor-level perception-aware N-MPC for GTMR,
- A fully onboard implementation and its validation in simulations and experiments.

The work presented hereafter led to two publications: [Jacquet, 2020] and [Jacquet, 2021]. A collaborative work exploiting this N-MPC, presented in Section 5.8: [Corsini, 2022].

5.2 Geometric Perception Criterion

In order to assess how the AR motion would affect the sensor visibility, it is mandatory to express a geometric relation between the sensor 6D pose and the object position. Given the feature model described in Chapter 3, the object orientation is not accounted for. The quantity of interest is thus ${}^{S}\mathbf{p}_{M}$ (or, indifferently, ${}^{B}\mathbf{p}_{M}$). In particular, controlling this quantity imply controlling the observation of the object. However, the actual quantities that are considered for the N-MPC motion are its state variable, and in particular its position and orientation in $\mathcal{F}_{W}$, ${}^{W}\mathbf{p}_{B}$ and ${}^{W}\mathbf{R}_{B}$. As a consequence, in order to be controllable, ${}^{B}\mathbf{p}_{M}$ needs to be expressed as a function of ${}^{W}\mathbf{p}_{B}$ and ${}^{W}\mathbf{R}_{B}$. It naturally derives from this that the controller needs to be provided with the value of ${}^{W}\mathbf{p}_{M}$.

As a complement to the discussion in Section 3.7 on the reference frame for the tracking of the markers of interest, we remark that performing the tracking in $\mathcal{F}_{S}$

rather than $\mathcal{F}_W$ does not affect the reference frame in which the parameters are provided to the N-MPC, which is required to be $\mathcal{F}_W$. Providing the N-MPC directly with ${}^S\mathbf{p}_M$ does not allow to intricate it with the N-MPC state as defined in Chapter 3, hence does not allow to produce motion accounting for the perception.

Then, using some roto-translations, the relative position ${}^S\mathbf{p}_M$ is f as

$$
\begin{aligned}
{}^S\mathbf{p}_M &= {}^S\mathbf{R}_B\,{}^B\mathbf{p}_M + {}^S\mathbf{p}_B && \text{(5.1a)}\\
&= {}^S\mathbf{R}_B({}^B\mathbf{R}_W\,{}^W\mathbf{p}_M + {}^B\mathbf{p}_W) + {}^S\mathbf{p}_B && \text{(5.1b)}\\
&= {}^S\mathbf{R}_B({}^W\mathbf{R}_B^\top({}^W\mathbf{p}_M - {}^W\mathbf{p}_B)) + {}^S\mathbf{p}_B && \text{(5.1c)}\\
&= {}^B\mathbf{R}_S^\top({}^W\mathbf{R}_B^\top({}^W\mathbf{p}_M - {}^W\mathbf{p}_B) - {}^B\mathbf{p}_S). && \text{(5.1d)}
\end{aligned}
$$

In Equation (5.1d), ${}^W\mathbf{R}_B$ and ${}^W\mathbf{p}_B$ are part of the N-MPC state vector, while ${}^W\mathbf{p}_M$ is provided as external parameters.

Leveraging this formula for ${}^S\mathbf{p}_M$ in the N-MPC allows to assess the relative GTMR-object pose, and propagate it through the receding horizon. Throughout the manuscript, we will exploit this to constrain or improve the detection.

5.3 Perception Objectives

The first aspect to be considered is to provide incentives to the N-MPC to fulfill the perceptive task, which is indeed tackled through the introduction of a proper formalization of the objectives $C^{\text{perception}}$.

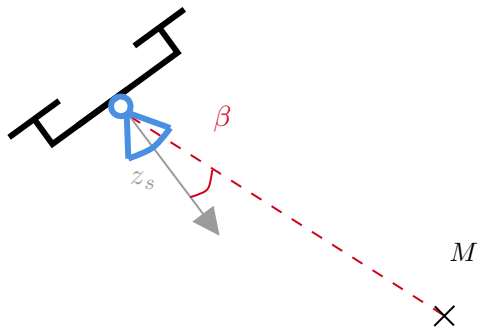
Let us consider, in a first step, the visibility coverage of a single object, denoted M, using a single sensor S. In order to maintain visibility over M,it is mandatory for the controller to guide the sensor toward this object.

The object position ${}^B\mathbf{p}_M$ being known through the horizon, the perception objective $C^{\text{perception}}$ has to be framed such that the controller tends to orient the sensor bearing toward the object. The bearing of the sensor is governed by the orientation of its principal axis $\mathbf{z}_S$. Consequently, the optimization of the visibility over the object is achieved through the minimization of the angular distance between the principal axis $\mathbf{z}_S$ and the bearing vector ${}^S\mathbf{p}_M$.

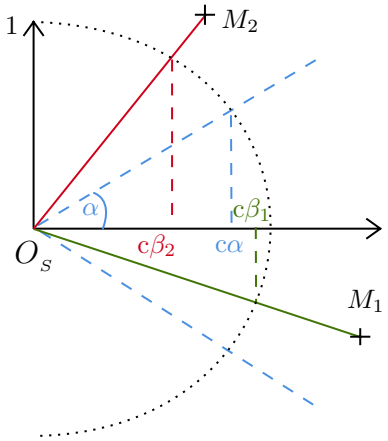
We take inspiration from [Penin, 2018] to geometrically formulate the visibility over a given feature. We denote $\beta \in [0,\ \pi]$ the non-oriented minimal angle between $\mathbf{z}_S$ and ${}^S\mathbf{p}_M$, see Figure 5.1.

The choice of the definition domain of β to be $[0,\ \pi]$ is achieved without any loss of generality, since it describes the angular distance between the two vectors. Furthermore, we assume that the object M is located in the half-space ${}^Sz_M > 0$, i.e. in front of the sensor, This assumption is motivated by the fact that the object is constrained to belong to the sensor FoV (see Section 5.4.1), hence does not imply any loss of generality. This allows to define $\beta \in \left[0,\ \frac{\pi}{2}\right[$.

Finally, considering that the cosine function is decreasing and positive on $\left[0,\ \frac{\pi}{2}\right[$, the minimization of the angular distance between the principal axis $\mathbf{z}_S$ and the bearing vector ${}^S\mathbf{p}_M$ can be equivalently rephrased as the minimization of $1 - \cos(\beta)$.



(a) The β angle between the sensor principal axis and the observed feature M.



(b) Visibility assessment using cβ. M_1 is visible (within the blue cone), while M_2 is not.

Figure 5.1: Depiction of the visibility criterion cβ.

Henceforth, we define

$$\mathrm{c}\beta = \cos(\beta). \tag{5.2}$$

This recast is meaningful as it avoids the use of a nonlinear trigonometric function. Indeed, cβ is computed through the projection of the bearing vector onto $\mathbf{z}_S$:

$$\mathrm{c}\beta = \frac{{}^S\mathbf{p}_M}{\|{}^S\mathbf{p}_M\|} \cdot \mathbf{z}_S, \tag{5.3}$$

Combining Equations (5.1) and (5.3) provides the desired closed-form formula for cβ, to include in $C^{\text{perception}}$.

Remark. The natural reference value for cβ is 1. Yet, this reference could also be defined as accounting for the current velocity and acceleration of the feature in the image plane. For instance, maintaining $\beta = 0$ while the feature moves toward the right leaves the left half of the FoV unexploited. A more complex reference value would improve the coverage and reactivity to sudden feature motion. Keeping a constant reference value equal to 1 has proven largely sufficient in our experiments, thus this strategy has not been implemented. However, it might prove useful when handling agile maneuvers or fast feature motions.

In addition, it might beneficial to introduce in $C^{\text{perception}}$ the minimization of the velocity in the sensor frame, as proposed in [Falanga, 2018]. The first benefit would be to anticipate the object motion in $\mathcal{F}_S$ in a more accurate way, penalizing large motions and enforcing to maintain the GTMR-object relative position constant. Second, in some particular cases, the velocity of the object in $\mathcal{F}_S$ might affect the quality of the detection, e.g. as a consequence of motion blur when using monocular cameras. To fulfill the first of these two goals, according to the object model introduced in Chapter 3 which consists of a punctual feature, reducing the tangential velocity is sufficient. This can be easily recast as the minimization of $\dot{\mathrm{c}\beta}$. Nonetheless, this approach does not capture entirely the second goal, since the detection of the feature might also be disturbed by fast radial motions.It thus turns out to be more fruitful to minimize directly the velocity vector ${}^S\dot{\mathbf{p}}_M$, which also possesses a simpler equation.

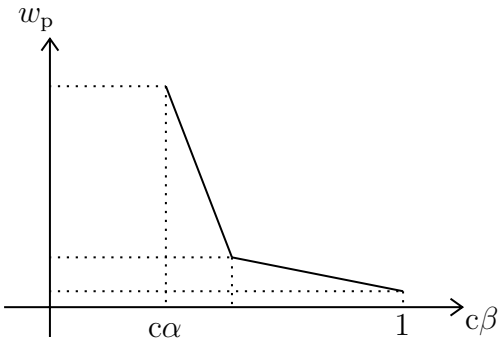


Figure 5.2: Piecewise-linear function used to adapt the weight on the perception objective.

The equation of ${}^{S}\dot{\mathbf{p}}_{M}$ is obtained through the differentiation of Equation (5.1d), utilizing that ${}^{B}\mathbf{T}_{S}$ is constant:

$$
\begin{aligned}
{}^{S}\dot{\mathbf{p}}_{M} &= {}^{B}\mathbf{R}_{S}^{\top}\left[{}^{B}\dot{\mathbf{R}}_{W}({}^{W}\mathbf{p}_{M} - {}^{W}\mathbf{p}_{B}) + {}^{B}\mathbf{R}_{W}^{\top}({}^{W}\dot{\mathbf{p}}_{M} - {}^{W}\dot{\mathbf{p}}_{B})\right] && \text{(5.4a)}\\
&= {}^{B}\mathbf{R}_{S}^{\top}{}^{W}\mathbf{R}_{B}^{\top}\left([{}^{B}\boldsymbol{\omega}_{W}]_{\times}({}^{W}\mathbf{p}_{M} - {}^{W}\mathbf{p}_{B}) + {}^{W}\mathbf{v}_{M} - {}^{W}\mathbf{v}_{B}\right) && \text{(5.4b)}\\
&= {}^{B}\mathbf{R}_{S}^{\top}{}^{W}\mathbf{R}_{B}^{\top}\left([{}^{W}\mathbf{R}_{B}^{\top}{}^{W}\boldsymbol{\omega}_{B}]_{\times}({}^{W}\mathbf{p}_{M} - {}^{W}\mathbf{p}_{B}) + {}^{W}\mathbf{v}_{M} - {}^{W}\mathbf{v}_{B}\right). && \text{(5.4c)}
\end{aligned}
$$

However, such penalization of the velocity didn't provide significant improvement of the scheme over our experiments. In fact, we conjecture that this penalization might only turn out useful for agile maneuvering, where motion blur becomes a prominent issue. Hence, the computation of ${}^{S}\dot{\mathbf{p}}_{M}$ is reported in this section, but such a term is not included in the proposed framework.

To conclude, the perception objectives $C^{\,\text{perception}}$ are written

$$
C^{\,\text{perception}} = w_\mathrm{p}(1 - \mathrm{c}\beta), \tag{5.5}
$$

where w_p is the tunable controller weight. This cost does not need to be quadratic since $(1 - \cos)$ is C^∞ and convex on $\left[0,\ \frac{\pi}{2}\right[$.

Remark. In order to enforce the visibility objective when the feature gets closer to the FoV boundaries, and relax it elsewise, the weight on w_p could be adapted w.r.t. the value of $\mathrm{c}\beta$. It can be done in a piecewise-linear fashion, as shown in Figure 5.2. This obviously requires more tuning (4 parameters instead of 1), but is computationally efficient and can improve the observed results. From a broader perspective, automatic adaptive weights in N-MPC start to appear in the literature, which allows to comply with various scenarios (e.g., precise hovering or agile maneuvering). In [Kostadinov, 2020], the authors show large tracking improvements using such an adaptive weighting policy.

5.4 FoV Constraints

The aforementioned objectives are designed to orient the sensor bearing toward the object of interest. But this is only possible if the object position is known by the

system, hence that measurements are retrieved. The motion of the AR, in particular when dealing with ample maneuvers, can easily cause the loss of visibility over the feature. Even if the Kalman filter would still yield an estimation of the object position, it does not capture the unpredictable inputs. To ensure the monitoring of the feature at each instant, we impose to maintain the visibility as a hard constraint in the system [Penin, 2017]. In addition, Section 5.4.4 proposes a method to relax this constraint with a slack variable.

To this end, we propose to decouple the range and bearing aspect of the visibility.

5.4.1 Range Constraint

The most straightforward constraint for the visibility is the observed feature needs to be in front of the sensor. Thus, the range has to be strictly positive. Furthermore, for most of sensors, the sensing is possible in a given, pre-defined, range. One can think of RGBD or lidar sensors, that have a minimum sensing distance due to the hardware-related minimum travel times of the laser beams. Similarly, these sensors have a maximum observation distance, after which the energetic dissipation of beams makes them undetectable by the device. Stereo cameras are also subject to similar issues, due to the limited spacing between cameras that cannot exploit parallax passed a certain distance. Even when considering standard monocular cameras, there are intrinsic limitations of the detection: the object must cover a sufficient amount of pixels in the image plane. It must also be far enough to be entirely seen, and further than the focal length to avoid being out of focus.

Such a constraint can be easily expressed as an inequality on the norm of the bearing vector ${}^{s}\mathbf{p}_{M}$. However, it can be noted that this constraint does not encapsulate the constraint that the object is in front of the sensor. An additional constraint on the positivity of ${}^{s}z_{M}$ can be expressed. Yet, we remark that the depth limitation is mostly driven by the distance along $\mathbf{z}_{S}$. Accordingly, the constraint can be recast as

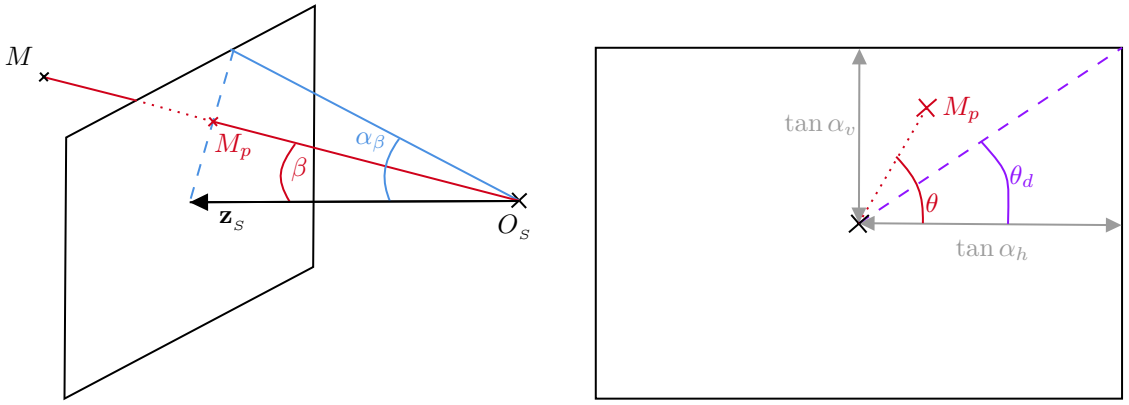
$$\underline{d} \leq {}^{s}z_{M} \leq \overline{d}. \tag{5.6}$$

where $\underline{d} > 0$ and $\overline{d}$ are respectively the lower and upper bounds of the distance along the sensor principal axis.

5.4.2 Conic FoV

Let us first approximate the sensor FoV as a cone. We leverage the formalism from Section 5.3 to express the visibility as the angular distance between the bearing vector and the principal axis. Equivalently, this is recast as a constraint on $\mathrm{c}\beta$. While the upper bound is constant and equal to 1, the lower bound $\underline{\mathrm{c}\beta}$ is defined, in a conservative way, as the cosine of the minimum of the halved horizontal and vertical angular FoV:

$$\underline{\mathrm{c}\beta} = \max(\cos\alpha_h, \cos\alpha_v). \tag{5.7}$$



(a) 3D view of the angle α_β.

(b) Depiction of the corresponding virtual plane ${}^{S}z = 1$.

Figure 5.3: Visual depiction of the state-dependent α_β angle.

5.4.3 Pyramidal FoV

However, it remains interesting to exploit the full visibility capabilities of the system, by modeling the full pyramidal FoV of the onboard sensor. To this end, a solution is to express a state-dependent limit for β. To maintain similar notation w.r.t. the FoV horizontal and vertical limits, we denote this lower bound α_β, where the subscript β recall the aforementioned state dependency. The lower bound of $\mathrm{c}\beta$ is therefore the cosine of this angle, denoted $\mathrm{c}\alpha_\beta$. Figure 5.3 pictures the angle α_β in $\mathcal{F}_S$, and the quantities involved in its computation. From Figure 5.3a, we assess that α_β depends on the position of M_p, the projected feature on the plane (${}^{S}z = 1$). However, due to the different horizontal and vertical FoV, the formula depends if M_p is "above" or "below" the FoV diagonal, therefore it depends of the angle θ, pictured in Figure 5.3b. We remark that the symmetry of the FoV plane can be exploited to restrict the problem to the "right" part, i.e. angles $\theta \in \left]-\frac{\pi}{2}, \frac{\pi}{2}\right]$. Because of the symmetry of the cosine, the problem can be reduced further to the upper-right quarter of the rectangle, i.e. $\theta \in \left[0, \frac{\pi}{2}\right]$.

The θ angle is defined as

$$\theta = \begin{cases} \tan\left(\frac{\left|{}^{S}y_M\right|}{\left|{}^{S}x_M\right|}\right) & , \quad \text{if } {}^{S}x_M \neq 0 \\ \frac{\pi}{2} & , \quad \text{otherwise} \end{cases}. \tag{5.8}$$

Then, the angle θ_d is defined by the FoV shape as

$$\theta_d = \operatorname{atan}\left(\frac{\tan \alpha_h}{\tan \alpha_v}\right) \in \left]0, \frac{\pi}{2}\right[. \tag{5.9}$$

And the equation of α_β is thus given by

$$\alpha_\beta = \begin{cases} \operatorname{atan}(\tan \alpha_h / \cos \theta), & \text{if } \theta < \theta_d \\ \operatorname{atan}(\tan \alpha_v / \sin \theta), & \text{otherwise} \end{cases}. \tag{5.10}$$

Therefore, we have the constraint

$$\mathrm{c}\alpha_\beta \leq \mathrm{c}\beta \leq 1. \tag{5.11}$$

Another typical approach is to decouple the horizontal and vertical axis of the sensor [Allibert, 2010]. Then a pair of decoupled constraints can be written concerning the position of the object projection onto the normalized image plane $I\colon {}^{S}z = 1$. The projected point is given by

$$ {}^{I}\mathbf{p}_{M} = \begin{bmatrix} {}^{I}x_{M} \\ {}^{I}y_{M} \\ 1 \end{bmatrix} = \frac{1}{{}^{S}z_{M}} \, {}^{S}\mathbf{p}_{M}, \tag{5.12} $$

which is always well defined, as ${}^{S}\mathbf{p}_{M} > 0$. It then leads to the two following constraints:

$$ 0 \leq \frac{|{}^{S}x_{M}|}{{}^{S}z_{M}} \leq \tan\alpha_{h}, \tag{5.13a} $$

$$ 0 \leq \frac{|{}^{S}y_{M}|}{{}^{S}z_{M}} \leq \tan\alpha_{v}. \tag{5.13b} $$

Using this pair of constraints is beneficial because it is computationally lighter than resorting to $\mathrm{c}\alpha_{\beta}$. This solution is thus chosen hereafter. The uni-dimensional $\mathrm{c}\alpha_{\beta}$ representation is however very useful to graphically picture the constraint, and is therefore used in the various plots presented in Section 5.7.

5.4.4 Relaxation of Constraint

Including such hard constraints on the observability allows to ensure that the feature is always observable, in the limits of the physical capabilities of the GTMR. In fact, when dealing with agile maneuvers and/or fast feature motions, it might be that no solution satisfies all the constraints. In such conditions, it is desirable to "soften" the visibility constraints, such that the N-MPC is able to provide a solution that still fulfills the actuation constraints, which are of prime importance. The visibility constraints on the other hand can be unfulfilled for a short period of time without disrupting the system stability, and leveraging the predictions of the feature filtering. A relaxation of the constraints can be achieved by the inclusion of a *slack variable* [Zeilinger, 2010; Penin, 2018], a virtual additional system input ρ which is included in the visibility constraint as

$$ \mathrm{c}\alpha_{\beta} - \rho \leq \mathrm{c}\beta \leq 1, \tag{5.14} $$

for a conic FoV and

$$ 0 \leq \frac{|{}^{S}x_{M}|}{{}^{S}z_{M}} \leq \tan\alpha_{h} + \rho, \tag{5.15a} $$

$$ 0 \leq \frac{|{}^{S}y_{M}|}{{}^{S}z_{M}} \leq \tan\alpha_{v} + \rho, \tag{5.15b} $$

for a pyramidal FoV. Concurrently, a penalization term is included in the N-MPC cost function, as

$$ w_{\rho} \left|\rho\right|^{2}, \tag{5.16} $$

typically with a very large weight w_ρ. This allows the system to relax the constraint only when no other viable solution can be found.

Additionally, and since the N-MPC inputs need to be constrained, an extra constraint is added to prevent the system from reaching undefined configurations, namely when ${}^{S}z_{M} \leq 0$:

$$0 \leq \rho \leq \overline{\rho}, \tag{5.17}$$

where the upper-bound $\overline{\rho}$ needs to be arbitrarily chosen. If the system is still unable to provide a solution with this relaxation, it is desirable to fallback to a backup control policy or give up the task to ensure safety.

However, the use of slack variables does not provide any guarantee on how long the visibility can be lost. Large weights are thus required to limit at best this effect, possibly causing instabilities. Consequently, the additional tuning imposed by such policy is important.

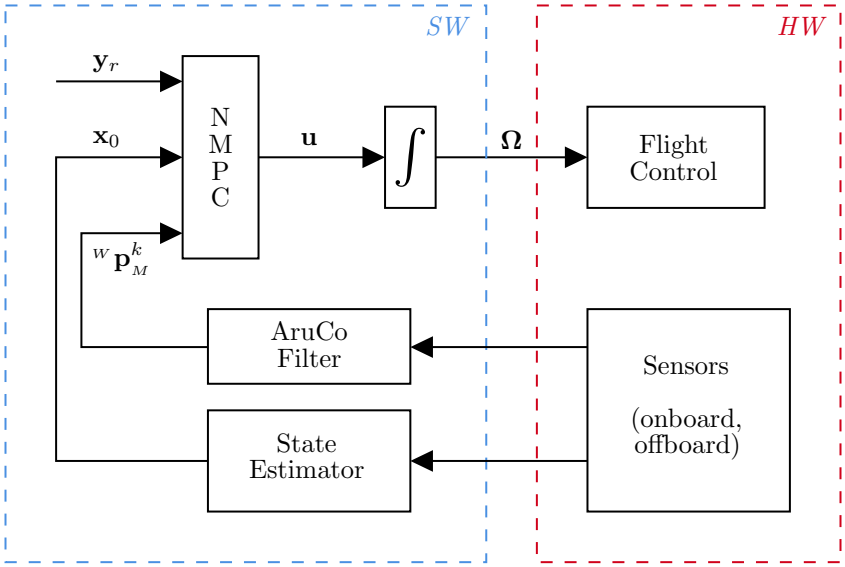


Figure 5.4: Block diagram of the proposed framework.

5.5 NonLinear Programming Formulation

Equipped with the newly defined objectives and constraints, the generic NLP formalism introduced in Section 4.4 can be instantiated for our specific problem.

The formula of $C^{\text{perception}}$ is given by Equation (5.5). Then, the perception bounds mapping $\boldsymbol{\psi}$ is expressed as the combination of Inequations (5.6) and (5.13):

$$\boldsymbol{\psi}(\mathbf{x}, \mathbf{p}) = \begin{bmatrix} \frac{|{}^{S}x_{M}|}{{}^{S}z_{M}} \\ \frac{|{}^{S}y_{M}|}{{}^{S}z_{M}} \\ {}^{S}z_{M} \end{bmatrix}. \tag{5.18}$$

Therefore, the NLP from Equation (4.11) is reformulated as

$$\min_{\substack{\mathbf{x}_0 \dots \mathbf{x}_N \\ \mathbf{u}_0 \dots \mathbf{u}_{N-1}}} \quad \sum_{k=0}^{N} \|\mathbf{y}_k - \mathbf{y}_{r,k}\|^2_{\mathbf{W}_\mathrm{m}} + \sum_{k=0}^{N} w_\mathrm{p}(1 - \mathrm{c}\beta) + \sum_{k=0}^{N-1} \|\mathbf{u}_k\|^2_{\mathbf{W}_\mathrm{i}} \tag{5.19a}$$

$$\begin{aligned}
s.t. \quad & \mathbf{x}_0 = \mathbf{x}(t) && && (5.19b)\\
& \mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k), && k\in\{0,\, N-1\} && (5.19c)\\
& \mathbf{y}_k = \mathbf{h}(\mathbf{x}_k), && k\{0,\, N\} && (5.19d)\\
& \mathrm{c}\beta_{i,k} = \mathbf{g}(\mathbf{x}_k, {}^{W}\mathbf{p}^{k}_{M}), && k\{0,\, N\} && (5.19e)\\
& \underline{\boldsymbol{\gamma}} \leq \boldsymbol{\gamma}_k \leq \overline{\boldsymbol{\gamma}}, && k\{0,\, N\} && (5.19f)\\
& \underline{\dot{\boldsymbol{\gamma}}}(\boldsymbol{\gamma}_k) \leq \mathbf{u}_k \leq \overline{\dot{\boldsymbol{\gamma}}}(\boldsymbol{\gamma}_k), && k\in\{0,\, N-1\} && (5.19g)\\
& |{}^{S}x_{M_k}/{}^{S}z_{M_k}| \leq \tan\alpha_h, && k\{0,\, N\} && (5.19h)\\
& |{}^{S}y_{M_k}/{}^{S}z_{M_k}| \leq \tan\alpha_v, && k\{0,\, N\} && (5.19i)\\
& \underline{d} \leq {}^{S}z_{M_k} \leq \overline{d}, && k\{0,\, N\} && (5.19j)\\
& \underline{\boldsymbol{\mu}_k} \;\leq\; \boldsymbol{\mu}(\mathbf{x}_k, \mathbf{p}_k) \;\leq\; \overline{\boldsymbol{\mu}_k}, && k\{0,\, N\} && (5.19k)
\end{aligned}$$

where $\mathbf{g}$ synthetically denotes the computation of $\mathrm{c}\beta$ from Equation (5.3), and the parameter ${}^{W}\mathbf{p}^{k}_{M}$ is the position of the feature in $\mathcal{F}_W$, predicted over the horizon using the KF from Section 3.7, for the *k-th* shooting point. Additional task-specific parameters $\mathbf{p}_k$ can be added to instantiate the motion constraint mapping $\boldsymbol{\mu}$, in Inequation (5.19k).

Following this paradigm, the N-MPC proposed in this chapter is schematized in the block diagram in Figure 5.4, reusing the formalism from Figure 4.3.

5.6 Extension to Multiple Features and Sensors

The quantities $\mathrm{c}\beta$ and ${}^{s}\dot{\mathbf{p}}_{M}$ can be expressed for an arbitrary amount of objects and sensors. Using the paradigm introduced in this chapter, it is straightforward to track several features with a single sensor. Using several of those, however, is slightly more cumbersome. A simplistic approach is to assign a specific sensor to the tracking of a given feature. This strategy is efficient and easy to implement, and the ensuing notations are presented in this section, The experiments and simulation in Section 5.7 make use of this strategy. The intuitive motivation for this is the fact that onboard sensors are observing distinct and disjointed domains. In particular, it is common to bear a down-facing sensor and one (or several) side-facing one . The specific features to observe are often assigned to a specific domain, and the sensors are placed on the robot depending on the task to tackle. It however does come with the burden of a prior sensor/feature pairing. This is an important drawback which need to be tackled for field deployment. The main theoretical limitation of this approach is that it prevents the GTMR to exploit all of its sensors to perform the tracking

Remark. A solution for the definition of a shared observability constraint with all sensors over is introduced as a remark in Section 7.4.1. It makes use of concepts introduced in Chapter 6, and its presentation is more relevant there. However, the

writing of as shared perception objective $C^{\text{perception}}$ remains not trivial, hence a prior pairing would still be required.

An online pairing is feasible, e.g. by associating each feature to the sensor which yields the smallest β angle. This is however complex to introduce in the N-MPC cost function without introducing discontinuities.A pairing external to the N-MPC, before each optimization cycle, would not be satisfactory since the pairing would be considered fixed over the receding horizon, preventing the controller to exploit this DoF. Answering these considerations is left for future works.

In the following, the tracking of a given object is assigned to a specific sensor. We denote n_{p} the number of perception objectives (i.e. the number of sensor-feature pairs). Using the subscripts $\bullet_j, \quad j \in \{1, n_{\text{p}}\}$ to denote the quantities associated with each pair, the perception objectives $C^{\text{perception}}$ is written

$$C^{\text{perception}} = \sum_{j=1}^{n_{\text{p}}} (1 - \text{c}\beta_j). \tag{5.20}$$

Then, the perception bounds mapping $\boldsymbol{\psi}$ are expressed as the combination of Inequations (5.6) and (5.13):

$$\boldsymbol{\psi}_i(\mathbf{x}, \mathbf{p}) = \begin{bmatrix} \frac{|{}^S x_{M_j}|}{{}^S z_{M_j}} \\ \frac{|{}^S y_{M_j}|}{{}^S z_{M_j}} \\ {}^S z_{M_j} \end{bmatrix}, \; j \in \{1, n_{\text{p}}\}, \tag{5.21a}$$

$$\boldsymbol{\psi}(\mathbf{x}, \mathbf{p}) = \begin{bmatrix} \boldsymbol{\psi}_1(\mathbf{x}, \mathbf{p}) \\ \vdots \\ g\boldsymbol{\psi}_{n_{\text{p}}}(\mathbf{x}, \mathbf{p}) \end{bmatrix}. \tag{5.21b}$$

Correspondingly lower and upper bounds $\underline{\boldsymbol{\psi}}$ and $\overline{\boldsymbol{\psi}}$, expressed in Inequations (5.6) and (5.13), stacked n_{p} times for each pair of sensor and object, are:

$$\underline{\boldsymbol{\psi}_j} = \begin{bmatrix} 0 \\ 0 \\ \underline{d}_j \end{bmatrix}, \quad \overline{\boldsymbol{\psi}_j} = \begin{bmatrix} \tan \alpha_{h,j} \\ \tan \alpha_{v,j} \\ \overline{d}_j \end{bmatrix}, \quad j \in \{1, \; n_{\text{p}}\}, \tag{5.22a}$$

$$\underline{\boldsymbol{\psi}} = \begin{bmatrix} \underline{\boldsymbol{\psi}_1} \\ \vdots \\ \underline{\boldsymbol{\psi}_{n_{\text{p}}}} \end{bmatrix}, \quad \overline{\boldsymbol{\psi}} = \begin{bmatrix} \overline{\boldsymbol{\psi}_1} \\ \vdots \\ \overline{\boldsymbol{\psi}_{n_{\text{p}}}} \end{bmatrix}, \tag{5.22b}$$

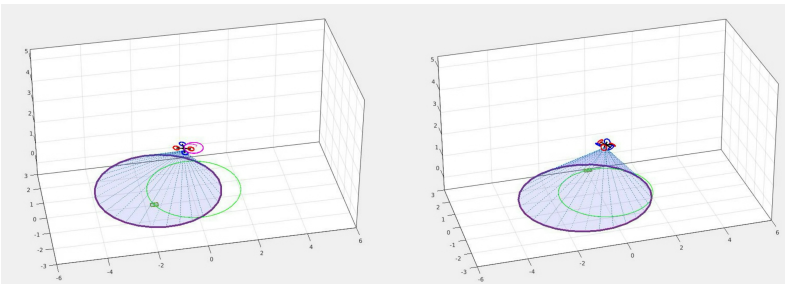
where $\alpha_{v,j}$ an $\alpha_{h,j}$ are the halved horizontal and vertical FoV of the sensor paired with the j-*th* feature, and $\underline{d}_j$, $\overline{d}_j$ are its range limits.

5.7 Experimental and Simulation results

This section presents the results achieved in real experiments and simulations. The GTMR employed are a collinear quadrotor (for real experiments) and a tilted-propeller

Variable	Simulation weights		Experiment weights
	Quadrotor	Hexarotor	
$\mathbf{p}$	30	10	80
$\mathbf{q}$	5	3	80
$\mathbf{v}$	1	5	10
$\boldsymbol{\omega}$	1	10	10
$\mathbf{a}$	10^{-4}	10^{-4}	10^{-4}
$\dot{\boldsymbol{\omega}}$	10^{-4}	10^{-4}	10^{-4}
$\mathbf{u}$	0	0	0
$\mathrm{c}\beta$	100	50	50

Table 5.1: Table of N-MPC weights for simulations and experiments in Section 5.7.



(a) Clickable image to the video on PeerTube.

(b) QR code to the video.

Video 5.1: Comparison between two types of ARs for the observation a circular motion.

hexarotor (for simulations). Both are equipped with two cameras, down-facing and front-facing, see Figure A.2a. Additionally, the experiments make use of two different cameras, to demonstrate that the framework can handle various types of sensors simultaneously. We refer to Appendix A for details on the experimental setup, and to Appendix B for a discussion on the computation time.

The controller weights used in the presented simulations and experiments are reported in Table 5.1. In this section, as well as in the experimental sections of Chapter 6 and Chapter 8, the reported weights are tuned manually. However, it can be noted that those are roughly consistent among the various chapters, despite the disparities in scenarios. Moreover, weights are maintained constant across all the experiments for a single chapter (e.g., for static and mobile cases), demonstrating a low sensitivity of the controller w.r.t. the weight tuning. No experiments with tilted-propeller hexarotors have been conducted in this thesis, as discussed in Section 9.3. Consequently, such weights are not reported in the designated tables.

5.7.1 Near Hovering while Observing a Circular Motion

Firstly, we report a numerical simulation performed in Matlab and Simulink. It can be seen is Video 5.1. The GTMR is requested to hover, while keeping visibility over the moving marker. The latter performs a circular motion, whose radius is chosen to be outside of the FoV projection at ground-level, for the requested altitude.

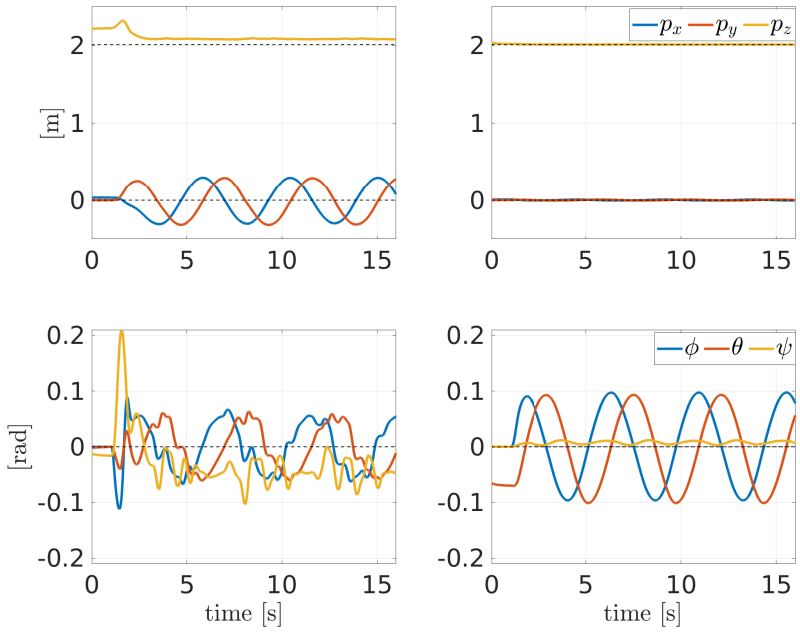


Figure 5.5: Position and attitude tracking when observing a feature moving in circle, with a quadrotor (left) and an hexarotor (right). The dashed black line is the requested hovering state.

Figure 5.5 presents the resulting position and attitude tracking of an underactuated quadrotor and a tilted-propeller hexarotor. The latter is able to stay much closer to the hovering state by slightly modulating its attitude, while the former has to make a circular motion in order to maintain visibility. With this simulation, we show that the controller is able to take advantage of the larger actuation of fully-actuated platforms in tasks where underactuation is detrimental w.r.t. other objectives, such as perception.

5.7.2 Hover-to-hover Under Visibility Constraints

This first set of experiments aims at demonstrating the capability of the proposed framework to modulate a reference task in order to maintain visibility over a set of features. In particular, the two cameras have to maintain visibility over a marker on a wall and on the ground, respectively. The two markers are fixed in $\mathcal{F}_W$, and the UAV is given a position reference trajectory that is not feasible under the visibility constraints.

Video 5.2 shows first an experiment in which the visibility constraints and objectives are disabled. Then the experiment presented in Figures 5.6 and 5.7 is then showed in the video, where the only difference in the setup is the enabling of the perception-awareness.

Results of the second experiment are presented in Figure 5.6, which depicts the (x, y) coordinates of the UAV, the feature positions and the reference trajectory. The color dots indicate the GTMR z coordinate, whose reference is constant and set to $z_r = 1$. As the (x, y) distance between the reference and the feature increases, the

(a) Clickable image to the video on PeerTube.

(b) QR code to the video.

Video 5.2: Experiments with a quadrotor in two conditions: with and without visibility constraints.

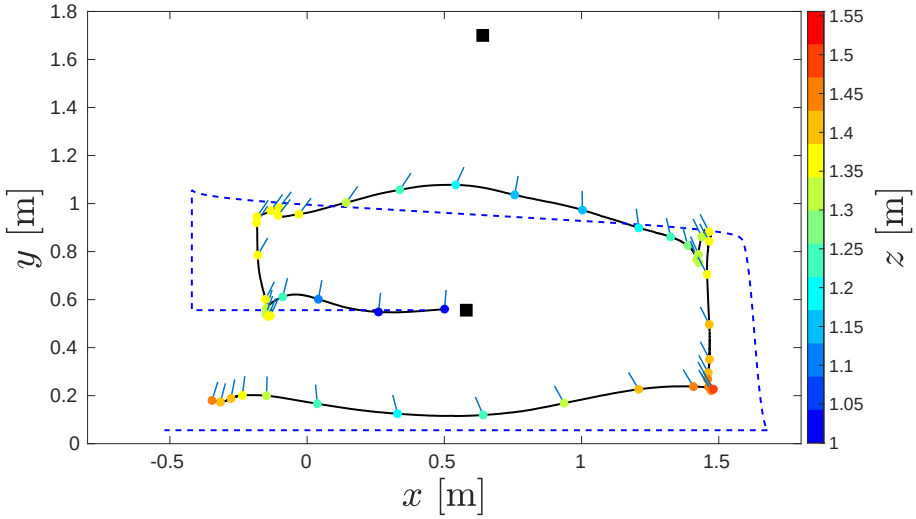


Figure 5.6: (x, y) position of the GTMR tracking a trajectory while enforcing visibility constraints. The dashed line is the reference trajectory (at constant altitude) while the solid black line is the GTMR path. The color dots represent the GTMR position along that path every 0.3 seconds, while their color represents the corresponding altitude z, the blue segments are the front camera heading. The two black squares are the target position.

tracking error increases in order to accommodate for visibility. In particular, when the AR moves along the x direction, the downward markers would get out of the FoV if the altitude was not modulated. Figure 5.7 shows the value of $c\beta$ for the two cameras with their respective lower bounds, i.e. the cosine angular FoV $c\alpha_\beta$. The resulting motion is a trade-off between the two objectives C^{motion} and $C^{\text{perception}}$, which satisfies the various constraints.

5.7.3 Mobile Feature Tracking

This experiment, reported in Video 5.3, exploits the same setup, but the feature on the ground is mobile, and the reference task given to the GTMR is to stay on top of it at a constant altitude ($z = 1$ m). The controller heavily exploits the rotation around

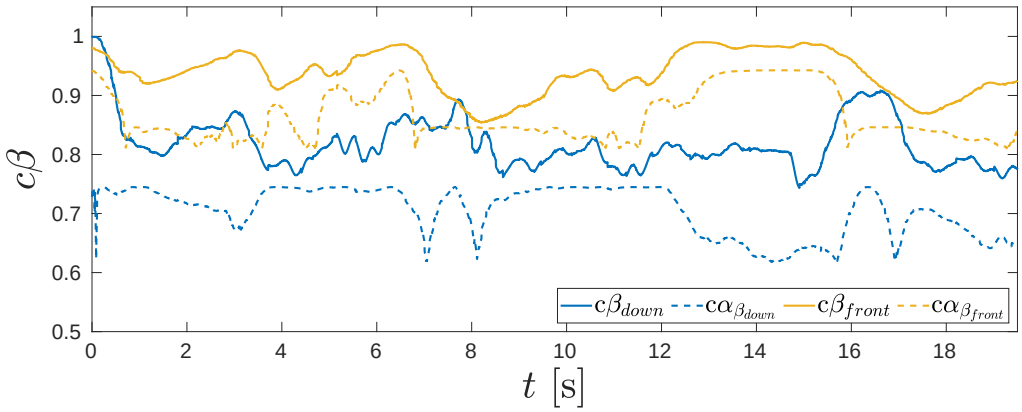
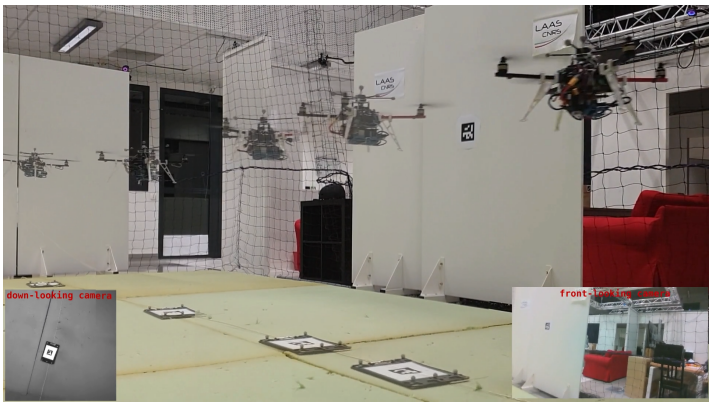


Figure 5.7: The measured cβ (solid) and corresponding lower bound cα_β (dashed) for the two pairs camera/marker (front-facing in yellow and down-facing in blue).



(a) Clickable image to the video on PeerTube.

(b) QR code to the video.

Video 5.3: Experiment with a quadrotor: tracking of a moving marker.

$\mathbf{z}_B$ to maintain visibility on the wall feature while moving. This also illustrates that the controller is able to autonomously satisfy the perception constraints and objectives without the need of any additional user inputs. Results are reported in Figure 5.8, which shows that the visibility constraints are always satisfied. The maximum speed and acceleration allowed for the feature in order for the AR to fulfill the constraints are dependent on the sensor FoV and the requested altitude z. In the presented experiment, the average target speed is 0.5 m/s.

Additionally, Table 5.2 presents the mean and standard deviation of the reprojection error between the measured feature poses and the ground truth (obtained using motion capture), with and without the uncertainty propagation method proposed in Section 3.6. These data are aggregated over several experiments covering three minutes of flight in each case, and in similar conditions. We note the disparity between the metrics for the two markers, which is caused by the design of the experiment. The front feature is often seen from the side, which worsens the AruCo position estimate. For both features, the proposed method increases the reprojection precision reducing the average error by 30% to 40% and standard deviation by 15% to 60%. It demonstrates the importance of a proper measurement covariance estimation in such perception-constrained controllers. A reprojection error of the order of magnitude of 10 cm can lead to failure in assessing the perception constraints to fulfill, in particular

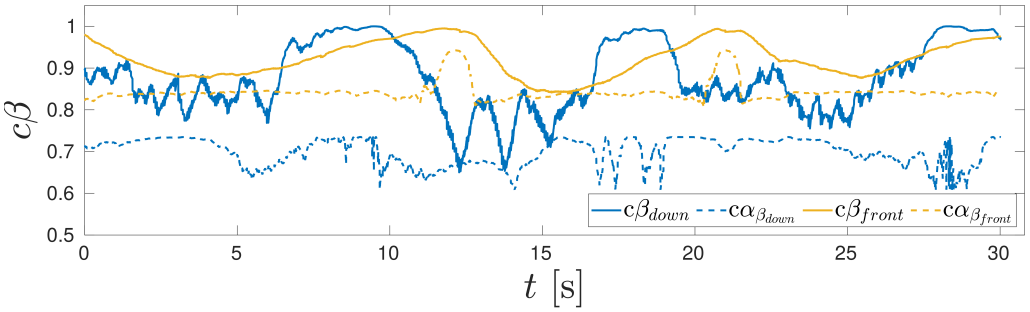


Figure 5.8: The measured cβ and its bound cα_β for the two pairs camera/marker (both front- and down-looking).

Uncertainty estimation	Down feature mean	Down feature standard deviation (std)	Front feature mean	Front feature std
without	0.084	0.158	0.349	0.091
with	0.058	0.065	0.215	0.077

Table 5.2: Reprojection error (mean and std, both in meters) with and without uncertainty estimation, for both the front and down features.

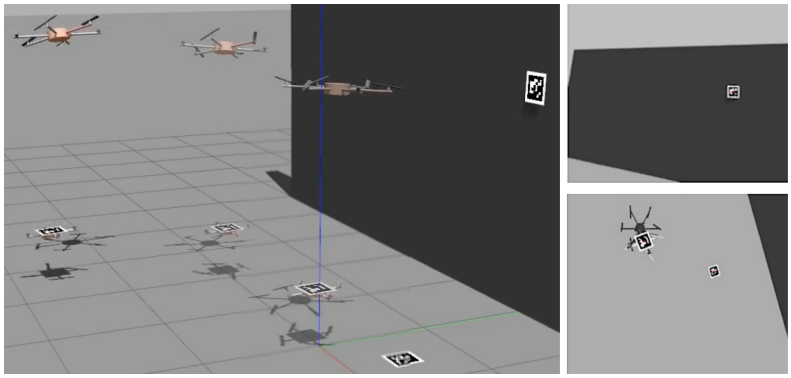
when dealing with agile motions.

Remark. The error values reported Table 5.2 are increased by the poor estimation of the camera/body transform ${}^{B}\mathbf{T}_{C}$, which was assessed manually. A proper extrinsic calibration should be conducted to reduce the reprojection error.

5.7.4 Simulation with a Tilted-Propeller Hexarotor

This Gazebo simulation uses a fully-actuated tilted-propeller hexarotor, as shown in Video 5.4. The scenario is similar to the experiment from Section 5.7.3, but is more challenging and aims at reaching the limits of the AR actuation. Visibility has to be maintained over a fixed marker on a wall with a front-looking camera, and over two markers with a down-looking camera; of which one is fixed while the second is mobile. The mobile feature is attached to a quadrotor, controlled with a geometric controller [Spica, 2013], which gains are de-tuned to achieve a slightly erratic motion.

This simulation aims at demonstrating the capability of the controller to exploit the full action span of the platform. In particular, the tilted-propeller hexarotor is able to hover with nonzero roll and pitch, as long as the motor velocities. Figure 5.9 shows that, e.g., in the phase between 22 and 40 seconds, the platform takes advantage of its full-actuation to hover while tilted, up to about 20°. The corresponding propeller thrusts are reported in parallel, to illustrate that the actuation of the GTMR reaches its limits during this phase. The system inputs, presented in Figure 5.9, also touch their respective lower and upper bounds along the motion. The actuation constraints are active and the platform is therefore exploited at the maximum of its capability by the N-MPC.



(a) Clickable image to the video on PeerTube.

(b) QR code to the video.

Video 5.4: Simulation with a fully-actuated hexarotor in a complex monitoring scenario.

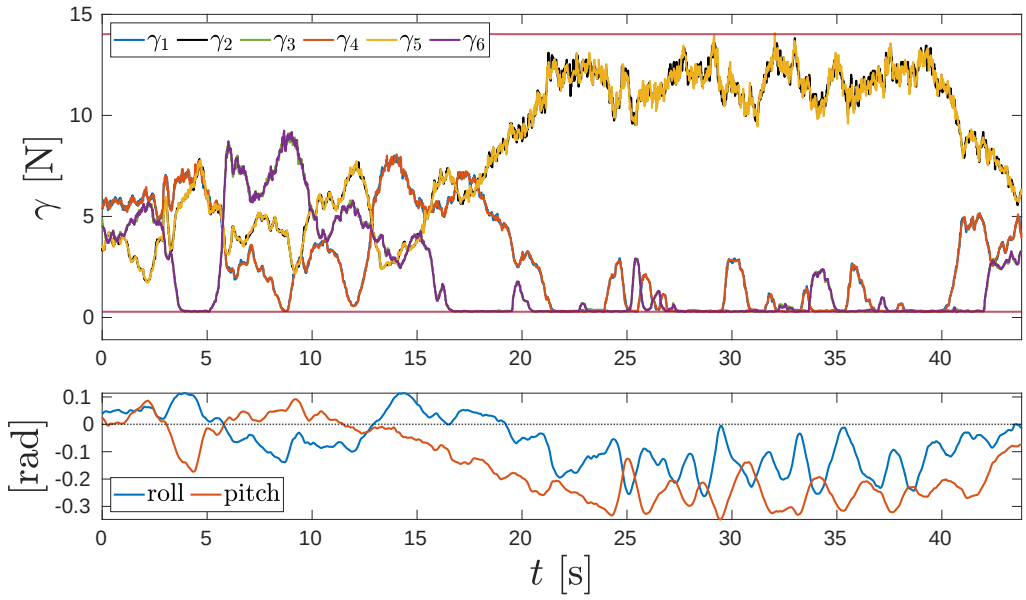


Figure 5.9: Top: Thrusts generated by the 6 propellers. Bottom: roll and pitch angles of the AR.

5.8 Application to Human-Robot Handover

5.8.1 Presentation

Following the initial work on the perception-constrained N-MPC introduced in this chapter, a collaborative work that exploits the capabilities of this framework has been conducted. It proposes a N-MPC for fully autonomous handover of an object between an AR and a human coworker. It includes the same formulation of the perception-related constraints and objectives as presented in this chapter, but also goes beyond by optimizing on the human ergonomics, in particular related to the minimization of the torque effort imposed on the coworker during the handover. A set of additional safety-related motion constraints are also enforced, exploiting the motion constraints $\boldsymbol{\mu}$. Finally, the motion objective C^{motion} is expressed relative to the human torso, rather than being specified in $\mathcal{F}_W$. Consequently, the initially planned trajectory does not need to be recomputed as the human moves.

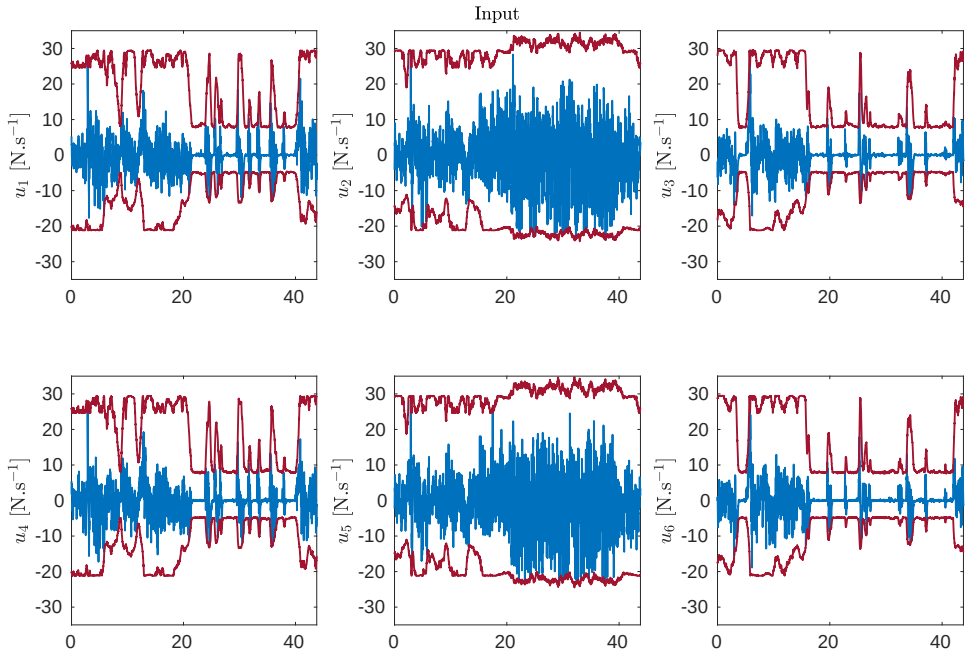
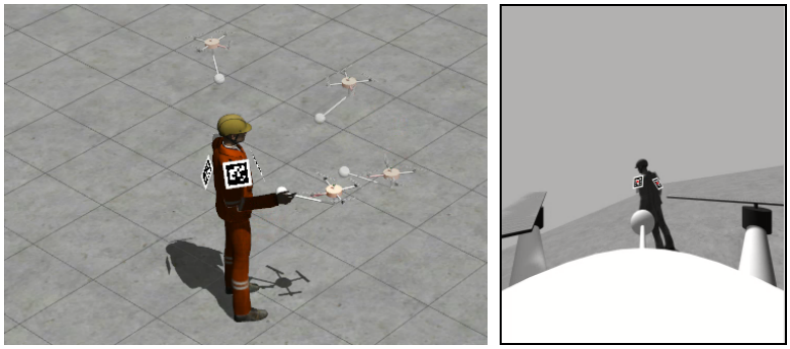


Figure 5.10: System inputs for the 6 propellers along the simulation.

The complete framework is not detailed hereafter, as the ergonomics and safety objectives are not the focus of this thesis, and were overseen by another PhD student. This section focuses only on the perception-related aspects.

Thus, the N-MPC is used in order to ensure that the human is monitored at each instant of the handover, and to account for her intrinsically unpredictable motion. Indeed, a potential loss of visibility implies the loss of awareness of the human position in the workspace, which would prevent the finalization of the handover task, and lead to potential hazards. These considerations naturally led to the use of a perception-constrained controller.

The GTMR used in the scope of this work is a collinear quadrotor, equipped with a front-facing monocular grayscale camera. To avoid the implementative burden of embedding a CNN-based human detector, the coworker is equipped with a set of four AruCo markers (on the torso, back, and shoulders). This allows her to be detected regardless of the angle from which she is observed. It can be pictured that such markers are printed onto her working suit. The detected markers are used as measurements of a Unscented Kalman Filter (UKF), which is tracking the position of the geometrical center of the four markers, which is located in the coworker torso. The position of this point is provided by the filter to the N-MPC, which enforces the visibility. The fact that visibility is not enforced onto a specific marker, but rather on the center of several markers, implies some implicit assumptions on the position of the markers, and the observation distance of the camera. In fact, if the markers at far from each other, or observed from too close, the detection of the marker might be disrupted. This assumption might be compromised as the AR performs the handover, hence is getting closer to the coworker hand, which might induce a loss of visibility over the marker located on her torso. This is preempted by extending the camera FoV vertically, either by tilting the camera upward, or placing it in “portrait” orientation.



(a) Clickable image to the video on PeerTube.

(b) QR code to the video.

Video 5.5: Simulation of a fully autonomous Human-Robot handover. On the left, successive positions of the simulated AR performing the handover. On the right, a frame of the robot's onboard camera.

In the proposed simulations, the GTMR is tasked to reach a point in front of the coworker, starting from a random location. Such phase is usually referred to as the *approach* phase [Strabala, 2013]. It consists of bringing the AR in a safe position from which the human can see it. From this point on, the *reach* phase is enabled , bringing the object in the handover position, in close range of the coworker. This is achieved through the ergonomics-related objectives that are enabled in this phase, whilst the motion-related ones C^{motion} are disabled. This drives the GTMR to perform the handover in the most natural and comfortable position for the coworker, rather than to a pre-computed position.

Remark. The *transfer* phase, involving the actual passing of the object, considering the underlying physical interactions, is not tackled in the scope of this work. Yet, the modeling that is employed in this work is an extension of the GTMR model from Chapter 3, which encompasses external wrench applied at an end-effector (i.e., the tool holding the object). We refer to Section D.2.2 for a suited definition of the dynamics.

5.8.2 Simulation

The achieved simulation is reported in Video 5.5.During the approaching phase, the yaw of the robot is modulated to maintain visibility. This is only driven by the N-MPC visibility objectives, which modulates the attitude while the AR and human move. Figure 5.11 reports the visibility task during this phase. As the plot suggests, the controller can maintain the human trunk inside the camera's FoV during the whole simulation. Large deviations from the reference value are noticeable when the human moves, and in the last portion of the plot, where the robot has to stop in the final position.

Future works will include the use of an onboard marker-less detection pipeline for the human. One of the main difficulties is to be able to locate the human from both far and close range using the same algorithm.

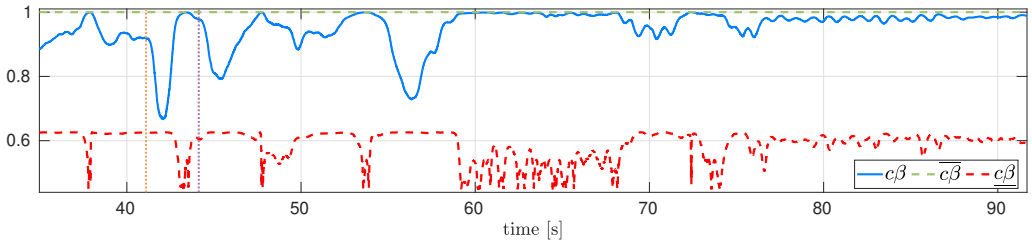


Figure 5.11: Visibility constraint over time during the approaching phase.

5.9 Conclusion

5.9.1 Synthesis

The proposed perception-aware N-MPC formulation is a control policy which considers, in a joint paradigm, the GTMR perception and control. In particular, the controller exploits the nonlinear coupling between the sensor orientation and the GTMR 6D pose to enforce the visibility coverage over a given number of features. Unlike similar prior works, it makes use of a generic GTMR model, ranging beyond standard collinear quadrotors. The controller is able to exploit the extended actuation capabilities of more complex platforms to perform the required tasks. Finally, the N-MPC is used as a full-state controller, accounting for the low-level actuation limits of the platform and generating the rotor velocities to be sent to the flight controller. The controller is provided with an onboard real-time implementation, which has been tested in simulations and experiments. Finally, the applicability of this controller is highlighted by the presentation of a practical use case: a human-robot handover.

5.9.2 Limitations

The framework faces a couple of limitations. The most prominent is that it does not allow for feature tracking without an exogenous position reference, as in Section 5.7.3. The perception objective $C^{\text{perception}}$ being the optimization of an angular value, it does not provide a satisfactory behavior when the position objective are removed. In particular, the AR goes up and tilts to reduce the β angle, and this objective alone is not sufficient to generate the adequate 3D motion. A solution consists in adding a distance penalization in $C^{\text{perception}}$, assuming that the detection is optimal at a "sweet spot", as in [Chung, 2004; Morbidi, 2013]. In the case of a downfacing camera, this is equivalent to the simple solution used in Section 5.7.3 where a reference position right above the feature is set. A complementary approach is presented in Chapter 8, which provides an objective function able to yield such tracking behavior.

Another limit is the requirement of a prior sensor/feature pairing, as discussed in Section 5.6: an online pairing strategy should be designed, in the case of sensors with a FoV overlap. It however prevents a smooth "switch" between sensors since there is a local maximum to cross. A more advanced resource allocation strategy must be designed in order to provide a satisfactory solution.

Finally, the detection model from Chapter 3 does not account for the feature orientation, thus assumes that it could be seen from any angle. In practical applications, this might not always be the case. For instance, a human coworker might

have only one visual marker on her torso. In such a scenario, additional contingent constraints could be added to the NLP. However, it would be preferable to change the detection model and assess the feature orientation, defining an explicit detection zone (e.g. a cone in front of the feature).

Chapter 6

Enforced Vision-Based Localization

Contents

6.1 Introduction **88**
6.2 Related Works **89**
6.2.1 Intertwined Control and Vision-Based Localization . . . 89
6.2.2 Minimum Requirement VISLAM Algorithm 89
6.3 Error State Kalman Filter **90**
6.3.1 Overview 91
6.3.2 Non-Vectorial Orientation Measurements 91
6.3.3 Filter Equations 92
6.4 Modified N-MPC formulation **94**
6.4.1 Visual Constraints 94
6.4.2 Visual Objectives 96
6.4.3 Optimal Control Problem 99
6.5 Experimental and Simulation Results **99**
6.5.1 Experiments with a quadrotor 100
6.5.2 Simulation with a tilted-propeller hexarotor 101
6.5.3 Motion along a longer path 102
6.6 Conclusion **104**
6.6.1 Synthesis and perspectives 104
6.6.2 Limitations Induced by the State Estimator 104

6.1 Introduction

As mentioned in Section 2.3, apart from object detection and feature tracking, one of the most widely spread uses of onboard vision is ego-localization (through VIO/VISLAM algorithms). This is particularly true with GTMR whose limited payload often prevents the use of Real-Time Kinematic GPS (RTK-GPS) devices. On the other hand, as mentioned in previous chapters, small and lightweight monocular cameras are easy to embed onboard.

To perform the localization, VIO software rely on the relative displacement of selected feature points, referred to as landmarks. On the other hand, VISLAM software rely on the pose estimation of a handful of static landmarks. In both cases, it is of paramount importance to maintain visual coverage of such landmarks, since the noisy inertial data alone are not sufficient to provide a reliable state estimation and would rapidly drift.

Thus, a perception-aware control scheme can be exploited to ensure the visibility over a sufficient amount of features. As a consequence, the platform would move around freely as long as the features are densely available, but would avoid configurations where the recovery of the state through visual cues is impossible. However, exploiting a camera for ego-localization might conflict with the requested task For instance, an exploration task or the transient phase toward a given destination might drive the AR through a feature-poor area. Such an event could compromise the quality of the pose estimation.

In this chapter, we propose perception-aware N-MPC framework, based on the one introduced in Chapter 5, and applied to the enforcement of vision-based localization using VIO or VISLAM. To achieve this goal, we propose to rework the various visual constraints and objectives imposed on the N-MPC to fit this use case. Given the scope of this chapter, the generic sensor model presented in Section 3.5 is instantiated as a monocular camera.

The proposed formulation is meant to be agnostic of the state estimator. Hence, after a short literature review of the related topics, we define a set of minimum requirement conditions that the estimator should meet to be used together with this N-MPC. Then, we present a specific example of VISLAM, exploiting AruCo markers and based on Kalman Filter. Afterward, the modified N-MPC formulation is introduced, before showing the simulation and experimental results.

The contributions of this chapter are:

- A perception-aware N-MPC formulation enforcing vision-based state estimation,
- A fully onboard implementation and its validation in simulations and experiments.

The work presented hereafter led to a submission: [Jacquet, 2022b].

6.2 Related Works

6.2.1 Intertwined Control and Vision-Based Localization

Acknowledging the existing techniques for Vision-Based state estimation, which are briefly presented in Section 2.3, [Greeff, 2020] proposes a N-MPC which enforces localization. The authors define a data-driven perception model and implement a chance constraint to produce valid observations. This approach is however path-specific since it requires a training to build the probabilistic observation model. Moreover, they make use of a gimbal-mounted stereo camera, which is not commonly found on standard platforms. Finally, their controller relies on differential flatness to linearize the system dynamics, hence is neither generalized to GTMR with larger actuation span, nor can ensure that the planned motion is feasible by the platform, leading to a potential loss of visibility over the landmarks.

An approach equivalent to the one introduced in Chapter 5 is thereby advisable. However, the proposed formulation is not transposable to this problem seamlessly. Indeed, the visibility coverage in Chapter 5 is expressed for specific object/sensor pairs, applied to phenomenon observation. In the scope of vision-based localization, it is prejudicial to maintain the visibility over specific landmarks. The goal is rather to maintain a sufficient number of landmarks in the camera FoV. All the landmarks are considered equivalent, regardless of their individual identification. Therefore, the approach from Chapter 5 or from similar N-MPC approaches such as [Penin, 2018] or [Li, 2021] is not well suited.

An approach similar to [Falanga, 2018], is in fact closer to the actual requirements of the task. Therein, the N-MPC is given incentives to maintain the detected landmarks from a VIO software close to the center of the FoV. Rather than considering the individual landmarks, the controller tries to optimize the visibility over their barycenter. However, this barycenter is precomputed outside of the N-MPC algorithm, which has no knowledge of the individual landmarks and thus cannot ensure their visibility. Therefore, it is implicitly assumed that enough feature points can be detected using this strategy, i.e., that the landmarks are dense and far from the camera. Such an assumption might be proven wrong in, e.g., some SLAM scenarios or in exploratory tasks.

6.2.2 Minimum Requirement VISLAM Algorithm

Following these considerations, a set of minimum requirements regarding the state estimator can be expressed. These requirements are motivated by the design N-MPC controller.

In order to exploit the GTMR model introduced in Section 3.4 in the N-MPC formalism, it is required to provide an estimation of every state variable before each optimization step. This implies in particular that the vision-based state estimator, be it a VIO or a VISLAM algorithm, needs to provide an estimate of the angular velocities.

Visual-inertial state estimators rely on an onboard gyroscope (included in an IMU along with an accelerometer, and sometimes a magnetometer) to provide those.

The IMU signals are affected by noise, and are subject to non-zero biases. Most of visual-inertial estimators use the high-frequency signal from the IMU as inputs, while the filter estimates only the gyroscope biases, and similarly for the linear accelerations retrieved through the accelerometer. Using the filter to estimate the angular rates and accelerations is possible [Engel, 2012]. Yet, the noisy and biased IMU signals cannot be directly integrated in the filter. A correction step is required to

1. scale the signals into the physical quantity,
2. correct the biases,
3. estimate the measurement noise associated with the measurement.

The scaling factors, biases and standard deviations are dependent on several factors, such as the temperature, and are subject to drift over time. Regular calibration of the IMU is required to re-estimate those parameters. This can be achieved using a calibration turntable [Syed, 2007]. Approaches that do not require any external material have also been proposed, with equivalent results [Fong, 2008; Tedaldi, 2014]. These methods rely on successive position schemes to estimate all the aforementioned parameters.

In order to exploit the N-MPC toward enforcing visual state estimation, the state estimation algorithm also needs to provide the individual poses of all the detected markers. VIO software should be able to provide the poses of detected features as outputs to be interfaced with the controller. In the specific case of VISLAM, the mapped markers could be also provided as the mapping is refined, even if they are not currently detected.

In the scope of this thesis, we make use of AruCo markers as landmarks to be tracked by the filter. To exploit at best the N-MPC predictive capabilities, we decided to use a VISLAM algorithm, which is able to provide a pose estimate of previously mapped tags. The markers are placed in fixed unknown positions in the workspace, and will be discovered as the GTMR moves.

We remark however that the definition of the inertial frame $\mathcal{F}_W$ is not straightforward since no external device locates the AR in a global, fixed frame, as a Motion Capture (MoCap) device or a Global Positioning System (GPS) would. It is therefore common, in visual-inertial localization systems, to choose $\mathcal{F}_W$ to be coincident with the initial body frame at the instant the system starts. The uncertainty of this initial pose can be considered but is most often neglected [Solà, 2017]. Thus the state estimator provides the relative displacement from this initial frame, denoted $\mathcal{F}_{B_0}$. One needs to carefully express all the global quantities, in particular the reference values for the motion objective C^{motion} of the N-MPC, in accordance with this choice.

6.3 Error State Kalman Filter

In order to meet the aforementioned requirements, we propose a specific example of VISLAM implementation, an EKF-based VISLAM approach, presented, e.g., in [Davison, 2003; Bloesch, 2015]. In particular, we make use of the so-called *error state*, also referred to as *indirect*, approach of Kalman filtering [Roumeliotis, 1999; Madyastha, 2011].

We first propose a brief recall of the concepts exploited in this filter. Then, we discuss the non-trivial use of quaternion orientation measurements. Finally, we detail the filter states and equations.

The maximum number of tracked markers is denoted $\overline{n_t}$, i.e. the map size; while n_t denotes the number of markers currently detected during the task. Markers detected passed the $\overline{n_t}$-*th* are ignored. We remark that since the framework does not rely on any prior knowledge of the marker poses, and that the initial configuration of the system must allow to retrieve the robot pose, we have that $0 < n_t \leq \overline{n_t}$ at each instant.

6.3.1 Overview

This approach decouples the *nominal* state (large signal) and the error state (small signal, thus linearizable and integrable). The nominal and error states are composed into the *true* state through a summation for the linear part and a suitable composition for the orientation part. It allows in particular to consider a minimal vectorial representation for the orientation error (e.g., angle-axis), while operating far from possible singularities. Operating on the error state, which is therefore small, also allows to neglect second order terms, which makes the computation of Jacobians easier and faster [Solà, 2017]. The error state KF is introduced in [Roumeliotis, 1999]. Therein, the error state is an approximate recast nominal state. It considers the orientation error as part of the state rather than the orientation itself. One of the advantages of this filtering method, claimed in [Roumeliotis, 1999], is that it can be developed with a very slow measurement period, up to the order of minutes. Additionally, the filter continues to provide estimates by acting as an integrator on the system state, in case of filter failure, since the prediction is decorrelated with the correction, and is thereby more robust. Such a filter is also proven more robust and more accurate in a comparative study with classical EKF approaches in [Madyastha, 2011].

The nominal and error states are respectively denoted $\mathbf{x}$ and $\delta\mathbf{x}$. The error-free dynamics of the system are integrated into $\mathbf{x}$, while errors related to noise and model imperfections are accumulated into $\delta\mathbf{x}$, and estimated using the ESKF. Afterward, the measurements make this error observable, providing a posterior Gaussian estimate of the error. Finally, the error is injected into the nominal state, providing a proper estimate of the true system state.

A comprehensive methodology for robot state estimation using an ESKF can be found in [Solà, 2017]. It details the concepts and equations with both quaternion or matrix representation of the orientation, based on some Lie Theory elements to work with the orientation manifold [Solà, 2018].

6.3.2 Non-Vectorial Orientation Measurements

A non-trivial aspect of this approach, which is not tackled in [Solà, 2017], is the use of non-vectorial measurements subject to non-additive noise. Such is the case, e.g., if the orientation measurements are formalized as a quaternion. As discussed in Section 3.3.2, $\mathbb{S}$ being a group, it is closed under multiplication. Consequently, and

opposed to vectorial measurements, the orientation measurement for quaternions is written (following, e.g., [Lu, 2019]) as

$$\mathbf{q}_m = \mathbf{q} \otimes \mathbf{q}_\mathbf{v} \in \mathbb{S}, \tag{6.1}$$

where $\mathbf{q}$ is the robot orientation, $\mathbf{q}_m$ is the measurement retrieved from the sensor, and $\mathbf{q}_\mathbf{v} = \text{Exp}(\mathbf{v})$ is the orientation noise, with $\mathbf{v} \sim \mathcal{N}(0, \mathbf{R})$ the Gaussian noise in its angle-axis representation, following a normal law of covariance $\mathbf{R} \in \mathbb{R}^{3\times3}$.

In the formalism of Kalman filtering, the orientation measurement model is obtained by exploiting Equation (6.1)

$$\mathbf{z}_\mathbf{q} = \mathbf{h}(\mathbf{x}) \otimes \mathbf{q}_\mathbf{v} \in \mathbb{S}, \tag{6.2}$$

where $\mathbf{x}$ is the KF state, $\mathbf{h}$ is the observation function for the orientation, $\mathbf{z}_\mathbf{q}$ is the measured quaternion orientation, and $\mathbf{q}_\mathbf{v}$ is the measurement noise.

Nonetheless, the residual cannot be written as $\mathbf{res} = \mathbf{z}_\mathbf{q} - \mathbf{h}(\mathbf{x})$. Rather, the residual on the manifold can be linearized, giving the observation model for $\delta\mathbf{x}$:

$$\mathbf{res} = \text{Log}(\mathbf{h}(\mathbf{x})^{-1} \otimes \mathbf{z}_\mathbf{q}) \tag{6.3a}$$
$$\approx \mathbf{H}\delta\mathbf{x} + \mathbf{D}\mathbf{v}, \tag{6.3b}$$

where

$$\mathbf{H} = \left.\frac{\partial \mathbf{h}(\mathbf{x})}{\partial \delta\mathbf{x}}\right|_{\delta\mathbf{x}=0}, \tag{6.3c}$$
$$\mathbf{D} = \left.\frac{\partial \mathbf{z}_\mathbf{q}}{\partial \mathbf{v}}\right|_{\mathbf{v}=0}, \tag{6.3d}$$

are the Jacobians used to propagate the error state covariance. The associated observation covariance is defined as

$$\mathbf{\Sigma}_{\mathbf{z}_\mathbf{q}} = \mathbf{D}\mathbf{R}\mathbf{D}^\top, \tag{6.4}$$

which is used to compute the Kalman gain (see Equation (6.12)).

A general formalism for ESKF on manifolds, using multiplicative noise measurements and detailing the equations, can be found, e.g., in [He, 2021].

6.3.3 Filter Equations

6.3.3.a State and Measurements Parametrization

The ESKF state is defined as the concatenation of the GTMR and landmarks states:

$$\mathbf{x} = [{}^W\mathbf{p}_B^\top \; {}^W\mathbf{q}_B^\top \; {}^B\mathbf{v}_B^\top \; {}^B\boldsymbol{\omega}_B^\top \; {}^B\mathbf{a}_B^\top \; {}^W\mathbf{t}_1^\top \dots {}^W\mathbf{t}_{\overline{n_\text{t}}}^\top]^\top, \tag{6.5}$$

where $\overline{n_\text{t}}$ is the maximum number of tracked features, and ${}^W\mathbf{t}_i$ is the state of the *i-th* feature in the inertial frame $\mathcal{F}_W$. This state is filtered using a simplified motion model, i.e., assuming constant linear accelerations and angular velocities. We remark that while it is important to consider the model nonlinearities from the controller point of view, in order to to predict an accurate behavior of the system, the resulting

motion can be instead observed as linearized, given a sufficiently small sampling time.

Then, the error state is defined, dropping the reference frames for legibility, as

$$\delta\mathbf{x} = [\delta\mathbf{p}^\top \ \delta\boldsymbol{\theta}^\top \ \delta\mathbf{v}^\top \ \delta\boldsymbol{\omega}^\top \ \delta\mathbf{a}^\top \ \delta\mathbf{t}_1^\top \dots \delta\mathbf{t}_{\overline{n_\mathrm{t}}}^\top]^\top, \tag{6.6}$$

where $\delta\boldsymbol{\theta} = \mathrm{Log}(\delta\mathbf{q})$ is the angular error associated to the orientation error $\delta\mathbf{q}$ [Solà, 2018].

Remark. We chose $\boldsymbol{\omega}$, $\delta\boldsymbol{\omega}$ and $\delta\boldsymbol{\theta}$ to be expressed in the local frame $\mathcal{F}_{B}$, following the model from Chapter 3.

The *i-th* observed feature state ${}^{W}\mathbf{t}_i$ can be defined in several ways [Bloesch, 2015]. The immediate parameterization is to consider the 6D pose of each feature independently, which allows the camera to observe jointly the feature and body poses. Since the landmarks are assumed static in the inertial frame, they are less subject to process noise than the body state, hence they would not drift as the GTMR moves. Thus, we define

$${}^{W}\mathbf{t}_i = \begin{bmatrix} {}^{W}\mathbf{p}_i \\ {}^{W}\mathbf{q}_i \end{bmatrix} \in \mathbb{R}^7. \tag{6.7}$$

Remark. This assumes a sensor model that slightly differs from the one introduced in Section 3.5, since the measurement comprises the 6D pose of the object of interest rather than being position-only. The measurement noise $\mathbf{R}$ is computed using the same approach as presented in Section 3.7.

Nevertheless, the main drawback of this approach is that the state grows linearly with $\overline{n_\mathrm{t}}$, each new tracked feature adding 7 new state variables. This is counterbalanced by the small number of features required to retrieve with accuracy the body state.

Finally, the filter typically uses measurements from the camera to observe conjointly $\delta\mathbf{p}$, $\delta\boldsymbol{\theta}$ and $\delta\mathbf{t}_i$, and an IMU to observe directly $\delta\boldsymbol{\omega}$ and $\delta\mathbf{a}$. Additionally, other onboard sensors can be used to improve the estimation, e.g. a magnetometer or an altimeter.

Remark. The accelerometer does not actually observe ${}^{B}\mathbf{a}$, but rather ${}^{B}(\mathbf{a} - \mathbf{g})$. Therefore, the observation function $\mathbf{h}$ is designed to compensate for the gravity w.r.t. the initial reference orientation ${}^{W}\mathbf{q}$. The uncertainty of the relative orientation ${}^{B}\mathbf{q}_{W}$, as well as the uncertainty associated with ${}^{W}\mathbf{g}$, are reverberated through the measurement covariance matrix $\mathbf{R}$.

6.3.3.b Discrete Time ESKF

This section presents the equations of the proposed filter. The filter prediction step equation, assuming constant accelerations and angular rates, is given by

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \begin{bmatrix} \mathbf{R}_\mathbf{q}(\delta t\mathbf{v} + \frac{1}{2}\delta t^2\mathbf{a}) \\ \mathbf{q} \otimes \mathrm{Exp}(\delta t\boldsymbol{\omega}) \\ \delta t\mathbf{a} \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \tag{6.8}$$

where $\mathbf{R_q}$ is the rotation matrix associated with the quaternion $\mathbf{q}$.

The system transition matrix is

$$\mathbf{F} = \begin{bmatrix} \mathbf{I}_3 & -\mathbf{R_q}[\delta t\mathbf{v} + \frac{1}{2}\delta t^2\mathbf{a}]_\times & \delta t\mathbf{R_q} & \mathbf{O}_3 & \frac{1}{2}\delta t^2\mathbf{R_q} & \\ \mathbf{O}_3 & \mathbf{R}_{\mathrm{Exp}(\delta t\boldsymbol{\omega})^*} & \mathbf{O}_3 & \delta t\mathbf{I}_3 & \mathbf{O}_3 & \\ \mathbf{O}_3 & \mathbf{O}_3 & \mathbf{I}_3 & \mathbf{O}_3 & \delta t\mathbf{I}_3 & \mathbf{O}_{15\times 7n_t} \\ \mathbf{O}_3 & \mathbf{O}_3 & \mathbf{O}_3 & \mathbf{I}_3 & \mathbf{O}_3 & \\ \mathbf{O}_3 & \mathbf{O}_3 & \mathbf{O}_3 & \mathbf{O}_3 & \mathbf{I}_3 & \\ & & \mathbf{O}_{7n_t\times 15} & & & \mathbf{I}_{7n_t} \end{bmatrix}, \tag{6.9}$$

which is used to predict the error state covariance $\mathbf{P}$, through the equation

$$\mathbf{P}_{k+1} = \mathbf{F}\mathbf{P}_k\mathbf{F}^\top + \mathbf{Q}. \tag{6.10}$$

From there, the measurements from both visual and inertial sources are provided and accumulated into $\delta\mathbf{x}$. For the orientation quaternion measurements, the observation model is given in Section 6.3.2. Otherwise, the vectorial measurements are defined as

$$\mathbf{z} = \mathbf{h}(\mathbf{x}) + \mathbf{v}, \quad \mathbf{v} \sim \mathcal{N}(0, \mathbf{R}). \tag{6.11}$$

Then, the correction step equations are

$$\mathbf{K} = \mathbf{P}_k\mathbf{H}^\top(\mathbf{H}\mathbf{P}_k\mathbf{H}^\top + \mathbf{R})^{-1}, \tag{6.12}$$

$$\delta\mathbf{x}_{k+1} = \mathbf{K}(\mathbf{z} - \mathbf{h}(\mathbf{x}_k)), \tag{6.13}$$

$$\mathbf{P}_{k+1} = (\mathbf{I}_{15+7n_t} - \mathbf{K}\mathbf{H})\mathbf{P}_k, \tag{6.14}$$

where $\mathbf{H}$ is the jacobian of $\mathbf{h}$ w.r.t. $\delta\mathbf{x}$.

The $\mathbf{H}$ matrix is obtained using the chain rule, as

$$\mathbf{H} = \frac{\partial\mathbf{h}(\mathbf{x})}{\partial\delta\mathbf{x}} = \frac{\partial\mathbf{h}(\mathbf{x})}{\partial\mathbf{x}}\,\frac{\partial\mathbf{x}}{\partial\delta\mathbf{x}}, \tag{6.15}$$

where $\frac{\partial\mathbf{h}(\mathbf{x})}{\partial\mathbf{x}}$ is the Jacobian of $\mathbf{h}$ w.r.t. its own arguments, and $\frac{\partial\mathbf{x}}{\partial\delta\mathbf{x}}$ is the Jacobian of the composition between the nominal and the error states, i.e.

$$\frac{\partial\mathbf{x}}{\partial\delta\mathbf{x}} = \begin{bmatrix} \mathbf{I}_3 & & \\ & \mathrm{Exp}(\delta\boldsymbol{\theta}) & \\ & & \mathbf{I}_{9+7n_t} \end{bmatrix} \tag{6.16}$$

6.4 Modified N-MPC formulation

6.4.1 Visual Constraints

In order to perform a reliable state estimation, the GTMR must always maintain visibility on some markers while performing its task. As mentioned in Section 6.2.1, it is important to let the robot move freely without accounting for visibility over specific markers. Hence, the constraints introduced in Section 5.4 are not suited.

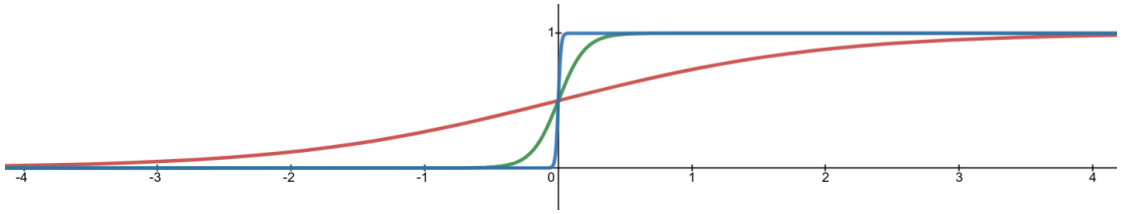


Figure 6.1: Graph of the logistic function (Equation (6.19)) for $\lambda = 1$ (red), $\lambda = 10$ (green) and $\lambda = 100$ (blues), with $x_0 = 0$.

Henceforth, we propose to integrate the constraint that "at least $\underline{n_\mathrm{t}} > 0$ marker(s) should be visible at each instant". It mathematically translates as the inequality constraint

$$\underline{n_\mathrm{t}} \leq \sum_i \xi_i = \xi \leq \overline{n_\mathrm{t}}, \tag{6.17}$$

where the quantity

$$\xi_i = \begin{cases} 1 \text{ if the marker } i \text{ is inside the FoV} \\ 0 \text{ otherwise} \end{cases} \tag{6.18}$$

transcribes the Boolean value associated with the visibility constraint defined in (4.10) and instantiated either in Inequations (5.7) or (5.13), according the the FoV shape.

The newly introduced variables ξ_i are functions of the state and parameters, hence can be propagated over the receding horizon. Nonetheless, these Boolean values cannot be integrated in the N-MPC as they are, since they would induce a discontinuity. This is circumvented by continuously approximating Equation (6.18). Boolean functions can typically be approximated through the use of *sigmoid* functions. In particular, sigmoids are monotonic and differentiable of $\mathbb{R}$, and have two horizontal asymptotes in $\pm\infty$, typically 0 and 1. They are inflected in a given abscissa, and can be tuned to tend more or less quickly toward the asymptotes.

In particular, the so-called logistic function (see Figure 6.1)

$$\begin{array}{rccc} f\colon & \mathbb{R} & \longrightarrow &]0,1[\\ & x & \longmapsto & \dfrac{1}{1+e^{-\lambda(x-x_0)}} \end{array}, \tag{6.19}$$

where $x_0 \in \mathbb{R}$ is the inflection abscissa and $\lambda \in \mathbb{R}^+$ defines the steepness of the transition from 0 to 1, is commonly used to approximate Boolean functions, because of the limit case

$$f \xrightarrow[\lambda \to +\infty]{} \left(x \mapsto \begin{cases} 0 & \text{if } x < x_0 \\ \frac{1}{2} & \text{if } x = x_0 \\ 1 & \text{if } x > x_0 \end{cases}\right). \tag{6.20}$$

In the case of a conic-shaped FoV, ξ_i is approximated as

$$\xi_i \approx \frac{1}{1+e^{-\lambda\left(\mathrm{c}\alpha - \mathrm{c}\beta_i\right)}}. \tag{6.21}$$

On the other hand, for a pyramidal-shaped FoV, a similar approximation is obtained as the product of two logistic functions:

$$\xi_i \approx \frac{1}{1+e^{-\lambda\left(\tan\alpha_h - \left|\frac{^c x_i}{^c z_i}\right|\right)}} \times \frac{1}{1+e^{-\lambda\left(\tan\alpha_v - \left|\frac{^c y_i}{^c z_i}\right|\right)}} \tag{6.22}$$

This approximation needs to be tuned through the choice of λ, such that the transition is not too steep to avoid numerical instability. It should also be steep enough such that the constraint Inequation (6.17) does not get fictitiously unsatisfied, e.g. if several features are visible and the summation of the corresponding ξ_i remains below $\underline{n_t}$ because of the approximation imprecision. In practice, $\lambda = 100$ provides a good compromise.

Remark. In Equations (6.21) and (6.22), an additional term can be included to depict the depth limit from Inequation (5.6), as in [Liu, 2017]. This term is hereafter omitted for simplicity.

The VISLAM from Section 6.3 provides an estimate of the positions of all the n_t known markers, regardless of their current visibility. These are considered in the computation of ξ and exploited by the N-MPC, as the loss of visibility over a feature could be compensated by the recovery of others.

6.4.2 Visual Objectives

6.4.2.a Barycenter of Bearing Vectors

Using arguments similar to those discussed in Section 6.4.1, for the sake of localization, there is no reason to optimize visibility over some specific markers if this opposes the requested motion task, and if some others are also available. Therefore, similarly to [Falanga, 2018], we propose to consider the barycenter of all detected features, whose angle w.r.t. $\mathbf{z}_C$ has to be minimized. Contrary to [Falanga, 2018], the aforementioned visibility constraint ensures the visibility over enough markers for the barycenter to be properly defined and computable.

Let us denote the M the barycenter of all the detected landmarks $M_i,\ i \in \{1,\ n_t\}$. The position of M is computed as

$$^C\mathbf{p}_M = \frac{1}{n_t}\sum_i {}^C\mathbf{p}_{M_i}. \tag{6.23}$$

In a formalism similar to Chapter 5, we denote β the angle between $^C\mathbf{p}_M$ and $\mathbf{z}_C$, and $c\beta$ is expressed as

$$c\beta = \frac{^C\mathbf{p}_M \cdot \mathbf{z}_C}{\|^C\mathbf{p}_M\|} = \frac{^C z_M}{\|^C\mathbf{p}_M\|}. \tag{6.24}$$

Instead of minimizing $1 - c\beta$, an alternative can be phrased making use of the triangle inequality, since all the quantities are strictly positive:

$$c\beta = \frac{^C z_M}{\|^C\mathbf{p}_M\|} \tag{6.25a}$$

$$= \frac{\cancel{\frac{1}{n_\mathrm{t}}} \sum_i {}^C z_{M_i}}{\left\| \cancel{\frac{1}{n_\mathrm{t}}} \sum_i {}^C \mathbf{p}_{M_i} \right\|} \tag{6.25b}$$

$$\geq \frac{\sum_i {}^C z_{M_i}}{\sum_i \left\| {}^C \mathbf{p}_{M_i} \right\|} \tag{6.25c}$$

$$\geq \sum_i \frac{{}^C z_{M_i}}{\sum_i \left\| {}^C \mathbf{p}_{M_i} \right\|} \tag{6.25d}$$

$$> \sum_i \frac{{}^C z_{M_i}}{\left\| {}^C \mathbf{p}_{M_i} \right\|} \tag{6.25e}$$

$$\mathrm{c}\beta > \sum_i \mathrm{c}\beta_i > \frac{1}{n_\mathrm{t}} \sum_i \mathrm{c}\beta_i \tag{6.25f}$$

In other words, Equation (6.25f) implies that the barycenter of the $\mathrm{c}\beta_i$ is upper bounded by the "$\mathrm{c}\beta$ of the barycenter". The equality is never reached because of the transition from Equation (6.25d) to Equation (6.25e), which is always strict given the sensor model from Section 3.5, implying ${}^C z_{M_i} > 0$ (and corollary, $\left\| {}^C \mathbf{p}_{M_i} \right\| > 0$). It thus entails that

$$1 - \mathrm{c}\beta < 1 - \sum_i \mathrm{c}\beta_i \tag{6.26a}$$

$$< n_\mathrm{t} - \sum_i \mathrm{c}\beta_i \tag{6.26b}$$

$$< \sum_i 1 - \sum_i \mathrm{c}\beta_i \tag{6.26c}$$

$$1 - \mathrm{c}\beta < \sum_i (1 - \mathrm{c}\beta_i) \tag{6.26d}$$

Therefore, the minimization of β can be equivalently rephrased as the minimization of $\sum_i (1 - \mathrm{c}\beta_i)$. This reformulation is advantageous implementation-wise, to avoid possible numerical singularities arising in the model induced by the normalization in Equation (6.24).

6.4.2.b Weighting the Barycenter

Contrary to Inequation (6.17), considering all the n_t tags for the computation of the barycenter is not desirable. Doing so would give an incentive to move the camera FoV away from the markers that are currently detected, and are thus actively used for the state estimation. Furthermore, it is desirable to increase the importance of the markers which are providing the best measurements in the cost function, since they contribute to a better precision in the GTMR state estimation.

These two aspects can be addressed by assigning a weight λ_i for each marker in the computation of the barycenter in Equation (6.23). This weight should be designed in such a way that it tends to 1 as the marker estimation quality improves, and rapidly decreases to 0 when the marker leaves the FoV, thus providing no benefit to the estimation process.

In order to assess the quality of the estimation retrieved from a Kalman filter, one can consider the "volume" of the uncertainty matrix $\mathbf{P}$. In particular, it is common to

consider its trace [Yang, 2007; Morbidi, 2013]. This point will be discussed in depth in Section 8.2, introducing the concepts related to Active Information Acquisition. We denote the covariance block in $\mathbf{P}$ related to the *i-th* tag by $\mathbf{P}_i$, and the related trace by $\mathrm{tr}(\mathbf{P}_i)$.

We remark that the diagonal of a covariance matrix is made of non-zero squared terms, hence $\mathrm{tr}(\mathbf{P}_i) > 0$. Thus, a suitable expression for λ_i is

$$\lambda_i = e^{-\mu\,\mathrm{tr}(\mathbf{P}_i)} \in]0,\ 1[\,, \tag{6.27}$$

where $\mu > 0$ arbitrarily defines the decreasing rate. This expression of λ_i holds the required properties:

- tends to 1 when the marker uncertainty tends to 0,
- decreases as the estimation gets worst,
- rapidly decreases toward 0 when the marker gets out of the field of view, since no new measurements are coming in the EKF to counterbalance the process noise applied to the marker state at each prediction step.

However, since the N-MPC does not include any internal representation of the ESKF uncertainty, $\mathrm{tr}(\mathbf{P}_i)$ cannot be written as function of the state, and thus cannot be propagated through the receding horizon. This quantity is rather to be computed at the initialization of each control cycle, then kept constant over the horizon. It still provides an accurate assessment of the estimation quality of each marker at a given instant, but cannot be used to predict the effect of the GTMR motion on this quality. A possible workaround strategy is to use the trace of the measurement covariance matrix $\mathbf{R}$. It can be written in closed-form as function of the GTMR-landmarks relative poses following Equation (3.38b). The N-MPC would be able to compute, for each shooting point, the uncertainty of the measurements that the sensor would produce while moving. Accordingly, denoting the individual marker measurement covariances with $\mathbf{R}_i$ and its trace $\mathrm{tr}(\mathbf{R}_i)$, the weights can be defined as

$$\lambda_i = e^{-\mu\,\mathrm{tr}(\mathbf{R}_i)} \in]0,\ 1[\,. \tag{6.28}$$

Such an approach would enable the N-MPC to predict bad marker detections, according to which uncertainty model is used for the markers. In particular, with the model presented in Section 3.6, the N-MPC would be able to account for the orientation covariance extremum occurring when the camera is frontoparallel to the marker. However, Equation (3.38b) is highly nonlinear and computationally heavy, possibly preventing an efficient convergence of the N-MPC. Exploiting the trace of the overall marker covariances $\mathbf{P}_i$ provides a reasonable compromise.

With λ_i being kept constant over the receding horizon, the normalization of the barycenter (6.26) can be omitted in the minimization problem. Finally, putting together Equations (6.23), (6.24), (6.26) and (6.27) yields a new formulation for $C^{\,\mathrm{perception}}$ for the specific problem of enhancing vision-based localization:

$$C^{\,\mathrm{perception}} = w_\mathrm{p} \sum_i \lambda_i (1 - \mathrm{c}\beta_i), \tag{6.29}$$

where w_p is the scalar tunable controller weight.

6.4.3 Optimal Control Problem

Following the paradigm from Section 4.4.3, the discrete-time NLP, over the receding horizon T, sampled in N shooting points, at a given instant t, is expressed as

$$\min_{\substack{\mathbf{x}_0 \dots \mathbf{x}_N \\ \mathbf{u}_0 \dots \mathbf{u}_{N-1}}} \quad \sum_{k=0}^{N} \|\mathbf{y}_k - \mathbf{y}_{r,k}\|^2_{\mathbf{W}_\mathrm{m}} + \sum_{k=0}^{N} \left(w_\mathrm{p} \sum_i \lambda_i (1 - \mathrm{c}\beta_{i,k}) \right) + \sum_{k=0}^{N-1} \|\mathbf{u}_k\|^2_{\mathbf{W}_\mathrm{i}} \tag{6.30a}$$

$$\begin{aligned}
s.t. \quad & \mathbf{x}_0 = \mathbf{x}(t) && && (6.30b)\\
& \mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k), && k \in \{0,\, N-1\} && (6.30c)\\
& \mathbf{y}_k = \mathbf{h}(\mathbf{x}_k), && k \in \{0,\, N\} && (6.30d)\\
& \mathrm{c}\beta_{i,k} = \mathbf{g}(\mathbf{x}_k, {}^{W}\mathbf{p}_{M_i}), && k \in \{0,\, N\} && (6.30e)\\
& \underline{\gamma} \leq \gamma_k \leq \overline{\gamma}, && k \in \{0,\, N\} && (6.30f)\\
& \underline{\dot{\gamma}}(\gamma_k) \leq \mathbf{u}_k \leq \overline{\dot{\gamma}}(\gamma_k), && k \in \{0,\, N-1\} && (6.30g)\\
& \underline{n_\mathrm{t}} \leq \xi_k \leq \overline{n_\mathrm{t}}, && k \in \{0,\, N\} && (6.30h)\\
& \underline{\boldsymbol{\mu}_k} \leq \boldsymbol{\mu}(\mathbf{x}_k, \mathbf{p}_k) \leq \overline{\boldsymbol{\mu}_k}, && k \in \{0,\, N\} && (6.30i)
\end{aligned}$$

Once again, the framework proposed in this chapter is schematized in the block diagram in Figure 6.2.

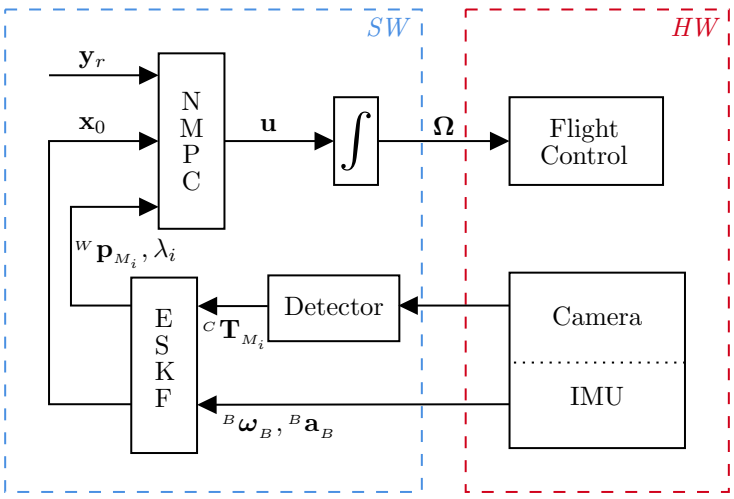


Figure 6.2: Block diagram of the proposed framework.

6.5 Experimental and Simulation Results

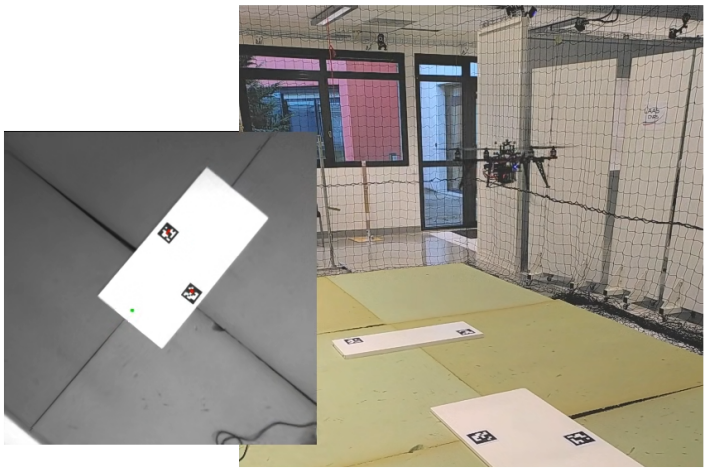
In the proposed simulations and experiments, the GTMR is equipped with a grayscale monocular camera. The image processing takes about 3 ms on CPU, using OpenCV. Although the observation of 1 marker is sufficient for the ego-localization, we impose $\underline{n} = 2$ to add redundancy and increase the resilience to, e.g., miss-detections. The controller weights used in the presented simulations and experiments are reported in Table 6.1.

Variable	Simulation weights		Experiment weights
	Quadrotor	Hexarotor	
$\mathbf{p}$	1	0.5	40
$\mathbf{q}$	1	50	80
$\mathbf{v}$	1	1	1
$\boldsymbol{\omega}$	5.10^{-1}	10	1
$\mathbf{a}$	10^{-4}	10^{-2}	10^{-4}
$\dot{\boldsymbol{\omega}}$	10^{-4}	10^{-2}	10^{-4}
$\mathbf{u}$	0	0	0
$\mathrm{c}\beta$	15	8	15

Table 6.1: Table of N-MPC weights for simulations and experiments in Section 6.5.

In the following experiments and simulations, the GTMR is a collinear quadrotor equipped with a down-facing camera, pictured in Figure A.2b, in order to underline the capability of the N-MPC to exploit the nonlinear position/attitude coupling for perception.

6.5.1 Experiments with a quadrotor



(a) Clickable image to the video on PeerTube.

(b) QR code to the video.

Video 6.1: Experiments with a quadrotor: illustration of the visibility constraint.

This first pair of experiments with a quadrotor is reported in Video 6.1. The (x, y) desired and actual trajectories are reported in Figure 6.3. The GTMR is commanded to move forward of about 2 m at constant height $z = 1$ m. As the distance to the left side markers increases, the quadrotor flies upwards to maintain the visibility, as indicated by the color gradient. In the upper graph, as the right-hand side tags enter the FoV, the quadrotor starts to descend, maintaining visibility over the newly detected features. In the experiment reported in the lower graph, as there is only one marker visible from the final position, the quadrotor does not reach the final position in altitude, but hovers at a higher altitude enforcing Equation (6.30h).

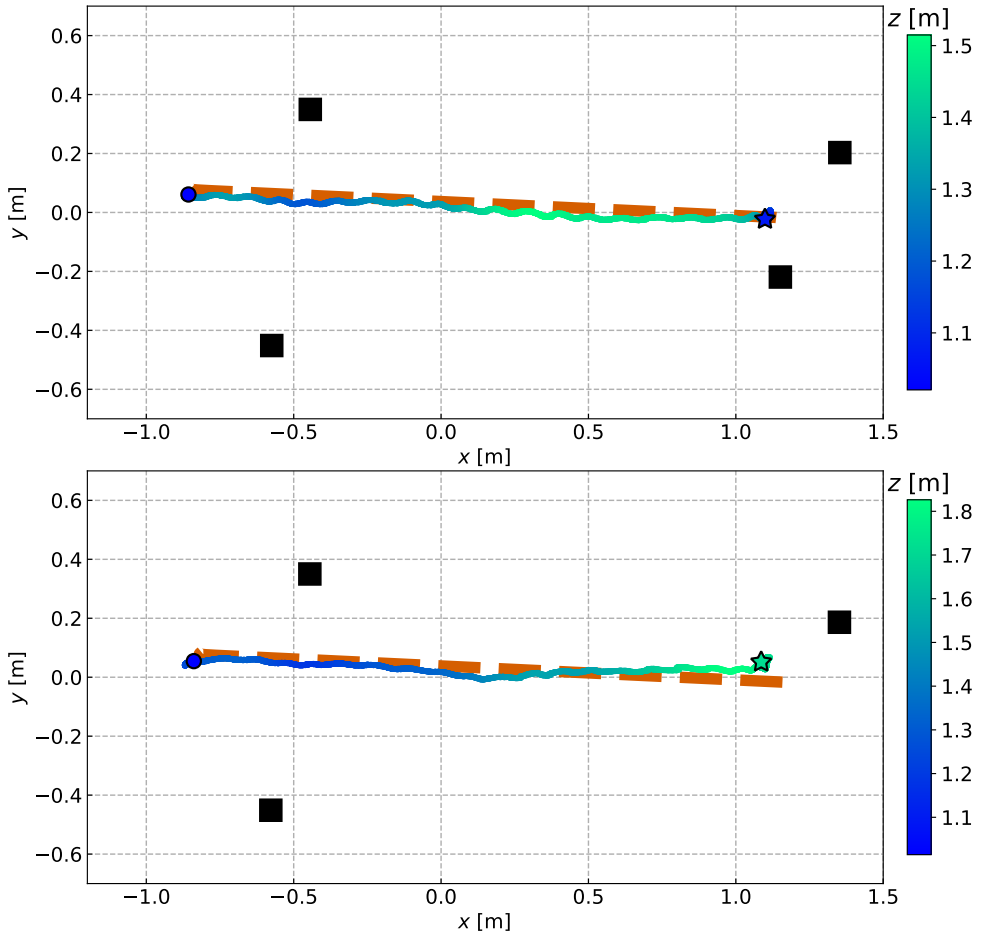
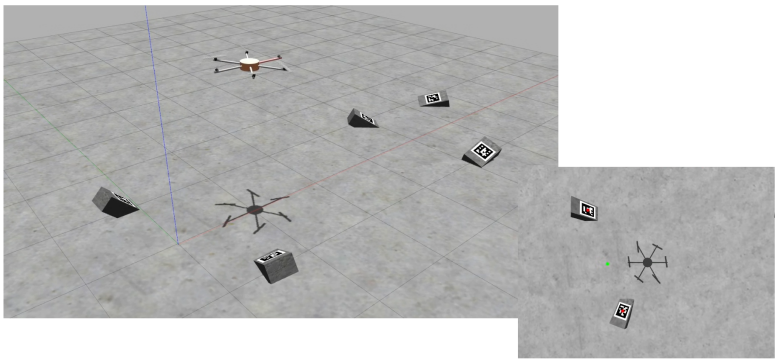


Figure 6.3: The (x, y) trajectory of the two experiments reported in Video 6.1. The z coordinate is denoted by the color gradient, following the dashed orange reference. The circle and star are the initial and final positions, while the black squares are the feature (x, y) positions (with $z = 0$).

6.5.2 Simulation with a tilted-propeller hexarotor



(a) Clickable image to the video on PeerTube.

(b) QR code to the video.

Video 6.2: Simulation with a tilted-propeller hexarotor: enforcing the visibility while moving.

A simulation in a similar scenario is performed and reported in Video 6.2, using a tilted-propeller hexarotor tasked to move forward by 4 m with z =1.5 m. The

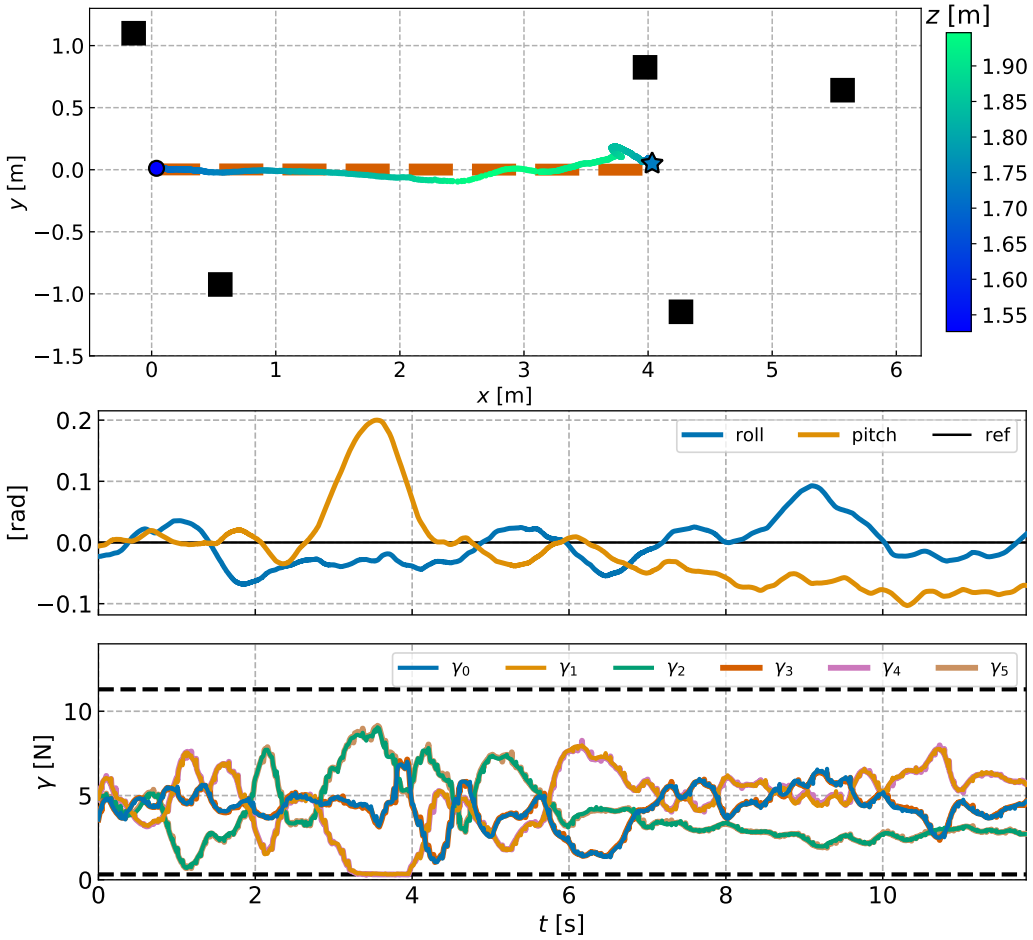


Figure 6.4: Simulation with a tilted-propeller hexarotor and a down-facing camera. Above, the (x, y) trajectory using the same presentation as Figure 6.3. Below, the roll/pitch angles of the tilted-propeller hexarotor, as well as the 6 propeller thrusts, over time.

resulting behavior is reported in Figure 6.4. The hexarotor achieves a good trajectory tracking, exploiting at best its larger actuation span both during the transient and final hovering phases. In the lower graph, around 3.5 s, the N-MPC forcibly reaches the lower bound for 2 in order to maintain the visibility.

In the final hovering phase, the hexarotor pitches in order to maintain visibility over the newly detected markers, while staying in the desired hovering position.

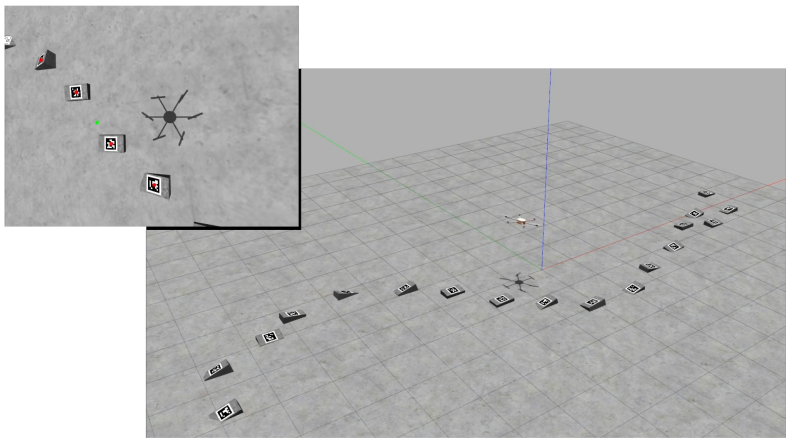
The main difference of behavior w.r.t. the one displayed using the quadrotor in Section 6.5.1 is indeed the usage on the larger actuation span to

1. rapidly counter-tilt in the transient phase as it detects the tags on the left,
2. and hover close to the desired positions by maintaining a constant pitching.

6.5.3 Motion along a longer path

The previously reported results illustrate the capability of the N-MPC to modulate its trajectory to ensure the feasibility of the localization. In this subsection, we show how $C^{\text{perception}}$ allows to mitigate a given reference to follow a feature-full path.

We propose a simulation where the GTMR follows a 16 m long path, at constant



(a) Clickable image to the video on PeerTube.

(b) QR code to the video.

Video 6.3: Two simulations with quadrotor and hexarotor: following a long path while performing state estimation.

height $z = 1.5$ m. As reported in Figure 6.5, the markers (black squares) are placed aside from the main path, in such a way that they cannot be seen by following the path at the requested altitude. Consequently, the quadrotor achieving such motion would modulate its (x, y, z) trajectory w.r.t. the reference, in order to maintain visibility over enough markers to recover its state. The hexarotor, however, exploits its larger actuation and the ability to tilt independently from the translation to achieve a better tracking of the reference trajectory.

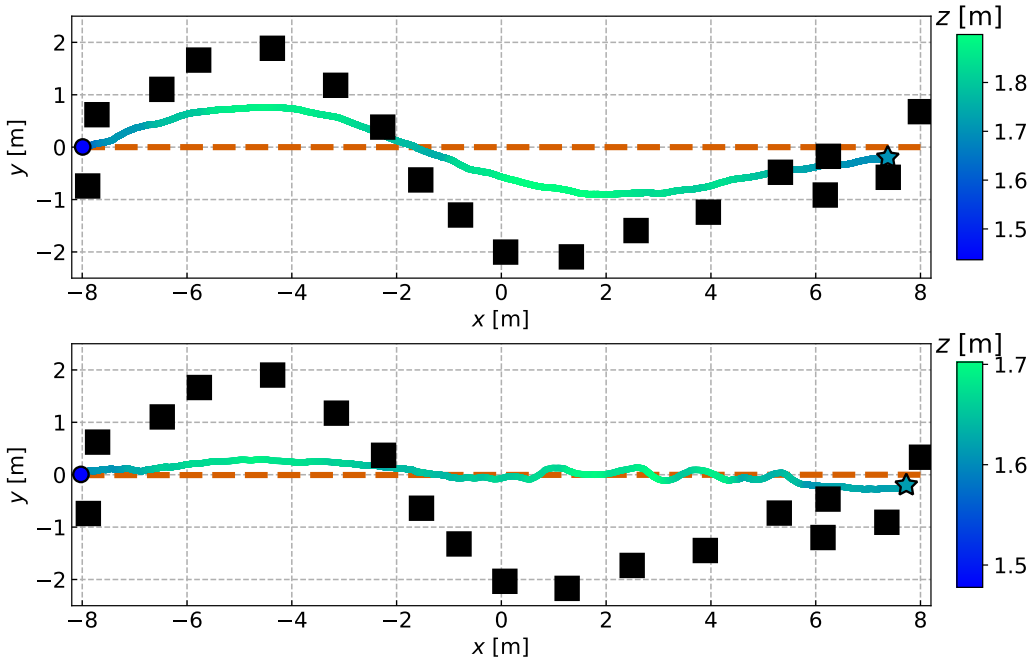


Figure 6.5: Simulations using a quadrotor (top) and a tilted-propeller hexarotor (bottom) along a 16 m path. The same presentation as Figure 6.3 is used. Both N-MPC are tuned with the same set of weights.

6.6 Conclusion

6.6.1 Synthesis and perspectives

We have shown how the N-MPC proposed in Chapter 5 can be adapted and used in the scope of enforced vision-based localization. A suitable modification of the constraints is required to consider all features as equivalent, and constrain the FoV bearing to capture a sufficient number to perform state estimation. On the other hand, a recast of the perception objective following the paradigm of [Falanga, 2018] allows to induce the desired behavior. It allows the AR to move in order to maintain the visual coverage of enough features to recover the GTMR state. The simulations and experiments performed demonstrate how the desired motion is altered to this purpose.

The immediate follow-up of the presented work would be to extend it to the use of a more comprehensive SLAM or VIO software [Campos, 2021; Delmerico, 2018], in order to improve its versatility, and to avoid resorting to any fiducial marker. The proposed framework needs to be slightly modified to be interfaced with a VIO software (i.e. that does not perform mapping), but no major changes should be performed as far as the N-MPC formulation is concerned. However, due to the mandatory entanglement between the state estimator and the N-MPC, depicted in Section 6.2.2, the state estimator needs to be implemented such that it outputs the various quantities exploited by the controller. Hence, the implementation work to modify an existing framework would be important.

Additionally, an exploratory behavior could be induced through a suitable term in the cost function. However, this requires a prior assumption on the marker distribution, as illustrated in the previous remark. A possible solution is the introduction of a probabilistic distribution of markers, e.g. with a rasterization of the environment in cells which may or may not contain features [Dames, 2020].

In the results presented in Section 6.5, the weighting terms λ_i from Equation (6.27) are computed externally, and kept constant over the receding horizon. Such an approach allows to consider the uncertainty of each feature estimate in the overall behavior of the system, but does not allow to generate motions improving these quantities. In Chapter 8, a scheme is proposed to propagate the uncertainty estimate in the N-MPC, which might be used to update the value of λ_i accordingly.

6.6.2 Limitations Induced by the State Estimator

An implementative limitation is the sensitivity of our proposed ESKF to the choice of $\mathbf{Q}$. Indeed, in order for the marker estimates to stay stable, the process noise applied to those in the filter must remain quite small. Corollary, the effect of the barycenter weighting described in Section 6.4.2.b does not provide a satisfactory effect regarding the assessment of the quality of observation. In fact, while the scalar weight rapidly decreases to 0 when the marker goes out of the FoV, the effect of the quality of the measurement is effectively not noticeable on the value of λ. This is directly imputed to the choice of a small process noise $\mathbf{Q}$, since it directly affects how the measurements will impact the overall estimate. As a practical result on the

overall system behavior, e.g. reported in Section 6.5.3, the final hovering position reached by the GTMR does not coincide with the final position requested by the reference trajectory. This is due to the fact that the barycenter of the observable markers is always 'in delay' w.r.t. the requested motion. We conjecture that with a more robust state estimator and a larger process noise, the proper effect of the weighting term λ_i would be to drive the AR closer to the requested hovering position. Indeed, the markers closer to the FoV boundary would be gradually disregarded, as they bear a larger uncertainty.

Remark. A workaround for this conservative behavior would be, for instance, to associate a stronger weight to the newly detected tags. This priority in the barycenter computation would induce a more exploitative behavior in the framework. Such prioritization is motivated by the assumption that the features are located in feature-rich clusters.

Chapter 7

Extension to Multi-Agent Systems

Contents

7.1 Introduction **108**
7.2 Centralized and Decentralized Implementations **108**
7.3 Collision Avoidance **110**
7.4 Constraints and Objectives with Several Agents **112**
7.4.1 Generalization of Perception Constraints 112
7.4.2 Generalization of Perception Objectives 113
7.5 Multi-Agent Collective Localization **113**
7.6 Conclusion **114**

7.1 Introduction

The N-MPC formulations presented in previous chapters are framed for the use of a single AR. In many contexts, it is beneficial to exploit the redundancy of Multi-Robots Systems (MRS). For instance, the observation burden can be spread among the whole system, thereby relaxing the constraints imposed on each agent.

We consider a team of n_{A} sensing agents, possibly heterogeneous, but complying with the model described in Chapter 3. The typical values for n_{A} to be considered are roughly around 2 to 5, after which point swarm-based implementations might be preferred, though they aren't tackled in this thesis. Yet, the proposed formulation can theoretically be extended to an arbitrary number of agents.

In this chapter, we discuss some critical aspects of the scaling to MRS, in particular the decentralization of the controller, and the inter-agent collision avoidance policies to be implemented. Then, the extension of the frameworks from Chapters 5 and 6 is discussed, and some solutions to a couple of practical considerations are introduced.

The contributions of this chapter are:

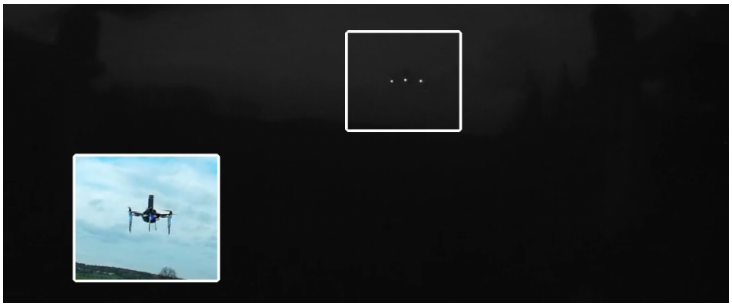
- A review of existing collision avoidance policies for MRS,
- The extension of the previously introduced framework to multi-agent scenarios.

7.2 Centralized and Decentralized Implementations

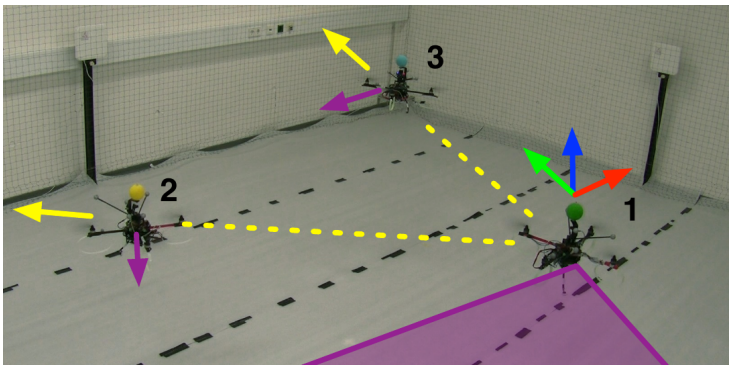
The multi-agent N-MPC can be formulated either as centralized or decentralized. In the centralized case, the N-MPC state and input vectors are made of the concatenation of those of the individual agents. The controller thus handles each agent as an independent part of a bigger system. This approach has been successfully tested with systems of 3 agents, in simulation, showing that the approach is theoretically viable. It is however poorly scalable with the number of agents. The size of the QP to be solved quickly increases, and consequently the required computational time. Since the subsystems are independent, the sparsity of the problem could be exploited to make it efficiently solvable. However, the main drawback of this approach is the inevitable latency induced by the wireless communication between the centralized controller and the individual agents. Moreover, since the N-MPC computes the motor torque inputs, a delay of several milliseconds between the issued command and the flight controller would disrupt the stability of the system. A specific, low-latency network might be introduced to address this issue, but in cannot be completely overlooked.

A decentralized approach might be used instead. In such a scenario, each agent runs its own N-MPC independently, as introduced in Chapter 4. It is however mandatory to assess the position of other agents, both for reasons of efficiency and safety This aspect is addressed in Section 7.3.

If communication is allowed, each agent can communicate its own state and measurements with the whole system. These can even be communicated for the full receding horizon, exploiting at best the predictive aspect of the N-MPCs. Since



(a) LED-based detection of an AR (courtesy of [Walter, 2018]).



(b) Colored markers for mutual detection in a multi-agent system (courtesy of [Franchi, 2012]).

(c) Circular fiducial marker used for mutual detection (courtesy of [Krajník, 2014]).

Figure 7.1: Three examples of hardware-based mutual AR localization systems.

the exchanged data do not concern the low-level actuation, which is performed individually by each agent, the communication latency is much less critical than in the centralized case.

Remark. We assume a complete communication scheme. Intermediate cases could indeed be considered (i.e., with partial communication schemes), yet this is left out of the scope of this discussion, which tackles the two limit cases.

If communication is instead not allowed, a similar strategy can be employed if the ARs are equipped with onboard sensors able to efficiently detect other agents. Such a sensor could be, e.g., additional cameras with the suited processing software [Aker, 2017]. To reduce the computational burden, some colored markers [Franchi, 2012; Tron, 2016] or even some extra fiducial markers [Krajník, 2014] can be added on the platform. There also exist some fast and efficient relative detection techniques for formation control exploiting lightweight ultraviolet LEDs [Walter, 2018]. Some examples are depicted in Figure 7.1. Consequently, each AR could run a filter to localize the other agents.

Remark. In cases where only the position of the other agents is retrieved, it can be assumed that the bearings of their sensors are maintained toward the features. The relative body-sensor poses and the allocation of the sensor/feature pairs can be known beforehand. This simplifies further the state estimation process which boils down to the filtering of a linear process with direct measurements. It can also be noted that in such cases, the various AR to track are merely additional features to maintain in sight, hence the visibility over each agent can be stated as additional constraints in a perception-aware N-MPC.

As a result, a decentralized implementation induces some additional hardware and software implementation burdens, especially in communication-less scenarios. It should however be preferred to centralized ones, since it is more scalable and more resilient to communication issues (such as latency or network blackout).

7.3 Collision Avoidance

Regardless of the choice of centralizing or decentralizing the control, it is of paramount importance to include an inter-agent collision avoidance policy in the framework. Deploying a MRS without such considerations is hazardous for the hardware, and indeed for the other users of the workspace (due to possible uncontrolled crashes, throwing of broken propeller blades, etc). This section proffers a summary of existing methods to be implemented in order to deploy a MRS in practical experiments. In the tail of the discussion from Section 7.2, the assumption is made that the positions of all ARs are known by each agent. To some extent, this position can even be assumed known through the receding horizon of the controllers (either by direct communication of the trajectories computed in the N-MPC, or by extrapolation of the current motion using a filter).

Collision avoidance in aerial MRS is a widely studied topic in the literature. Recent survey papers propose an overview of the various existing strategies [Huang, 2019; Yasin, 2020]. The first immediate approach is to detect close-range ARs (or, more generally, any obstacle) and reactively alter the trajectory for avoidance, e.g. by stopping the current motion or moving toward another direction. The main advantage is the reduced computation load imposed on the ARs. Yet, this strategy is not best suited to prevent inter-agent collisions since the ARs are not passive obstacles. Indeed one can easily picture a case where both agents attempt to prevent collisions by moving in the same direction, ultimately leading to a crash. Stopping the motion, be it for a short amount of time, is not desirable either since it might delay or prevent the correct execution of the task. As a matter of fact, accounting for the agent positions directly in the controller or the trajectory planner is more appropriate than reactive policies. In fact, the planning or control-level approaches are more commonly used, and are extensively discussed in the aforementioned surveys. Those can be classified in three groups:

- geometric methods,
- force-field methods,
- and optimization-based methods.

The geometric approaches consist of computing the relative distances amidst robots, as well as their relative velocities or other geometric attributes. For instance, a collision cone [Chakravarthy, 1998] depicting the set of velocities which would yield a collision can be exploited to analytically or numerically compute collision-free trajectories. In the scope of MPC, this class of strategies could be stated as state constraints, e.g. on position, velocities or acceleration. A guidance cost function can also be formulated, e.g. based on relative acceleration [Watanabe, 2006]. This category also encompasses the most basic avoidance scheme consisting of assigning a dedicated workspace to each agent. A common workspace division strategy is to assign

"slices" of altitude z and can be employed, e.g., in air traffic management. Despite not being satisfactory, since the possible configurations are drastically diminished and no flexibility or reorganization is allowed, this method is efficient and does not require the knowledge of any additional quantities.

Remark. Even though the state constraints would not disturb the convergence of the N-MPC until it becomes "active" (i.e. the variable gets close to the boundaries), imposing such constraints always impacts the computational time since it enlarges the final QP to be solved. Corollary, having very conservative bounds in constraints also enlarges the search space and is reverberated onto the computation time.

Force-field approaches are based on the concept of repulsive force between each agent of the MRS. Inspiration is taken from electric charge repulsion or attraction, which decreases with distance. Consequently, a virtual potential field is computed in the location of the robot, to assess whether the robot is pushed away from an obstacle (e.g., another robot). This field is necessarily statically generated around the obstacle. In [Choi, 2020], it is generated as function of the velocity vector of moving obstacles. It allows to improve the reactivity by anticipating the field shape in the near future. Repulsive force-field approaches can obviously be coupled with attractive force-field motion generation strategies [Azzabi, 2017]. This allows to generate motion toward an attractive goal (e.g., for feature tracking) whilst avoiding obstacles. This approach is however prone to local minima, in particular en symmetrical environments. Force-field methods are in general used for collision-free route planning in obstacle-dense environments, e.g. for autonomous ground vehicles. Their complex implementation makes them not suited for collision avoidance with a small number of agents. Moreover, a safe separation distance between robots is not proven in general.

Finally, optimization-based methods rely on avoidance trajectory computation based on some uncertain information data. Those methods include probabilistic approaches to minimize the collision probability. Centralized multi-agent path planning with collision constraints is often tackled using numerical optimization, e.g. in [Augugliaro, 2012]. For larger scale systems or when centralization is not allowed, local replanning strategies are employed. These optimization-based methods are indeed also suited for integration in predictive control schemes. In [Baca, 2016], potential collisions are assessed over the receding horizon of the MPC, and the height reference is modulated accordingly. This replanning is based on a prioritized planning policy, where each AR is assigned a motion priority, and only the lowest priority one modulates its reference when a conflict is detected. Given a small enough MPC sampling time and/or a sufficient size margin, the collision assessment over the horizon should not be impacted by the discretization. Otherwise, some continuous path validation [Bury, 2019] might be employed along with a proper curve fitting of the MPC shooting nodes, but such methods are typically challenging to run in real-time.

In the MRS experiments conducted for this thesis (presented in Chapter 8), a basic geometric collision avoidance scheme is used. The knowledge of the agents position at all times is exploited to define, in twos, a linear state constraint on the position variables for each pair of ARs. Therefore, the avoidance policy is phrased, using the paradigm from Chapter 4, using the motion constraints mapping $\boldsymbol{\mu}$. Such constraint is formalized by constraining the per-component difference between both

ARs to be greater than a lower bound:

$$\begin{bmatrix} 2l+\epsilon \\ 2l+\epsilon \\ 2l+\epsilon \end{bmatrix} \leq {}^{W}\mathbf{p}_{B} - \mathbf{p}_{Ai}, \tag{7.1}$$

where l is the AR radius, $\epsilon > 0$ is an arbitrary security margin, and $\mathbf{p}_{Ai}$ is the position of the i-*th* other agent, in $\mathcal{F}_{W}$, passed as parameter to the N-MPC. We thereby define

$$\underline{\boldsymbol{\mu}} = \begin{bmatrix} 2l+\epsilon \\ 2l+\epsilon \\ 2l+\epsilon \end{bmatrix} + \mathbf{p}_{Ai}. \tag{7.2}$$

7.4 Constraints and Objectives with Several Agents

7.4.1 Generalization of Perception Constraints

The visibility constraints introduced in Chapter 5 are limited to the case of a single agent. In Section 5.6, the use of several sensors is discussed, though some limits are presented therein. It turns out that the extension of this framework to multiple agents can be framed using the very same paradigm, while the sharing of the workspace among agents is considered separately, as described in Section 7.3.

In the case where the system is assigned to the tracking of a single feature, the perception constraints can be relaxed, as the measurements are exchanged and the observation burden is shared. In such cases, a policy similar to the one introduced in Section 6.4.1 is required. We define, rather than the Boolean quantity ξ_i as the visibility i-*th* feature by the agent, a similar variable, denoted ξ'_i, characterizing the visibility over the feature by the i-*th* agent. Hereby, again using a logistic approximation, a constraint similar to Inequation (6.17) can be written, assessing that a minimum number $\underline{n_{\mathrm{A}}} > 0$ of AR has to effectively maintain visibility at each time:

$$\underline{n_{\mathrm{A}}} \leq \sum_{i} \xi'_i \leq n_{\mathrm{A}}. \tag{7.3}$$

The value of $\underline{n_{\mathrm{A}}}$ is typically set to one, or two for redundancy. By knowing the expected ξ' of the full system, each agent can generate its motion accordingly.

When monitoring a set of several features, a couple of solutions are available. The first naive approach is to externally assign the tracking of each given feature to a subset of the sensing agents. This is the same policy as introduced in Chapter 5 to handle the tracking of several features with several sensors. It is indeed subject to the same limitations as those discussed in Section 5.9. It is also viable to write a set of constraints equivalent to Inequation (7.3) for each of the n_{t} tracked features. Those are consequently written as

$$\forall j \in \{1, n_{\mathrm{t}}\}, \quad \underline{n_{\mathrm{A}}} \leq \sum_{i} \xi'_{i,j} \leq n_{\mathrm{A}}, \tag{7.4}$$

where $\xi'_{i,j}$ denotes the Boolean variable assessing visibility of the i-*th* agent over the j-*th* feature.

The $\xi'_{i,j}$ can either be computed from the agents and feature poses, or directly communicated other the network. Nonetheless, if the communication frequency is low and a local filtering of the AR poses is performed, the former solution is preferable since it would improve the framework by using more accurate values. In any case, the predictive aspect of the controller is exploited and these variables are shared for the whole receding horizon.

Remark. The same solution is suitable to write a shared observability constraint when using a single AR with several sensors. In such a scenario, the constraint would state that at least one among all the sensors must maintain visibility over each feature at all times.

7.4.2 Generalization of Perception Objectives

Regarding the formalization of the perception objectives $C^{\text{perception}}$, these are rigorously the same as Equation (5.5) in the single feature case. Each AR is tasked to maintain its bearing toward the feature while performing its task.

The multiple feature case is however facing the limits discussed in Section 5.6. It implies a prior or online assignment policy of each feature to a dedicated sensor, or the multiplication of the terms in would cost function would collide and the resulting behavior would not be satisfactory. The remark in the aforementioned section, discussing the writing of a shared objective among agents, also applies here.

Finally, it can be noted that in the case of perception-based inter-localization among the agents, and if the relative body/sensor poses are known beforehand, an additional objective can be added in order to reduce the chances of visibility loss. Knowing the position and bearing of the other sensors, a behavior can be induced using a policy similar to Equation (5.5), driving the agent to stay within the other agents FoV, thus facilitating mutual localization. To this end, for each agent i, we can write such objective, denoted C_{collab}, as

$$C_{\text{collab}} = w \sum_{j=1,\ j\neq i}^{n_{\text{A}}} (1 - \text{c}\beta_{ji}), \tag{7.5}$$

where $\text{c}\beta_{ji}$ is the cosine of the bearing angle from the sensor of the j-*th* agent to the agent i, and w a weighting scalar. The latter is common for each agent j, since there is no a priori preference for this collaborative cost.

7.5 Multi-Agent Collective Localization

While the previous section shows how to extend the perception-related constraints and objectives defined in Chapter 5, this one proposes an approach to enlarge the visual-localization oriented N-MPC introduced in Chapter 6. Each agent can provide additional measurements to the whole system, in particular if agents are mutually observing each other. This additional information is used by the local state estimators, but does not affect the N-MPC formulation from Section 6.4.

Following the paradigm introduced in Section 6.3, a minimal criterion for sharing meaningful information is that each estimator shares a common reference frame $\mathcal{F}_W$.

In Section 6.3, this frame was arbitrarily chosen to be the initial body $\mathcal{F}_{B_0}$. To overcome this limitation, a priori common reference must be stated. In particular, the 6D pose of $\mathcal{F}_{B_0}$ for each agent must be assessed in a common frame. Several solutions are available and contingent on the practical use cases, e.g.:

- placing the agents in known initial positions, e.g. with markers on the ground,
- assessing the initial poses with an external device, e.g. a MoCap,
- initializing the system with all agents visually estimating their pose relative to a common and fixed landmark.

The feature poses estimation is performed in $\mathcal{F}_W$, hence ${}^W\mathbf{t}_i$ is also shared over the network and used as direct measurements, to fine-tune the individual estimates of each filter. The tags being fixed in the inertial frame, the time delay induced by the network communication is not a major problem, though a suited method can be employed to handle such delay [Larsen, 1998]. Improving the estimate of the marker poses is important since the onboard measurements of each agent are jointly estimating the body and marker states.

Remark. A larger filter estimating the full system state can be written, but is subject to two main limitations:

- first, the filter state is already large due to the simultaneous estimation of all features, and thus the scaling to several agents is problematical,
- communication of measurements would induce inevitable delays in the measurements, which need to be tackled.

Possible solutions include the so-called Distributed Kalman filter [Roumeliotis, 2000; Olfati-Saber, 2007], or multi-agent SLAM [Atanasov, 2015a]. Such solutions have not been explored in the scope of this thesis.

7.6 Conclusion

This chapter tackles the extension of previously introduced N-MPC frameworks to the scope of multi-agents scenarios. The focus has been set on maintaining the formulation close to the single agent case. The main limitation is the difficulty to scale the perception objective w.r.t. the number of agents/sensors, as discussed in Chapter 5. Moreover, while this simple MRS scheme allows to leverage the extra sensing capability in order to relax the individual burden imposed on each agent, the multiplicity of sensors is not exploited to improve the feature state estimation. Given the proposed formulation, the agents would not be actively cooperating toward improved sensing. Intuitively, it can be conjectured that observing the feature from different angles allows to retrieve more information. In the literature, the problem of actively generating motions providing "better" observations is studied. In particular, it can be achieved through the use of several sensing robots. The following chapter is devoted to the use of such techniques in combination with our N-MPC formulation.

Chapter 8

Active Information Acquisition

Contents

8.1 Introduction **116**
8.2 Literature Review **116**
8.2.1 Problem Definition and Formalization 116
8.2.2 Choice Information Metric 118
8.2.3 Active Information Acquisition with Aerial Robot 119
8.3 Intermittent Observation Modeling **121**
8.3.1 Intermittent Kalman Filter 121
8.3.2 Intermittent Observations 122
8.3.3 Observation Uncertainty Model 122
8.4 Leverage Uncertainty Minimization in N-MPC **124**
8.4.1 N-MPC with the Kalman Filter state 124
8.4.2 AIA with Multiple Sensing Agents 125
8.4.3 NonLinear Programming 126
8.5 Experimental and Simulation Validation **128**
8.5.1 Static Feature Sensing 128
8.5.2 Asymmetric Sensing Team 131
8.5.3 Mobile Feature Tracking 133
8.6 Conclusion **135**
8.6.1 Synthesis and Discussion w.r.t. State of the Art 135
8.6.2 Scalability w.r.t. the Number of Features 136

8.1 Introduction

The quality of the detection is not accounted for in the framework introduced in Chapter 5, which cannot predict the accuracy of future measurements. In particular, the observation distance and angle can induce measurements with large uncertainty. This is prejudicial for the perception task, in particular if the feature pose is to be estimated accurately. Moreover, a large reprojection might lead to an unexpected loss of visibility, compromising the feasibility of the various tasks. Thus, it is desirable to introduce some knowledge of the detection quality in the N-MPC. This also entails the possibility to generate motions that effectively improve this quality.

More generally, the problem of optimizing the quality of observations through the generation of appropriate motion is referred to as an Active Information Acquisition (AIA) problem, and has been widely studied in the literature over the past years.

This chapter introduces a N-MPC formulation to solve an AIA problem. We choose to restrain the domain of interest to the monitoring of a single, mobile feature. The various challenges induced by the scaling to multiple features are discussed in Section 8.6.2. Given the nature of AIA problems, they are often tackled using MRS, since the addition of sensors is leveraged to improve the observation quality. Thus, a team of $n_{\mathrm{A}} \geq 1$ agents is considered, all of which comply to the GTMR model but may be heterogeneous, i.e., having different shapes, number of rotors or sensing parameters. Throughout this chapter, when relevant, the quantities referring to the *i-th* agent are subscripted $\bullet_i$.

First, we propose a literature review on the AIA problem. Then, we detail the mathematical modeling and our methodology, in particular to integrate the uncertainty minimization over the receding horizon. The formulation is valid both for single robot or MRS. Finally, some experimental and simulation results are presented, demonstrating emergent collaborative behaviors.

The contributions of this chapter are:

- A N-MPC framework for collaborative optimal sensing of a mobile feature with an heterogeneous sensing team,
- The integration, in the N-MPC, of a Kalman-Bucy intermittent filter,
- A fully onboard implementation and its validation in simulations and experiments.

The work presented hereafter led to a publication: [Jacquet, 2022a].

8.2 Literature Review

8.2.1 Problem Definition and Formalization

AIA problems, also referred to as *Active Sensing* problems, have been studied in the literature for decades. Early work on optimizing configurations for mobile sensing systems dates back to the late 60s [Meier, 1967; Athans, 1972], where first controllable sensor systems have been studied. This set of problems is related to, e.g.,

sensor selection [Joshi, 2008], sensor placement [Vitus, 2012] or sensor scheduling [Le Ny, 2010]. Other approaches consider the actual control of dynamic sensors for active object detection [Sommerlade, 2008; Eidenberger, 2010]. Applications of such techniques include, e.g. environmental monitoring [Tokekar, 2014], search-and-rescue [Kumar, 2004], active object detection [Atanasov, 2014a], or distributed SLAM [Atanasov, 2015a].

AIA can be generically framed as the problem of leveraging the mobility of a sensor to increase detection performances. In the scope of robotics, it translates as generating motion to improve detection from the onboard sensor. More specifically, the problem is not only formalized as the problem of allowing the detection, but is associated with a quantitative metric, often called *information measurement* or *metric* [Atanasov, 2015b]. Consequently, concepts from estimation theory, filtering and belief propagation are exploited in order to express this information metric, denoted $\mathcal{I}$. The metric $\mathcal{I}$ has to be defined according to the problem which is tackled. For instance, it can be characterized as the quantity of information gathered in an exploration task [Le Ny, 2009; Jang, 2020], the cumulative probability of detection [Bourgault, 2004], or as the "amount" of feature estimation uncertainty [Yang, 2007; Schlotfeldt, 2019].

To solve the problem, three mathematical models are conjointly exploited:

1. the motion model of the sensor, which is in our case coincident with the motion model of the GTMR since the sensor is rigidly attached to the body;
2. the motion model of the observed feature;
3. the observation model of the sensor.

A feature estimator is designed, exploiting the sensor observations and feature motion models. Such a motion model is often basic, e.g. a constant velocity/acceleration model or a double integrator. The observation model depends of course on the sensor mounted on the platform. Thereby, a suitable motion is generated in order to collect observation maximizing the retrieved information, optimizing the information metric $\mathcal{I}$.

These AIA problems are either tackled in the scope of multi-robot cooperation [Yang, 2007; Schlotfeldt, 2019], or the motion generation of a single robot [Chen, 2016; Liu, 2017; Thomas, 2017]. In the first case, the focus is often set on cooperation among agents, e.g. through the maximization of mutual information [Charrow, 2014] in exploratory or multi-feature tracking scenarios. These approaches often make use of simplified linear motion models for sensing robots. In the second case, the motion generation for the agent is often exploited in single-feature tracking scenarios, sometimes in complex environments.

Such AIA problem can be solved through various approaches. A cell-based rasterization of the workspace can be employed to drive the robots to maximize the explorative coverage through a probabilistic search algorithm. It allows in particular to exploit efficient minimization algorithms in combination with a suited filter and cell representation. The works conducted in [Dames, 2020] make use of the PHD filter [Mahler, 2003] which naturally pairs with Voronoi cell-based control algorithms, e.g. Lloyd's iterative minimization algorithm [Cortes, 2004; Pimenta, 2008]. Because of the stochastic nature of these algorithms, they cannot be applied to the frame of

N-MPC. These approaches are however highly scalable w.r.t. the amount of tracked features.

Another solution is to separate the estimation process from the controller design. The optimal sensor configuration can be analytically computed, for a given scenario, and the controller is designed in order to maintain the formation [Martínez, 2006; Tallamraju, 2019]. The optimal configuration is however hard to obtain analytically in general, and this approach is highly task-specific.

Therefore, some works propose to compute the optimal configuration online, through the use of optimization algorithms, using a shared estimator and letting each agent move to reduce the overall uncertainty [Yang, 2007; Schlotfeldt, 2019]. Then, the AIA problem is cast as a stochastic OCP. However, it is demonstrated in [Atanasov, 2015b] that under the correct assumptions, the separation principle [Meier, 1967; Athans, 1972] between estimation and control holds. Such assumptions are:

- the target motion model is linear and subject to Gaussian noise;
- the observation model is linear in the feature state (not necessarily in the GTMR state;
- the GTMR motion model is deterministic;
- the information metric $\mathcal{I}$ is a function of the GTMR state and the feature estimation covariance matrix.

Hence the stochastic OCP can be equivalently rephrased as a deterministic OCP [Atanasov, 2014b].

In general, denoting the GTMR state $\mathbf{x}$, its inputs $\mathbf{u}$, the feature state $\mathbf{x}_M$, its covariance $\mathbf{P}$, and the sensor measurements $\mathbf{z}$, the AIA problem can be stated as

$$\begin{aligned} \min_{\mathbf{u}} \quad & C(\mathcal{I}(\mathbf{x}, \mathbf{P})) \\ s.t. \quad & \dot{\mathbf{x}} = \mathbf{f}_r(\mathbf{x}, \mathbf{u}) \\ & \dot{\mathbf{x}}_M = \mathbf{f}_m(\mathbf{x}_M, \text{noise}), \\ & \mathbf{z} = \mathbf{f}_o(\mathbf{x}, \mathbf{x}_M, \text{noise}) \\ & \dot{\mathbf{P}} = \boldsymbol{\kappa}(\mathbf{x}_M, \mathbf{z}, \text{noise}) \end{aligned} \tag{8.1}$$

where C is a suitable cost function according to the choice of $\mathcal{I}$, $\mathbf{f}_r$ and $\mathbf{f}_m$ respectively denote the robot and feature dynamics, $\mathbf{f}_o$ is the sensor observation model, and $\boldsymbol{\kappa}$ synthetically denotes the uncertainty propagation, e.g. the Kalman filter Riccati map. This problem is most often discretized rather than continuous, and recent works tend to propose non-myopic formalisms, yet greedy approaches have been the most widely used.

Remark. The AIA problem is sometimes framed as the maximization of information rather than the minimization of uncertainty. Since we are interested in active pose estimation, the formalism of Equation (8.1) is more explanatory.

8.2.2 Choice Information Metric

Regarding the definition of $\mathcal{I}$, two possible schemes are presented in [Chung, 2006] to reduce the estimation uncertainty. First, one could consider the minimization

can be the measurement covariance matrix $\mathbf{R}$. In case of multiple sensing agents, the measurement matrices can be fused, according to the relation from [Bar-Shalom, 1988], which essentially consists of the inversion of the sum of inverse covariances. Second, the overall filter estimates uncertainty $\mathbf{P}$ could be used instead. No definitive conclusion is drawn in [Chung, 2006] regarding the scheme to prefer. However, the latter has been largely adopted in related works, while the former is rarely used, to the best of our knowledge. It can be instantly remarked that using $\mathbf{P}$ is strictly more computationally expensive, since it requires going one step further in the estimation process. Also, when the filter process noise approaches infinity, the two approaches converge, since the prediction would be fully discarded in favor of any incoming measurements.

In order to design $\mathcal{I}$ to effectively reduce the feature estimation uncertainty, arises the need for a mathematical tool that represents the "magnitude" of the covariance matrix $\mathbf{P}$. In information theory, minimizing the *Shannon differential entropy* of a continuous Gaussian variable is equivalent to minimizing the determinant of its covariance matrix. In the case of a symmetric definite-positive matrix, the determinant is upper-bounded by the trace, which can be used as an alternative metric. In [Mihaylova, 2002; Yang, 2007; Morbidi, 2013], the choice of the optimal criterion for gradient descent is discussed. The proposed approaches are to minimize its determinant, its trace, or its largest eigenvalue. These are respectively referred to as *D-optimal*, *A-optimal* and *E-optimal* [Kiefer, 1974]. D-optimal design aim at reducing the generalized covariance, i.e. the volume of the uncertainty ellipsoid (the determinant, or log-determinant, of $\mathbf{P}$), while A-optimal design reduces the average covariance (the trace of $\mathbf{P}$). E-optimality consists in minimizing the largest eigenvalue of $\mathbf{P}$, hence the largest principal axis of the uncertainty ellipsoid. Pieces of evidence that D-optimal criterion is more robust to undesirable behaviors are reported in [Atanasov, 2015b]. For instance, [Carrillo, 2015] shows circular motions in which A- and E-optimal criterion decrease while the overall uncertainty is not reduced. It uncovers the presence of local minima that these criteria are not able to capture. Other observations in the same direction are reported in [Shahidian, 2017]. On the other hand, in [Salaris, 2019], arguments toward the use of the E-optimal criterion are provided, which optimizes the worst-case performance.

The discussion on the proper optimal criteria is not settled. Under the assumption that the local minima cases are marginal, the A-optimal criterion still provides a meaningful trade-off, in particular due to its linearity and computation efficiency.

Other formulations of $\mathcal{I}$ are described in [Atanasov, 2015b], such as conditional entropy, mutual information or probability of error. More generally, probabilistic entropy-based information measure can be exploited [Grocholsky, 2002]. Accordingly, a dedicated measurement filtering strategy needs to be chosen, which constrains the choice of $\mathcal{I}$. Kalman filtering techniques are often employed [Atanasov, 2015b], which are able to aggregate the measurement Gaussian uncertainties into an information metric.

8.2.3 Active Information Acquisition with Aerial Robot

AIA problems are often tackled with fleets of robots, sometimes collaborating with ground robots [Schlotfeldt, 2018; Cai, 2021]. It allows to leverage the heterogeneity

in terms of motion capabilities. In particular, ground robots have a larger payload and can employ heavier, more precise, sensors. AR are however agile and have much larger motion capabilities, but are limited in the choice of onboard sensors by their limited payload. In many works on AIA, the focus is oriented toward multi-agent cooperation while exploiting simplistic motion models. Therefore, AR are often chosen for implementing simulations or experiments because of their large motion span, which allow a decent approximation through, e.g., double integrator systems. The AIA solving algorithms are thus used as global or local trajectory planners, while the actual control is cascaded. However, the arguments from Section 4.2 can be used to again motivate the use of nonlinear models for the AR motion. Few works account for accurate models of AR in the scope of active sensing [Tokekar, 2014; Chen, 2016].

However, in the vein of the results from [Atanasov, 2015b] regarding the recasting of AIA problems into deterministic OCP, some works are exploiting MPC in the scope of AIA. In [Papaioannou, 2019], a predictive formulation for search and tracking problems is proposed and approximated using genetic algorithms. A MPC-based strategy is introduced in [Patil, 2014] for trajectory planning leveraging Gaussian uncertainties. Interestingly, it proposes to work with limited sensing capabilities, which are considered in the MPC formulation.

Another interesting solution is proposed in [Liu, 2017], which uses the A-optimal criterion as the cost function of the MPC to generate optimal sensing motions. The authors make use of the Intermittent Kalman Filter (IKF) [Sinopoli, 2004; Yang, 2017] to represent the limited sensing capabilities of the robot. This filter uses a Boolean term to represent whether or not a measurement from a given sensor is provided to the filter. If no measurement is provided, an update step is still performed with a virtual measurement of arbitrarily large (or infinite in the limit case) covariance. This idea is used to represent whether or not the feature is detected by the robot, e.g. if it falls into its FoV. This allows the MPC to propagate this knowledge through the horizon as the relative feature-robot pose evolves, and thus reverberates it the in the filter covariance, which trace is minimized. Because of the nonlinear closed-form expression of this Boolean term, the authors resorted to several relaxations of the problem. This work is focused on a single ground robot.

A preliminary tentative is made to combine the approaches from [Liu, 2017] and [Schlotfeldt, 2018] in [Singh Bal, 2019]. It leverages the heterogeneity of the aerial and ground robots in a centralized controller, aiming at reducing the feature estimation uncertainty of an IKF.

However, all these works are restricted to the use of linear MPC. To the best of our knowledge, N-MPC has never been employed for similar problems, while it provides large benefits compared to the approach from, e.g., [Liu, 2017], which relies on several relaxations, and does not account for accurate relative pose uncertainties.

In this chapter, we will build upon the work introduced in [Liu, 2017] and combine it with recent state of the art perception-aware N-MPC policies. We will make use of the formalism introduced in this thesis in order to build a N-MPC framework able to perform AIA in the scope of mobile feature tracking.

8.3 Intermittent Observation Modeling

As previously described, the MPC in [Liu, 2017] is used in conjunction with a specific type of Kalman filter, called the IKF. It models the assessment that a measurement comes into the filter. In the single agent case, this approach is would be redundant with the visibility constraints expressed in Chapter 5. However, the IKF is of major use in the multi-agent cases, since it allows to mathematically assess which agent will impact the overall estimation by providing a measurement.

In this section, we propose to recall the underlying concepts on the IKF. Then, the modeling of the measurement covariance matrix is detailed. The leveraging of IKF in the N-MPC framework will be discussed in the upcoming section.

8.3.1 Intermittent Kalman Filter

The so-called IKF is a Kalman filter introduced in [Sinopoli, 2004]. It aims at accounting for information loss over the network, when the sensor sends data remotely to the estimator. The information loss is modeled as a binary random process, whose probability is integrated into the filter update step. In practice, the absence of measurement is handled by feeding the filter with a "dummy" measurement, of arbitrary large covariance σ. Considering the limit case $\sigma \to \infty$, this is equivalent to neglecting the measurement in the update step. In this limit case, the resulting Kalman filter update equations are modified by pre-multiplying the innovation term by an *observation function* λ, which in fact turns out to be the probability distribution of the binary random process describing the information loss.

In [Yang, 2017], this filter is extended to a multi-sensor framework. Therein, the intermittent aspect is exploited to discriminate sensors that successfully transmitted newly captured data, from those which suffered information loss. The filter equations consider the same limit case $\sigma \to \infty$, disregarding the faulty measurements. In order to perform a uniform update, the aggregated measurements are stacked in a common vector, while the uncertainty matrices are stacked in blocks. Accordingly, the various observation functions are stacked to pre-multiply the innovation part of the innovation term.

The filter has been introduced as a discrete Kalman filter. However, these equations need to be integrated into the N-MPC in order to propagate the overall uncertainty through the receding horizon. Hence, we propose to reformulate those in the frame of a time-continuous Kalman-Bucy filter. We consider the same linear Gaussian motion and observation models as presented in Section 3.7 for the feature M:

$$\dot{\mathbf{x}}_M = \mathbf{A}\mathbf{x}_M + \boldsymbol{\eta}_{\mathbf{Q}}, \quad \boldsymbol{\eta}_{\mathbf{Q}} \sim \mathcal{N}(0, \mathbf{Q}), \tag{8.2}$$

$$\mathbf{z} = \mathbf{H}\mathbf{x}_M + \boldsymbol{\eta}_{\mathbf{R}}, \quad \boldsymbol{\eta}_{\mathbf{R}} \sim \mathcal{N}(0, \mathbf{R}), \tag{8.3}$$

where $\mathbf{A}$ and $\mathbf{H}$ are respectively the transition and observation matrices, and $\mathbf{Q}$ and $\mathbf{R}$ are the process and measurement noise matrices. Thus, the corresponding differential equations are

$$\dot{\hat{\mathbf{x}}}_M = \mathbf{A}\hat{\mathbf{x}}_M + \lambda\mathbf{K}(\mathbf{z} - \mathbf{H}\hat{\mathbf{x}}_M), \tag{8.4}$$

$$\dot{\mathbf{P}} = \mathbf{A}\mathbf{P} + \mathbf{P}\mathbf{A}^\top + \mathbf{Q} - \lambda\mathbf{K}\mathbf{R}\mathbf{K}^\top, \tag{8.5}$$

where $\hat{\mathbf{x}}_M$ is the estimate of $\mathbf{x}_M$, $\mathbf{P}$ is the estimate covariance matrix, and $\mathbf{z}$, $\mathbf{R}$ are the concatenated measurement vectors and covariance matrices. The Kalman gain is given by $\mathbf{K} = \mathbf{P}\mathbf{H}^\top\mathbf{R}^{-1}$. The observation function λ is described hereafter.

Both in [Sinopoli, 2004] and [Yang, 2017], a stability analysis of the filter is conducted, demonstrating its applicability.

8.3.2 Intermittent Observations

The fact that the sensor provides a measurement of M, i.e. that the M falls into its FoV, is described using an observation function λ defined as

$$\lambda = \begin{cases} 1, & \mathbf{x}_M \text{ is inside the FoV} \\ 0, & \mathbf{x}_M \text{ is outside the FoV} \end{cases} . \tag{8.6}$$

It depends on the relative pose between the GTMR and the feature, hence can discriminate whether a predicted configuration would yield a measurement [Liu, 2017].

In order to avoid the discontinuity induced by λ, the same approach as in Section 6.4.1 is employed to approximate λ as a sigmoid function. More specifically, a product of two sigmoids can be used in order to add describe both the FoV angular and range limits. However, as discussed in the upcoming section, the influence of the range on the uncertainty is modeled in the measurement uncertainty, driving the AR to maintain a reasonable range w.r.t. the feature. Hence, the second sigmoid associated with the range can be omitted. The approximated expression of λ is then given by

$$\lambda = \frac{1}{1 + e^{-\lambda(\mathrm{c}\alpha - \mathrm{c}\beta)}}, \tag{8.7}$$

using the notations $\mathrm{c}\beta$ and $\mathrm{c}\alpha$ from Section 5.3 and Section 5.4. Equation (8.7) describes the observation function for a conic FoV. A similar formulation with a pyramidal FoV could be written using Equation (5.11) or (5.13).

8.3.3 Observation Uncertainty Model

The observation uncertainty model described in Section 3.6 is nonlinear and computationally heavy, and thus is tough to integrate efficiently in the N-MPC. In particular, the algorithmic computation of the Jacobian and its inversion implies a large amount of computation. Moreover, this model is specific to the use of fiducial markers and cannot be generalized to any object detection process. For these reasons, it is more convenient to use a simplified, more generic model of the observation uncertainty. For a range and bearing sensor, following the linear Gaussian sensing model (8.3), the measurement covariance matrix $\mathbf{R}$ can be pictured as a 3D ellipsoid, centered on the measurement mean. Its eigenvalue σ_z along the bearing vector (i.e., associated with the range) is different from the eigenvalues σ_{xy} along the two orthogonal directions (associated with the bearing), as suggested in [Chung, 2004; Yang, 2007; Morbidi, 2013]. For cameras (RGBD, stereo or monocular), σ_z is typically larger than σ_{xy}, since the depth information either comes from image processing that adds

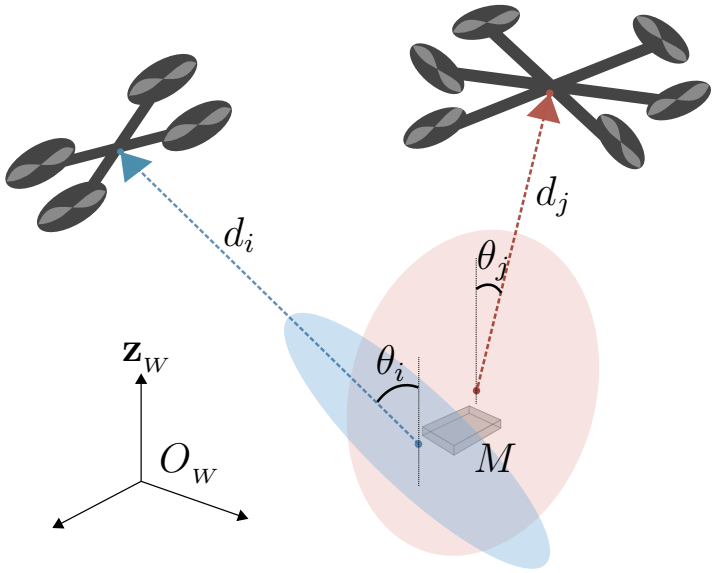


Figure 8.1: Two heterogeneous ARs, numbered i and j, equipped with sensors, observing a static feature M. The measurement uncertainties are represented by the two colored ellipsoids, centered on their respective measurement means. The dashed lines correspond to the two bearing vectors. d and d_j are the respective distances toward the feature.

up the noise from several pixel measurements (typically stereo alignment or PnP), or from an imprecise infrared measurement.

Thereby, the measurement uncertainty matrices $\mathbf{R}$ can be written using a SVD [Beder, 2006; Yang, 2007], whose transformation matrix $\mathbf{T}_\theta$ is the 3D rotation matrix about the angle θ between $\mathbf{z}_W$ and the bearing vector (see Figure 8.1).

Additionally, the distance toward the feature influences the uncertainty. The further the feature will be from the sensor, the less precise will be the measurement. For instance, for cameras, the angular arc covered by a pixel increases with the distance, thus the measurement uncertainty grows for a constant pixel noise as the distance increases. On the contrary, the closer the feature is from the sensor, the more difficult can be the detection, with RGBD sensors for instance. It is commonly assumed, e.g. in [Chung, 2004; Morbidi, 2013], that the observation is optimal at a given distance d_{ref} from the feature, around which the uncertainty evolves in a parabola. This so-called "sweet spot" is often to be guessed manually, given e.g. the size of the feature or the sensor FoV. We propose to scale the eigenvalues σ_{xy} and σ_z according to the distance to this sweet spot d_{ref}.

Finally, the measurement covariance matrix is written

$$\mathbf{R} = \mu(d - d_{ref})^2 \mathbf{T}_\theta \begin{bmatrix} \sigma_{xy} & 0 & 0 \\ 0 & \sigma_{xy} & 0 \\ 0 & 0 & \sigma_z \end{bmatrix} \mathbf{T}_\theta^\top, \tag{8.8}$$

where d is the distance from sensor to feature, μ is a scalar scaling factor that defines the acceptable sensing range around d_{ref}.

Remark. We remark that this assumption is meaningful for the linear part in the case of fiducial markers. The orientation uncertainty cannot be realistically captured using this approximation, because of the local minimum arising when the camera

and marker are frontoparallel. However, contrary to Chapter 6, the orientation of the feature is not considered, since it is assumed punctual in the model from Section 3.5.

As stated in Section 3.5, the measurements are expressed in $\mathcal{F}_W$, and so is the feature motion model. Hence, the impact of the uncertainty of the associated frame transformation on the measurement uncertainty $\mathbf{R}$ is left out of the scope of this thesis. To account for this, the covariance should be propagated from $\mathcal{F}_S$ to $\mathcal{F}_W$ using the GTMR state covariance. In practice, it would require to have an uncertainty model expressed in terms of the GTMR state and inputs, hence to integrate knowledge on this uncertainty in the N-MPC state vector. Since the most reasonable assumption is to consider the current state uncertainty as constant over the full receding horizon, its impact on the minimization process is marginal and can be neglected.

8.4 Leverage Uncertainty Minimization in N-MPC

8.4.1 N-MPC with the Kalman Filter state

The aforementioned observation model allows to compute at each instant the measurement uncertainty associated with a measurement that the sensor would yield in this particular robot-feature relative pose. Moreover, this computation can be achieved in closed-form in terms of state and parameters of the N-MPC. Therefore, this formulation allows to predict virtual measurements, over the receding horizon, and thus to predict how the GTMR motion will affect the object observation. This allows the controller to leverage the position and orientation of the sensor for improving the estimation.

In order to account for the estimation uncertainty over the receding horizon, its computation must be possible at each shooting point. But while the $\mathbf{R}$ matrix can be computed through Equation (8.8), the computation of $\mathbf{P}$ require knowledge of the current filter state and covariance. These quantities are directly available for the current time instant, but are not for subsequent shooting points, since the virtual measurements previously computed would have affected the filter. To alleviate this problem, the state vector of the N-MPC is extended with the Kalman filter state $\hat{\mathbf{x}}_M$ as well as a minimal vector representation of the (symmetric) estimation covariance matrix $\mathbf{P}$. This minimal representation, $\mathbf{P}_\triangle$, is composed of the lower triangular part of $\mathbf{P}$:

$$\mathbf{P}_\triangle = \left[p_{ij}\right], \quad 1 \leq i \leq j \leq n_M, \tag{8.9}$$

where n_M is the cardinal of $\mathbf{x}_M$. The N-MPC state vector of each sensing agent is therefore written as

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_{\text{GTMR}} \\ \hat{\mathbf{x}}_M \\ \mathbf{P}_\triangle \end{bmatrix}, \tag{8.10}$$

and its time derivative $\dot{\mathbf{x}}$ is obtained from Equations (3.21), (8.4) and (8.5). This approach allows to propagate the estimation covariance through the receding horizon, accounting for the virtual measurements, predicted at each shooting point. In the following, this representation of the Kalman filter will be referred to as the *internal* filter.

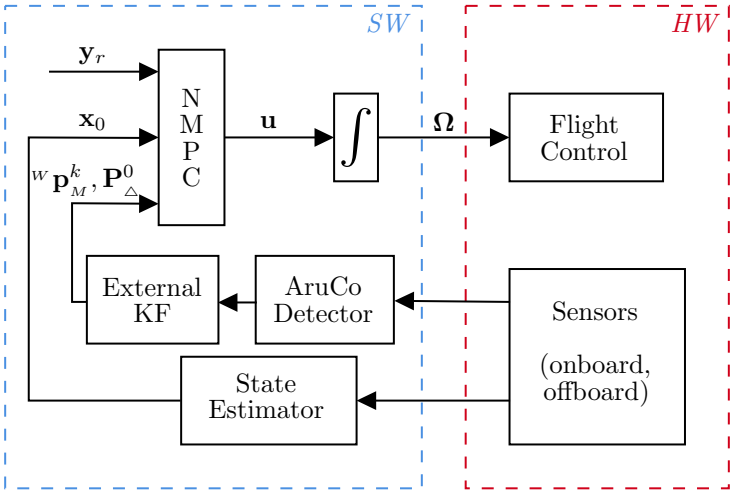


Figure 8.2: Block diagram of the N-MPC-AIA framework. The N-MPC state in made of $\mathbf{x}_0$, ${}^{W}\mathbf{p}^{0}_{M}$ and $\mathbf{P}^{0}_{\triangle}$, while ${}^{W}\mathbf{p}^{k}_{M}$ are provided by the external KF for each shooting node to compute the virtual measurements over the horizon.

We remark that this internal filter does not perform the actual feature state estimation. It is required to maintain the use of an *external* filter, which performs the estimation and closes the loop by providing the initial value for $\mathbf{x}_M$ and $\mathbf{P}_{\triangle}$ to the N-MPC at each optimization cycle.

In order to reduce the amount of extra state variables in the N-MPC, we use a constant position model, which yields a minimal Kalman filter state size of $n_M = 3$. This also implies that $\mathbf{A} = \mathbf{0}_3$ and $\mathbf{H} = \mathbf{I}_3$, which significantly simplifies the Riccati equation (8.5). To further reduce the state, improving the performances at the cost of precision, the cross-variance terms p_{xy}, p_{xz} and p_{yz} could be also neglected, since the 3D coordinates of M can be considered independent random processes.

Finally, to overcome the imprecision induced by this simplistic motion model, the predicted feature pose for each sampling point of the horizon can be computed using the external filter, which relies on a more realistic motion model (e.g. constant acceleration, as discussed in Section 3.7). These poses are passed to the N-MPC as parameters, used to compute the virtual measurements and the associated matrices. Hence, coupled with a high process noise $\mathbf{Q}$ in the internal Kalman filter, this allows an accurate internal representation of the feature motion.

Echoing to the block diagrams introduced in previous chapters, the proposed framework is depicted in Figure 8.2.

8.4.2 AIA with Multiple Sensing Agents

As mentioned in Section 8.2, such problems are mostly tackled with fleets of ARs, thus we formalize the NLP accordingly. However, recalling the discussion in Section 7.2, a decentralized implementation is preferred. Thus, rather than concatenating the states of each agent $\mathbf{x}_i$ into the full system state, only the measurement vectors $\mathbf{z}_i$, the uncertainty matrices $\mathbf{R}_i$, and the observation terms λ_i are exchanged. Hence, in a decentralized formulation, each agent is controlled by its own N-MPC, estimates the

feature position using a Kalman filter, and communicates its predicted measurements to the rest of the team. The corresponding framework is illustrated in the block diagram Figure 8.3. Then, each N-MPC uses the concatenated measurements as external parameters to update their internal Kalman filter, using Equations (8.4) and (8.5). This allows, as we demonstrate in Section 8.5, to account directly for all the observations at the control-level. Additionally, these extra measurements are exploited in the external filter, to improve the feature state estimation. This assumes a perfect instantaneous communication, which is far from being true in real scenarios. In case of failed or delayed communication, the ARs would still be able to rely on the previous measurements, which is a correct approximation when dealing with relatively slow motions, but needs to be considered when dealing with agile maneuvers.

Remark. A centralized version can of course be proposed, in which the controller has full knowledge of the system and computes the control inputs of each agent accordingly. The increasing size of the system state severely limits the scalability, but this approach has been successfully tested for a system of 2 robots.

8.4.3 NonLinear Programming

The NLP is formalized using the paradigm from Section 4.4.3. The observation task consists in minimizing the cumulative sum of estimation uncertainties over the receding horizon. The associated cost function $C^{\text{perception}}$ is simply written

$$C^{\text{perception}} = w_{\text{p}} \text{tr}(\mathbf{P}), \tag{8.11}$$

where w_{p} the tunable controller weight.

The maximization of $\text{c}\beta$ is no longer required, since given the perception model from Section 8.3.3, the optimal sensing configuration yield by the minimization of $\text{tr}(\mathbf{P})$ implies maintaining the feature aligned with the sensor principal axis. This makes the $\text{c}\beta$ objective from Section 5.3 redundant, and thus superfluous.

As mentioned in the introduction paragraph of Section 8.3, the use of IKF allows to assess the fact that the onboard sensor will provide an actual measurement, through the λ variable. The first noticeable effect is that the observation function λ acts as a soft FoV constraint. Indeed, M getting out of the FoV implies that λ tends toward 0, hence that the predicted virtual measurements computed over the horizon will have no effect on the overall uncertainty $\mathbf{P}$. The N-MPC will consequently avoid these configurations, regardless of the number of agents n_{A}, all the more so since the internal process noise $\mathbf{Q}$ is large. The constraint can thereby be omitted to simplify the NLP.

But the main advantage of IKF concerns the multi-robot collaboration. It allows each agent to assess whether the others are providing effective measurements. This is directly impacted onto the overall estimation. A practical instance of ensuing collaborative emergent behavior is reported in Section 8.5.2

The discrete-time NLP for each sensing agent, over the receding horizon T, sampled in N shooting points, at a given instant t, is expressed as

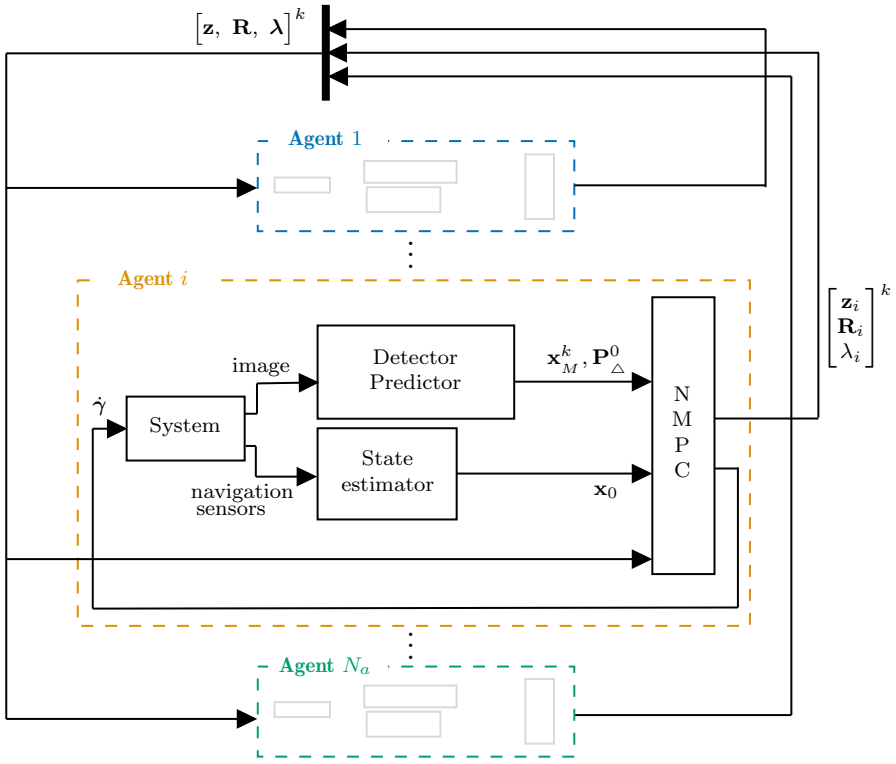


Figure 8.3: Block diagram of the decentralized AIA framework for MRS. The controller of the *i-th* agent is briefly detailed, echoing to Figure 8.2. All the measurements $\mathbf{z}$, covariances $\mathbf{R}$ and observation functions λ, predicted over the receding horizon are communicated to the whole system. The right topscript recalls which variables are provided for all the k shooting nodes of the N-MPC, and which are only provided for $k = 0$.

$$\min_{\substack{\mathbf{x}_0 \dots \mathbf{x}_N \\ \mathbf{u}_0 \dots \mathbf{u}_{N-1}}} \quad \sum_{k=0}^{N} \|\mathbf{y}_k - \mathbf{y}_{r,k}\|^2_{\mathbf{W}_\mathrm{m}} + \sum_{k=0}^{N} w_\mathrm{p} \mathrm{tr}(\mathbf{P}_k) + \sum_{k=0}^{N-1} \|\mathbf{u}_k\|^2_{\mathbf{W}_\mathrm{i}} \tag{8.12a}$$

$$\begin{aligned}
s.t. \quad & \mathbf{x}_0 = \mathbf{x}(t) && && (8.12b)\\
& \mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k, {}^W\mathbf{p}_M^k), && k\in\{0,\ N-1\} && (8.12c)\\
& \mathbf{y}_k = \mathbf{h}(\mathbf{x}_k), && k\in\{0,\ N\} && (8.12d)\\
& \mathbf{P}_k = \mathbf{g}(\mathbf{x}_k), && k\in\{0,\ N-1\} && (8.12e)\\
& \underline{\gamma} \leq \gamma_k \leq \overline{\gamma}, && k\in\{0,\ N\} && (8.12f)\\
& \underline{\dot{\gamma}}(\gamma_k) \leq \mathbf{u}_k \leq \overline{\dot{\gamma}}(\gamma_k), && k\in\{0,\ N-1\} && (8.12g)\\
& \underline{\boldsymbol{\mu}_k} \ \leq \ \boldsymbol{\mu}(\mathbf{x}_k, \mathbf{p}_k) \ \leq \ \overline{\boldsymbol{\mu}_k}, && k\in\{0,\ N\} && (8.12h)
\end{aligned}$$

where $\mathbf{g}$ denotes the function that reconstructs the $\mathbf{P}$ matrix from of $\mathbf{P}_\triangle$ in $\mathbf{x}$.

Remark. The proposed approach is to consider an additive cost function for each stage along the horizon, while a terminal-stage-only cost could be defined. Both functions can be effective and depend on the applications. When handling agile maneuvering, it might not be desirable to have a growing uncertainty for each stage, so the additive cost might not be suitable. On the other hand, for coverage maximization, this

Variable	Simulation weights		Experiment weights
	Quadrotor	Hexarotor	
$\mathbf{p}$	0	1	50
$\mathbf{q}$	10^{-1}	50	80
$\mathbf{v}$	5.10^{-1}	2.10^{-1}	1
$\boldsymbol{\omega}$	5.10^{-1}	10	1
$\mathbf{a}$	10^{-4}	10^{-2}	10^{-4}
$\dot{\boldsymbol{\omega}}$	10^{-4}	10^{-2}	10^{-4}
$\mathbf{u}$	0	0	0
$\mathrm{tr}(\mathbf{P})$	80	0	50

Table 8.1: Table of N-MPC weights for simulations and experiments in Section 8.5.

additive cost should be preferred, since it allows a smoother growth of the cost. In the following, an additive formulation is proposed.

8.5 Experimental and Simulation Validation

In the following experiments and simulations, the GTMR is equipped with a down-facing camera, as pictured in Figure A.2b. The ARs are controlled using the proposed decentralized N-MPCs. Once the ARs are in position, the observation task is enabled (by changing the weights in Equation (8.12a)) for each agent simultaneously. At the same time, the weights on the position task are set to zero, and small weights are applied to maintain constant roll/pitch and small velocities, in order to penalize large motions and ensure the stability of the system. Therefore, the motions observed are driven only by the observation task in the cost function. When using multiple agents, the robots exchange data directly through wifi. The communication frequency is 10 Hz. The controller weights used in the presented simulations and experiments are reported in Table 8.1.

8.5.1 Static Feature Sensing

This section presents the behavior of a system of $n_{\mathrm{A}} = 1$, 2 or 3 quadrotor(s), assigned to reduce the estimation uncertainty of a static feature. Various behaviors emerge when more sensing capabilities are added to the system and are reported hereafter. The cases $n_{\mathrm{A}} = 1$ and $n_{\mathrm{A}} = 2$ are performed on real quadrotors, while the case $n_{\mathrm{A}} = 3$ is performed in simulation. Those are reported in Video 8.1.

The (x, y) positions of the agents as well as the value of the estimation uncertainty $\mathrm{tr}(\mathbf{P})$ over time are reported in Figure 8.4. It illustrates how the system configuration is affected by the observation task to yield a lower uncertainty. We note that the value of $\mathrm{tr}(\mathbf{P})$ reported therein is computed using a Kalman filter external to the control framework from Figure 8.3, which aggregates the measurements from all the agents, and is thus decorrelated from the N-MPC processes.

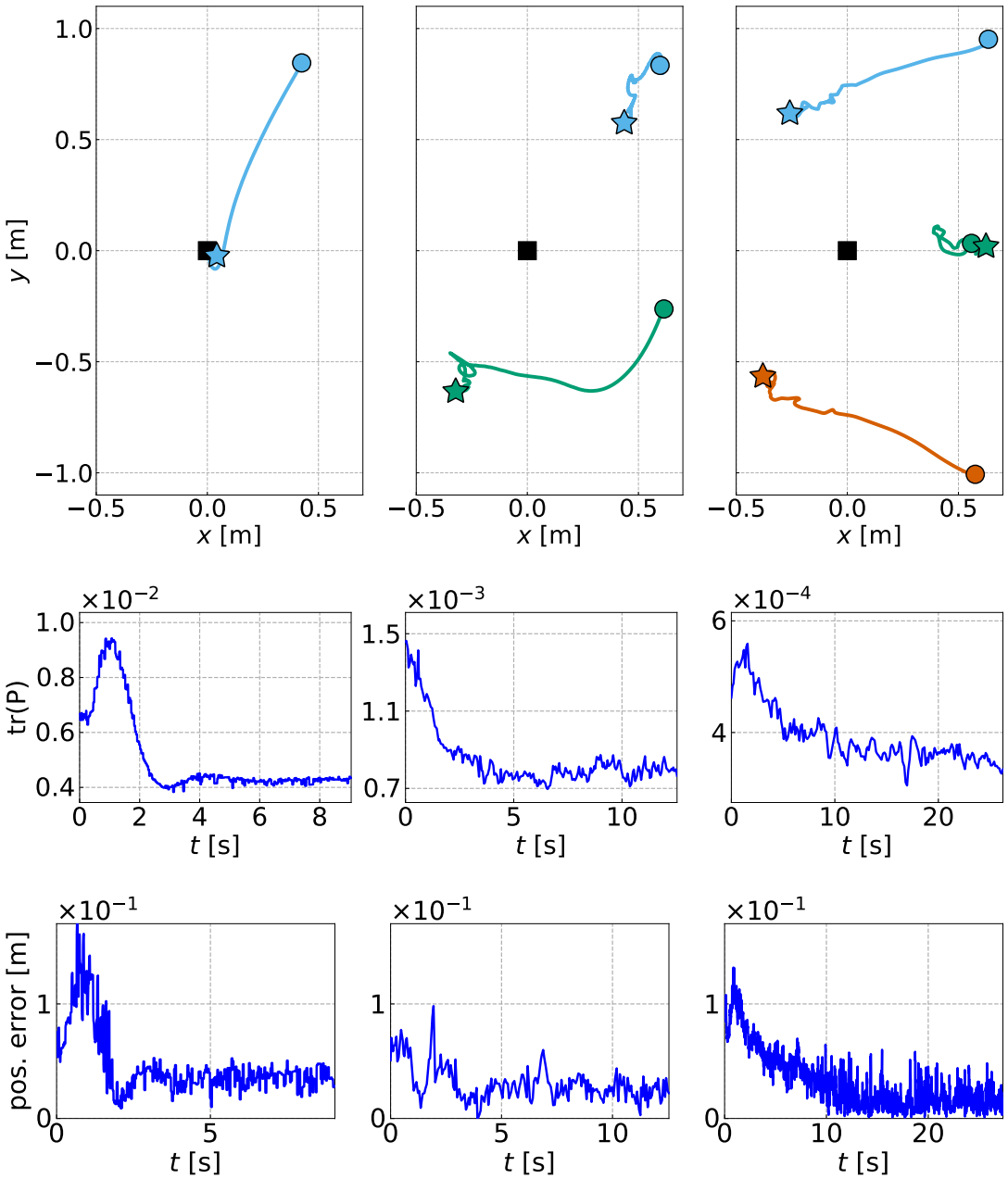
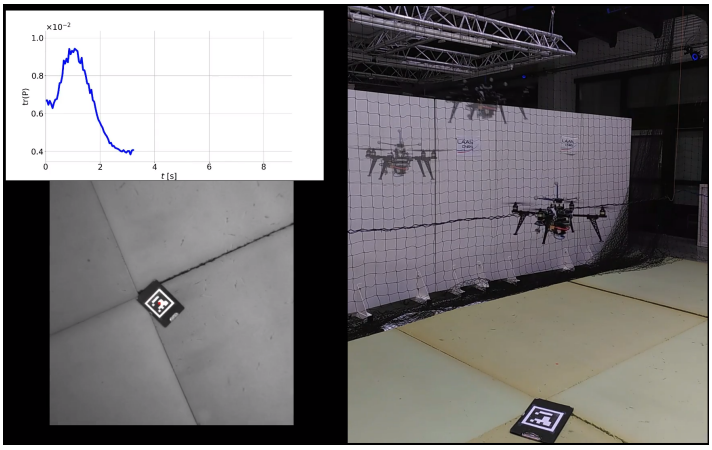


Figure 8.4: The (x, y) position over time (top), the evolution of tr($\mathbf{P}$) over time (middle) and the position reprojection error overtime; for the 1 agent (left), 2 agents (middle) and 3 agents (right) cases. The initial positions of the ARs (i.e., when the observation task is enabled) and the final positions are respectively denoted by the circles and stars. The feature position in $(0, 0)$ is denoted by the black square.



(a) Clickable image to the video on PeerTube.

(b) QR code to the video.

Video 8.1: Experiments and simulation with 1, 2, and 3 ARs in a collaborative observation task.

8.5.1.a Case $n_{\mathbf{A}} = 1$

In this scenario, minimizing $\mathrm{tr}(\mathbf{P})$ is equivalent to minimizing $\mathrm{tr}(\mathbf{R})$. Indeed, given the model presented in Section 8.3.3, $\mathrm{tr}(\mathbf{R})$ is minimal when the feature is aligned with the principal axis of the sensor, at a distance d_{ref}. Indeed, the GTMR driven by the N-MPC, whose motion is reported in the left column in Figure 8.4, converges to the (x, y) position of the feature. The AR motion is faster at the beginning since the cost gradient is steeper, as it can be seen in Video 8.1. The N-MPC also makes the quadrotor move upward, since tilting to move toward the marker while maintaining the initial height would induce a loss of visibility, thus a higher estimation covariance.

The bottom graph shows the evolution of $\mathrm{tr}(\mathbf{P})$ over time. The system is reducing this uncertainty w.r.t. its value at the initial position. We note that the estimation uncertainty first increases (since the GTMR tilts and moves upward) before converging to a smaller value, showing that the proposed algorithm is non greedy and is able to overcome the initial local minimum.

8.5.1.b Case $n_{\mathbf{A}} \geq 2$

When the system is made of more than one single agent, the aforementioned sensing configuration is no longer optimal. The system exploits the extra sensing capabilities to observe the feature more efficiently, by placing each agent on opposite sides w.r.t. the feature, hence improving the overall sensing, as shown in the second and third columns of Figure 8.4. In particular, in the 3 agent case, the achieved configuration is similar to the one reported in [Chung, 2004; Tallamraju, 2019], which was analytically computed. The agents are positioned around the feature, spaced by 120°. However, we remark that the convergence in that latter case is slower since the initial estimation is already good, and the benefit of moving to a different position is smaller.

	MPC driven config.			naive config.	
	1 agent	2 agents	3 agents	2 agents	3 agents
mean(tr(**P**))	5.04e-3	0.85e-3	0.39e-3	3.90e-3	0.72e-3
std(tr(**P**))	1.54e-3	0.16e-3	0.05e-3	0.81e-3	0.72e-3
mean(tr(**R**))	4.4e-3	6.8e-3	6.6e-3	4.0e-3	4.1e-3
std(tr(**R**))	6.7e-3	9.8e-3	9.8e-3	5.2e-3	7.2e-3

Table 8.2: Position covariances (measurement and estimation) mean and std using 1, 2 or 3 agents. The first three columns report the statistics gathered using the proposed framework, while the last two correspond to flights in a naive configuration, described in Section 8.5.1.c.

8.5.1.c Estimation improvement

We validate the pertinence of the proposed framework by effectively measuring the estimation improvement yielded by the sensing configuration. We report in Table 8.2 some quantitative statistics on the estimation (trace of **P**) and measurement uncertainties (trace of **R**) for the position of the feature, using 1, 2 and 3 ARs. These values were aggregated over several experiments similar to those reported in Figure 8.4, with various initial starting configurations. It covers a total flight time of about 4 minutes for each scenario. For a fair comparison, the statistics were all gathered in a simulated environment, such that the detection process in not altered by disparities in the camera quality. It allows in particular a meaningful evaluation of the 3 agent case which was not performed in real experiments. The values recorded in real experiments for the 1 and 2 agents cases are nevertheless similar to those reported in Table 8.2 for simulations. The last two columns report the statistics gathered with respectively 2 and 3 agents, manually driven right above the feature (referred hereafter as *naive* configuration, since it is the one observed with 1 agent case and naively extrapolated). As expected, adding additional measurements improves the estimation uncertainty (of about an order of magnitude), compared to the 1 agent case (first column). However, both with 2 and 3 agents, the N-MPC driven configurations yield much smaller uncertainties than the respective naive configuration,.

Indeed, in this naive configuration, the agents observe the feature from about the same angle. But, due to the shape of uncertainty ellipsoids (see Figure 8.1), it is advantageous to place the cameras from aside, observing from different angles. It allows to compensate the poor range estimation of both sensors by exploiting the better bearing estimations (or vice-versa if $\sigma_{xy} > \sigma_z$) [Beder, 2006]. We note that this estimation improvement is achieved by degrading the individual measurement, demonstrating an emergent collaborative behavior, despite the decentralization.

8.5.2 Asymmetric Sensing Team

This section presents a simulation using an asymmetric heterogeneous sensing team. The two agents are one quadrotor and one tilted-propeller hexarotor, showing the capability of the framework to handle various types of GTMR using different dynamics. In addition, the tasks assigned to the agents are asymmetrical:

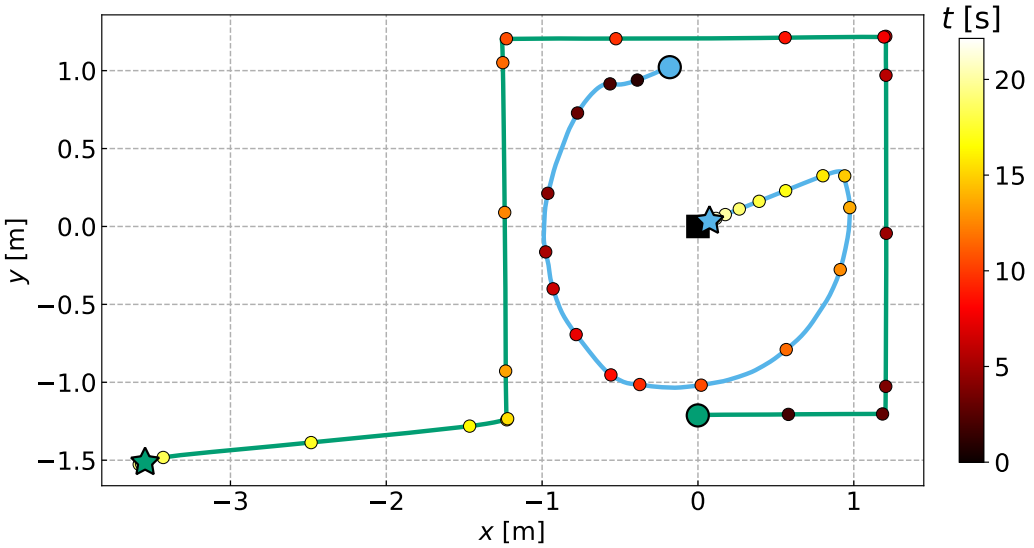
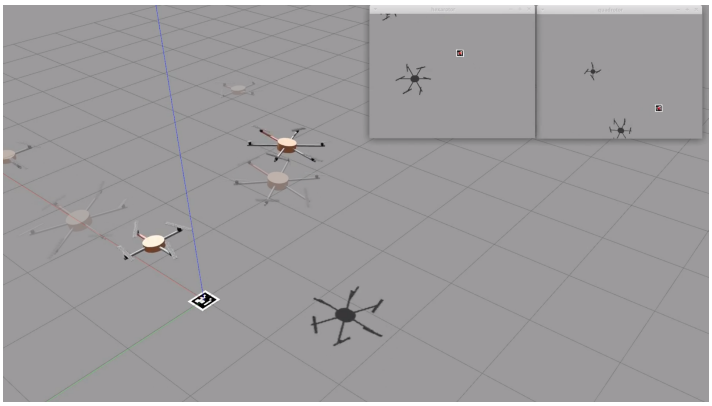


Figure 8.5: The (x, y) position of both GTMR (blue for hexarotor, green for quadrotor). The colored dots represent their respective position each second, while the color gradient indicates the time at which the positions were recorded. The circles and stars are respectively the initial and final positions of the ARs, while the black square is the feature position.

1. the hexarotor is tasked as in Section 8.5.1, i.e. has a minimal motion task ensuring the system stability, and is tasked to improve the overall estimation;
2. the quadrotor has a given motion task to achieve (i.e. reach a set of consecutive waypoints), but no observation task ($w_{\mathrm{p}} = 0$). The quadrotor however participates to the observation, and shares its measurements with the system.

The (x, y) configuration of the system over time is reported in Figure 8.5. When the quadrotor moves to reach a waypoint, the hexarotor positions itself on the opposite side, achieving a configuration similar to the 2 agent case presented in Section 8.5.1.b, and thus improving the overall estimation. Since the quantities $\mathbf{z}$, $\mathbf{R}$ and λ are predicted over the receding horizon, the reactivity of the hexarotor for the system reorganization is improved. It can be observed in Video 8.2 that the hexarotor motion starts as soon as the quadrotor is commanded to reach its next



(a) Clickable image to the video on PeerTube.

(b) QR code to the video.

Video 8.2: Heterogeneous system simulation: two different ARs performing two different tasks simultaneously.

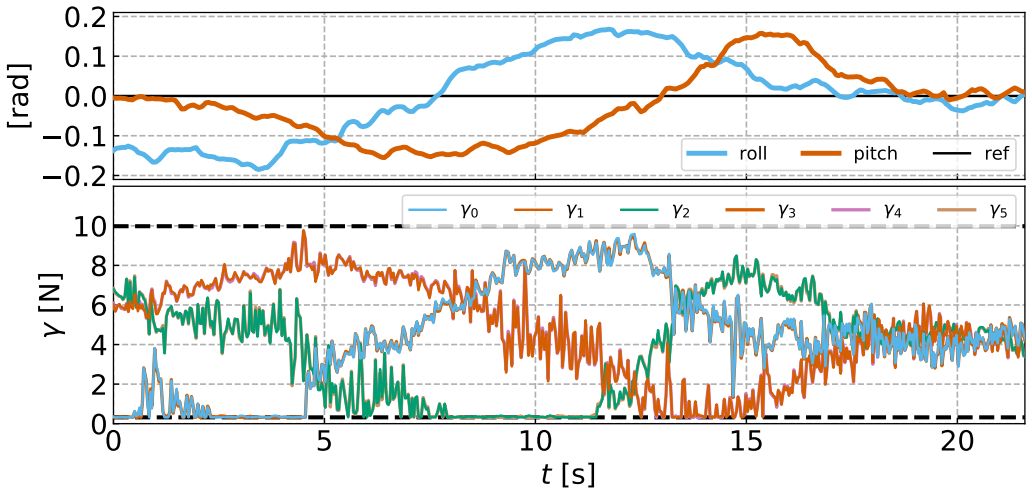


Figure 8.6: Above, the roll and pitch of the tilted-propeller hexarotor over time, with their reference value (0 rad). Below, the thrusts exerted by the 6 propellers, with the lower and upper bounds marked by the black dashed lines.

waypoint. However, as the hexarotor gets closer to the optimal sensing configuration, it slows down, since the cost gradient gets flatter.

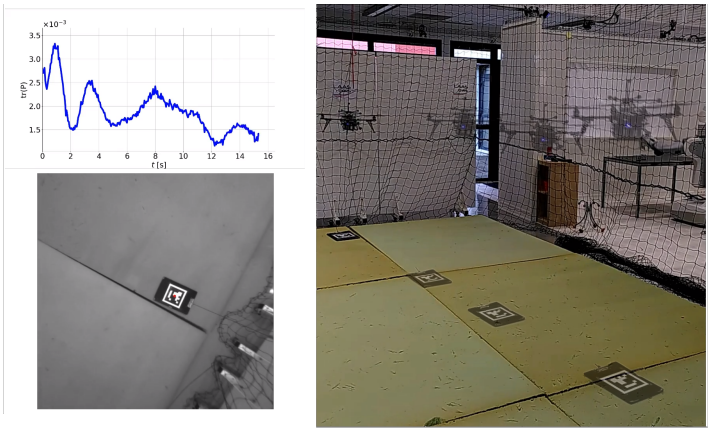
When the quadrotor goes toward its final waypoint (after 15 s), it loses sight of the feature. The hexarotor then handles the observation burden alone and moves as in the 1 UAV case, i.e. goes on top of the feature, since this configuration minimizes both the measurement uncertainty and the attitude error, as depicted in Section 8.5.1.a. This is an effect of the observation function λ, which, when becoming 0, indicates that the agent has no influence on the overall estimation, allowing the system to reconfigure itself accordingly.

The roll and pitch of the hexarotor are reported in Figure 8.6 (top). The hexarotor stays stable while tilted, hence reducing its measurement uncertainty, while maintaining a configuration that minimizes the overall estimation with the quadrotor. Figure 8.6 also reports the propeller thrusts $\boldsymbol{\gamma}$ (bottom), which actually reach their bounds during this motion, demonstrating that the N-MPC exploits the full actuation span of the GTMR.

8.5.3 Mobile Feature Tracking

In order to demonstrate the applicability of the controller as well as its capability to track a moving feature, a real experiment have been performed using a single quadrotor and is reported in Video 8.3. The setup is similar to the one presented in 8.5.1, i.e. the quadrotor is tasked to reduce the observation uncertainty, while a minimal motion task is defined to ensure its stability. However, the marker is moving in straight line, at around 0.3 $\mathrm{m.s^{-1}}$.

The path followed by the feature and the sensing agent is reported in Figure 8.7. The quadrotor is placed 1 m away from the feature, and the observation task is enabled, causing it to move. The marker starts moving, and the AR catches up with its motion. As opposed to the results displayed in Chapter 5, this tracking is achieved without any exogenous position reference.



(a) Clickable image to the video on PeerTube.

(b) QR code to the video.

Video 8.3: Two successive experiments: tracking of a moving marker with 1 and 2 ARs.

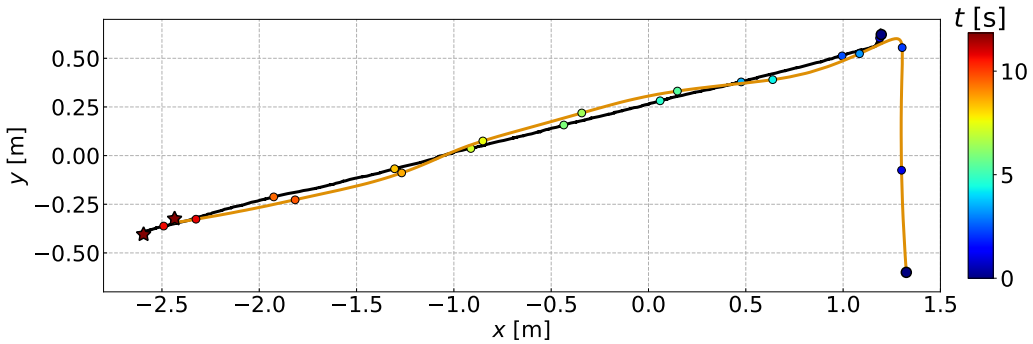
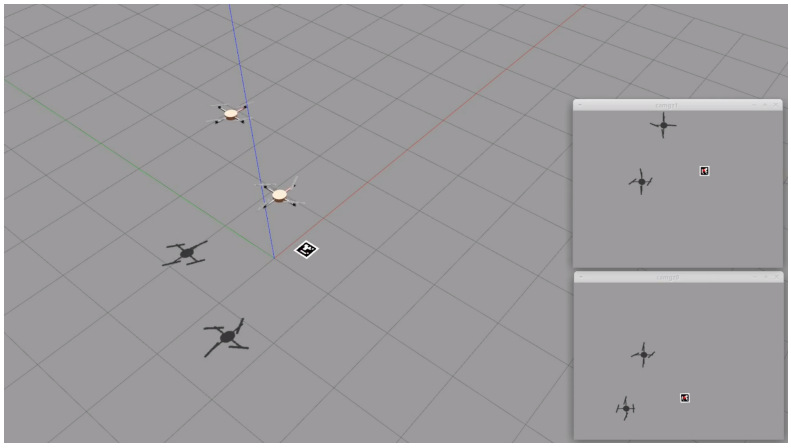


Figure 8.7: The (x, y) path of the marker (black) and AR (orange) along time. The initial position for both are denoted with the black and orange circles, respectively. The colored dots represent their respective position each second, while the color gradient indicates the time.



(a) Clickable image to the video on PeerTube.

(b) QR code to the video.

Video 8.4: Additional simulations: tracking of a moving feature with 1 and 2 ARs.

Additional simulations performed with a moving feature and quadrotors can be found in Video 8.4

8.6 Conclusion

8.6.1 Synthesis and Discussion w.r.t. State of the Art

The proposed N-MPC framework is shown to be able to solve AIA problems for a GTMR, equipped with an onboard sensor. It leverages the mathematical formulation of the AR-feature relative pose for the computation of the measurement uncertainty matrix $\mathbf{R}$ and the observation function λ, which are in turn used to compute the observation uncertainty matrix $\mathbf{P}$. This allows, in particular, to produce optimal sensing trajectories without any exogenous position reference given to the GTMR.

The framework is also scalable to multiple heterogeneous agents, yielding emergent cooperative behaviors. It has been tested in experiments and simulations, both in homogeneous and heterogeneous tasks.

Compared to other non-MPC approaches to AIA [Atanasov, 2014b; Morbidi, 2013; Dames, 2017], our approach couples the AR low-level control with the observation task, in the vein of Chapter 4. Nonetheless, considering the active observation as an AIA problem allows more versatility than the perception-aware N-MPC previously introduced, since the uncertainty minimization can account for external measurements of the feature. In particular, it allows the decentralization of the observation as a shared objective rather than constraining each individual agent. More generally, any additional measurement source can be considered, such as static cameras that would provide additional measurements to the N-MPC. This is of particular interest in applications where ARs can be exploited for video coverage in conjunction with manual or crane-driven cameras, e.g. in sports events [Zemas, 2017].

There exists, however, a corpus of works tackling such problems with linear MPC [Atanasov, 2014b; Liu, 2017; Tallamraju, 2019]. But considering linear models does not come without drawbacks, as discussed in Chapter 2. The fact that, using N-MPC, a quadrotor is commanded to move upward to maintain visibility while moving toward a feature, as in the attached video, is a consequence of considering its complete nonlinear dynamics. Additionally, there is a nonlinear coupling between the agent and feature states in both the measurement uncertainty matrix $\mathbf{R}$ and the observation function λ. The propagation of these quantities over the horizon is not possible without simplifications. In [Liu, 2017], this problem is solved by considering an arbitrary constant value for $\mathbf{R}$; and relaxing Equation (8.7) by precomputing the nonlinear orientation terms in a first step. These two simplifications have some impact on the accuracy of the propagation. Conversely, the main improvement of our approach is that considering the nonlinear coupling in $\mathbf{R}$ makes the N-MPC able to not only maintain the visibility, but also effectively reduce the observation uncertainty. In particular, the collaborative emerging behaviors observed in Section 8.5.1 are a consequence of considering the nonlinear observation model.

Finally, other works [Chung, 2004; Tallamraju, 2019] have an 'offline' approach to the AIA problem. Therein, the controller maintains a precomputed formation that is proven to yield optimal measurements. Such an approach is more rigid, considers a fixed number of agents and does not allow the system to reconfigure according to additional tasks. We rather propose an approach where finding the compromise between minimizing the uncertainty and accounting for a separate motion task is

devoted to the N-MPC, according to a set of weights and parameters (which can be changed online by a supervisor to prioritize one or the other).

8.6.2 Scalability w.r.t. the Number of Features

The main limitation of this framework is the poor scalability w.r.t. the number of features to track. Adding extra features to be tracked in the framework would imply the extension of the state, as well as the addition of more nonlinear equations in the NLP. As a result, a sub-optimal solving policy, such as the RTI used in our implementation, might not be able, at a certain point, to produce a valid solution. To overcome this problem, a more suitable filter should be employed. In [Schlotfeldt, 2018], a Kalman filter [Atanasov, 2014a] is used for multiple feature tracking. Alternatively, following the literature, the PHD filter might be considered for its advantageous scalability [Dames, 2020], in larger scale AIA applications. However, an uncertainty metric has to be defined and expressed in closed-form as a function of the feature and agent poses, similar to Equation (8.5). Moreover, a solution to the state extension issue should be produced, in order to include in the N-MPC a minimal state representation which allows to propagate the uncertainty.

Part III

Conclusions, Perspectives and Appendices

Chapter 9

Conclusion

Contents

9.1 Synthesis of Contributions **140**
9.2 Overall Conclusion **141**
9.3 Perspectives **142**
9.4 Final Thoughts **143**

This conclusive chapter first provides a synthesis of the previous chapters and the contributions therein, followed by a general conclusion on the applicability and limitations of such frameworks. Then, the research perspectives for future works are exposed, before a concluding word from a personal point of view.

9.1 Synthesis of Contributions

Throughout this thesis, we addressed the conception of a perception-aware optimal control framework for generic multi-rotor aerial vehicles, devoted to perform tasks whose objectives are separable between motion-related and perception-related. This encompasses many common scenarios in which a sensing AR or team of ARs is employed, ranging from object detection, mobile phenomenon monitoring, or autonomous navigation based on visual localization. The proposed framework is designed to exploit Nonlinear MPC for its capability to jointly express constraints from various semantics under the same mathematical paradigm. Moreover, this framework is exploited to produce constrained inputs for the AR low-level actuation, discarding the need for intermediate regulation. Finally, onboard real-time C++ implementations are provided.

Chapter 2 proposes a review of the literature on the topics of perception in aerial robotics. First, a brief survey of existing designs is proposed, illustrating the need for a generic model. Then, the general uses of perception with ARs are presented, followed by the common strategies for perception-driven control. Finally, after a brief presentation of the concept underpinning optimal control, a survey of N-MPC controllers in aerial robotics is proposed.

The next chapter, Chapter 3, closes the preliminary part of the thesis by introducing the mathematical tools and models exploited in the subsequent chapters. In particular, a generic model for sensing ARs is presented, as well as the mathematical phrasing of the sensor measurements and the associated filtering processes.

The second part starts with the formalization of the problem considered throughout the thesis, in Chapter 4. Firstly, the motivations for the use of N-MPC are discussed. Consequently, a generic N-MPC scheme is designed to handle the tasks of interest. This generic formulation abstracts the perception tasks and objectives in a common mathematical formalism. In the vein of previous works, this controller is designed to produce the motor-torque inputs of the AR, such that the embedded constraints are enforced until the actuator inputs are sent to the flight controller.

In Chapter 5, the perception is properly addressed in the scope of objects detection and monitoring. A geometric observability criteria is expressed and exploited to formalize the suitable objectives and constraints. A validation campaign is performed through simulations and experiments, which demonstrates the applicability of the proposed framework. Therein, the focus is made on the assessment of the constrained behavior yielded by the controller, for both a collinear quadrotor and a tilted-propeller hexarotor. The N-MPC accounts for the specific limitations of both platforms to fulfill its objectives at best. Finally, a practical use case of such controller is described through the presentation of a collaborative work on human-robot object handover in which the perception-constrained control is meaningfully adopted.

In Chapter 6, the previously introduced N-MPC is exploited to enforce visual self-localization while moving in an unknown environment. After a succinct overview of visual-inertial localization techniques, a concrete instance of such estimator is detailed. Therein, we discuss the interfacing of any off-the-shelf estimator with the proposed controller. Consequently, the controller is reframed to comply with the newly stated requirements. Again, an experimental validation is provided, analyzing the resulting system behavior.

Chapter 7 discusses the extension of the perception-aware framework in the scope of Multi-Robots Systems. The chapter describes the various challenges arising during the scaling, in particular regarding the computation distribution and the collision avoidance. Then, the considerations regarding both frameworks from Chapters 5 and 6 are discussed.

Finally, Chapter 8 provides an extended framework in the scope of multi-agent Active Information Acquisition, which aims at tackling some limitations raised in Chapter 5 and Chapter 6. In particular, the framework actively minimizes the object detection uncertainty. The literature on these topics is thoroughly analyzed, and the proposed controller is consequently described. An experimental validation is also conducted to demonstrate the emerging cooperative behaviors yielded by the N-MPC equations.

9.2 Overall Conclusion

The work conducted throughout this thesis yields a paradigm to express perception-aware controller, in the scope of aerial robots performing pre-defined tasks. Numerous challenges could be addressed with such framework. For instance, autonomous monitoring of mobile phenomenon (e.g., in sport events) where several drones can be deployed to provide active coverage, or collaboration with human coworkers in construction works, where such framework could be used, to ensure the safety. Additionally, the perception is modeled such that it allows the N-MPC to collaborate with other robots, fixed sensors, or even human operators. Furthermore, many other considerations that can be modeled in similar ways (i.e. through a state-dependent formalization of some assessment variable) might be introduced in the framework. For example, aerial manipulation might be improved by leveraging a force-torque sensor in a similar fashion.

Despite the complexity of the employed models and equations, the proposed results demonstrated real-time performances. The proposed experiments were achieved using a reasonably powerful onboard computer, exploiting a standard-grade CPU. Although smaller onboard PC often embedded on Micro ARs are not powerful enough to perform the optimization in real-time, similar computation performances could be achieved onboard smaller UAVs with dedicated computers, or through a suited implementation (GPU, Field-Programmable Gate Array (FPGA)...) In particular, recent works presented in Section 2.3 propose onboard CNN for image processing showing real-time performances, paving the way for more and more vision frameworks, such as the one designed in this manuscript.

However, N-MPC frameworks have many applicability limitations that need to be addressed before being deployed in real scenarios. An instance of these is the

sensitivity of the framework to the tunable weights. A framework such as the AIA N-MPC from Chapter 8 requires a large weight on $C^{\text{perception}}$ to yield some result because of the small order of magnitude of the covariances exploited. On the other hand, this makes the framework sensitive to large increases of this covariance, leading to possible instabilities. Similarly, the scalability to multiple agents/sensors/features is compromised by the complexity of the involved models. The computation burden imposed on the onboard PC becomes too heavy to envision such techniques without some suited simplifications, and some more suited tools, in particular regarding feature tracking filters. Another limitation of constrained N-MPC is that some unmodeled event might break the constraints (e.g. collision or wind burst that lead to a loss of visibility, the breaking of a propeller...), thus leading to the impossibility of the framework to provide any input to the motors. Some fail-safe strategies are thereby required to safely handle such events. More generally, the use of N-MPC schemes should always be monitored through a high-level supervisor in charge of enabling/disabling the various constraints and objectives, and which is able to switch to a backup emergency control scheme in case of non-convergence.

9.3 Perspectives

The research presented along this manuscript left open many questions, both theoretical and practical, which would hopefully lead to fruitful works in this domain.

On the theoretical aspect, there is still a lot of work that must be conducted toward the certification of such algorithms in order to aim for an actual field deployment. An extensive convergence analysis of the OCP is mandatory. Although [Grimm, 2005] provides proof that a long enough receding horizon ensure the stability, this is done for unconstrained N-MPC and with careful assumptions on the cost function. However, evidences are provided that a well designed terminal cost can result in a faster convergence.

It seems inevitable, due to the multiplication of the objectives and constraints assigned to the ARs, to ultimately lead to unsolvable conflicts. This is illustrated with perception tasks as defined in Chapter 5, where an infinity of configurations are fulfilling the perception objective, but some are jeopardizing the stability (with a large tilting angle) and thus conflicting with the motion task. In that case, hierarchical approaches should be employed to prioritize the system stability over the fulfillment of the "work" task, for obvious safety reasons, as it is commonly done in more complex robotic systems such as redundant manipulators or humanoids [Mansard, 2009].

On the practical aspects, many improvements are yet to be made regarding the proposed solutions. An obvious first step toward more realistic scenarios would be to get rid of the fiducial markers, through the use of actual computer vision algorithms. As previously mentioned, CNN are nowadays the de facto standard for perceptive data processing, be it 2D or 3D imagery. Embedding these algorithms implies a specific retraining in order to fit the use-case conditions. The perception model used throughout this thesis might need some fine-tuning since the punctual assumption might be problematic (e.g. if the object to detect is quite large and needs to be seen entirely to be detected by the software). The exploitation of CNN providing an uncertainty estimate is particularly promising for filtering [Tremblay, 2018].

Furthermore, a supervision algorithm should be employed to monitor the feasibility of the optimization, and trigger a safety backup solution. This can be the case when the OCP becomes unsolvable, in particular when using the RTI solving strategy which might prevent the convergence toward an existing solution. This is even more relevant when the AR is achieving agile motion. Such supervisor can also be employed for a better handling of collision avoidance, e.g. through a priority-based avoidance scheme.

Additionally, the proposed experiments could be pushed further, e.g. by replacing the MoCap with some efficient visual-inertial state estimator. Transitioning to outdoor environments is challenging and a proper state estimation process is a mandatory feature. Using such estimator is also important for testing the framework from Chapter 6. Similarly, more diverse sensors should be employed to prove the applicability of our framework with those. Among those, lidar and event based cameras constitute two truly interesting alternatives to classical cameras, because of their respective ability to produce precise mapping of the surroundings, and to provide very high-frequency sensing at a small computational cost. Finally, an in-depth benchmarking needs to be conducted to numerically assess the advantages and drawbacks of the full-state torque-level N-MPC compared to cascaded approach in the scope of perception-oriented controllers.

Throughout this thesis, tilted-propeller hexarotors have not been successfully used in experiments because of the increased sensitivity of the N-MPC to model errors. Indeed, the fully-actuated system possesses more than one equilibrium point, therefore the basin of attraction is much smaller than in the underactuated case. Additionally, fully-actuated platforms usually have a more complex mechanical design, increasing the margin for errors. For instance, the uncertain tilting angle of the rotors induces uncertainty on the actuator thrusts (not only on the norms, but also the directions). The electronics can also become a bottleneck, since more bandwidth is required to read the propeller data, implying a lower state feedback frequency. For all the aforementioned reasons, successful experiments with such platform has proven more difficult. During this thesis, the focus has not been set on improving the model knowledge of the platform currently available at LAAS-CNRS (see Figure D.2). Successfully flying such platform with our proposed controller would be a logical next step in order to thoroughly validate the proposed framework. We are however confident that there is no theoretical limitation at stake, since previous works [Bicego, 2020] tackled the flight of a real tilted-propeller hexarotor with a similar N-MPC. This raises the need of a comprehensive study of the robustness of N-MPC to model mismatches, in the vein of [Sun, 2022].

9.4 Final Thoughts

Despite the many aspects that are still to be improved, MPC and N-MPC gained a lot of attention from the aerial robotics research community over the past years. These are powerful tools used mainly to tackle complex scenarios where traditional reactive controllers are not best suited. The study of those controllers led, on the one side, to the emergence of very efficient N-MPC algorithms, and on the other side to complex modeling suited for a large range of applications. The authors in [Sun,

2022] provided, through a comprehensive comparison, some pieces of evidence that N-MPC controllers might outperform and be more resilient to model mismatches than state-of-the-art geometric controllers in the scope of agile maneuvering or drone racing, when employed with a proper regulation. The recent emergence of several tentatives to consider perception and control in a joint paradigm is an indication that the topics treated throughout this thesis are relevant to the field, and that algorithms and technologies are becoming mature enough to contemplate this possibility. This thesis humbly contributed to this ongoing field of study.

Appendix A

Simulation and Experimental Setup

Figure A.1: Snapshot of a quadrotor used in the various experiments. It is equipped with MoCap reflecting balls on top and a front-facing camera.

Software Description

The whole framework is implemented in C++, using GenoM [Mallet, 2010] which is a middleware-independent component generator. GenoM allows to design real-time modular software architectures, most specifically for robotics. Each module is defined as a state machine, formalized through a generic template file which allows to define states, transitions, internal variables, etc. Then, a piece of code is associated with each state, and transition conditions are specified directly in C++. Finally, some compilation templates are defined to generate the corresponding component in a given middleware, e.g. ROS[1] or Pocolibs[2]. Therefore, all components share the same consistent behavior. More details are available on the OpenRobots platform[3]. Pocolibs is a synchronous system communication middleware which allows to run parallel components (on various computers) and handles the communication among those, as ROS would do.

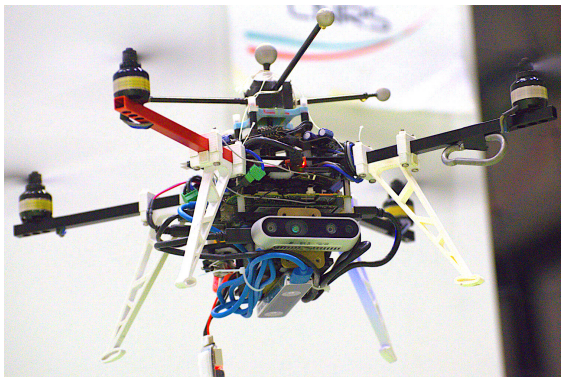
The hardware interface as well as the state estimation and high-level path planning are done using the TeleKyb3 software[4]. The reference path planning is performed using a motion planner described in [Boeuf, 2015]. When a waypoint is requested by the supervisor, the path planner uses B-splines to interpolate a path from the current position that satisfies some rough kinodynamic constraints. The computed trajectory provides a reference in position and yaw, as well as their first- and second-order derivatives. This trajectory is sampled at 100Hz, and sent to the controller in open loop.

The state estimation is achieved through a UKF [Julier, 1997], fusing data from an

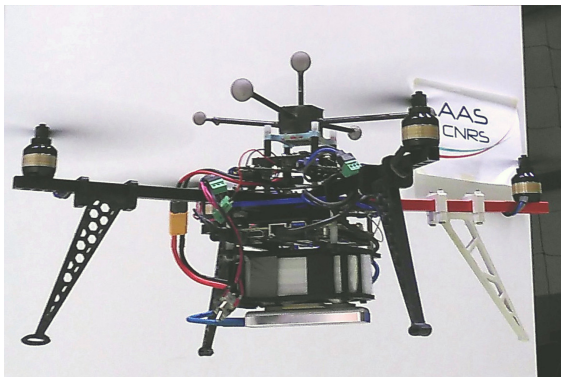
[1] https://www.ros.org/
[2] https://git.openrobots.org/projects/pocolibs
[3] https://git.openrobots.org/projects/genom3
[4] https://git.openrobots.org/projects/telekyb3

onboard IMU and an external MoCap, (except, obviously, in the hardware experiments conducted in Chapter 6, since the ESKF described therein is implemented and used). This UKF uses the formulation proposed in [Crassidis, 2003]. The UKF estimates the robot position, velocity, acceleration, attitude (as quaternions) and angular rates. It makes the assumption of constant linear and zero rotational accelerations. As in Section 6.2.2, this filter assumes a pre-calibrated IMU (to estimate its scaling factors, biases and standard deviations). Specifically, we employ the calibration procedure introduced in [Tedaldi, 2014]. The state estimation software runs at 1 kHz. The various state estimation sensors frequency (both for simulation and experiments) and generated noise (for simulation) are reported in Table A.1.

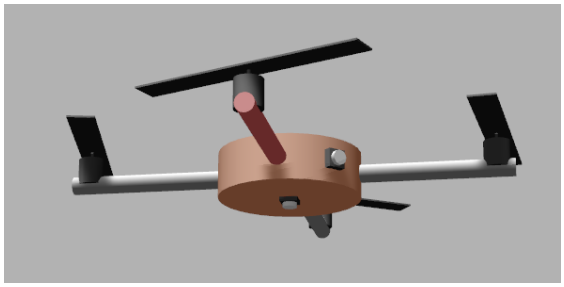
The N-MPC implementation is based on CPPMPC, the C++ implementation of the MATMPC software [Chen, 2019], which uses the direct multiple shooting method. The mathematical equations are translated to C++ code using CasADi [Andersson, 2019], and the discretization is done using a 4^{th} order explicit Runge-Kutta integrator. Details on this implementation are reported in Appendix B.



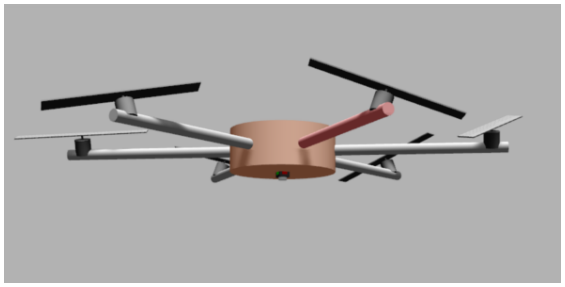
(a) Snapshot of a flying quadrotor with both down-facing and front-facing cameras, used the experiments from Chapters 5.



(b) Snapshot of a flying quadrotor with a down-facing camera, used the experiments from Chapters 6 and 8.



(c) Snapshot of the simulated quadrotor in Gazebo, with both down-facing and front-facing cameras.



(d) Snapshot of the simulated hexarotor in Gazebo, with a down-facing camera.

Figure A.2: Quadrotors and simulated ARs used in simulations and experiments.

Sensor	Frequency	Gaussian std
Gyroscope	500 Hz	0.02 rad/s
Accelerometer	500 Hz	0.1 m/s^2
MoCap	100 Hz	0.003 m
Rotor positions	20 Hz	0.03 rad

Table A.1: State estimation sensors frequencies and simulated noise.

(a) The T265, a stereo fisheye camera for visual odometry.

(b) The D435, a stereo infrared RGBD camera.

Figure A.3: The two RealSense cameras used.

Simulation and Hardware

This overall software stack is either connected to an AR for real experiments through a flight controller (in our case, a MikroKoper[5] or Paparazzi[6] board), or to the Gazebo simulator which emulates the same interface through a dedicated plugin[7]. Changing between both is therefore seamless – at least implementation-wise.

The drones used in the experiments are collinear quadrotors with arms of 23 cm. Each weight 1.3 kg, including battery, Intel NUC[8] and cameras. The same values are used in simulation. Additionally, the simulated tilted-propellers hexarotor is a Tilt-Hex [Rajappa, 2015]. The all $\alpha_{a,i}$ and $\beta_{a,i}$ propeller angles are equal and set respectively to 20° and 0°, which have empirically been shown to provide a good trade-off between actuation span and energy-efficiency. The hexarotor weights 2.5 kg, with arms of 39 cm. The computer runs with Ubuntu 18.04, with an Intel Core i7-8565U and 8GB of DDR3 RAM.

Cameras

The GTMR is equipped with monocular cameras. In Gazebo, we use the embedded camera simulator, with a halved horizontal FoV $\alpha_h = 1\,\text{rad}$, and an aspect ratio $\alpha_h/\alpha_v = \frac{4}{3}$. The simulated camera is grayscale and its frequency is set to 60 Hz. On the actual platform, we use the Intel RealSense T265[9] and D435[10], see Figure A.3.

[5] https://wiki.mikrokopter.de/en/FlightCtrl
[6] https://wiki.paparazziuav.org/wiki/Main_Page
[7] https://git.openrobots.org/projects/mrsim-gazebo
[8] https://en.wikipedia.org/wiki/Next_Unit_of_Computing
[9] https://www.intelrealsense.com/tracking-camera-t265/
[10] https://www.intelrealsense.com/depth-camera-d435/

These are chosen for their practicality of use, both hardware and software (lightweight, easy embedding, dedicated off-the-shelf open libraries...), though they are rigorously exploited as monocular cameras. None of the depth sensing nor tracking functionalities are used. The T265 is grayscale and runs at 60 Hz. It is rectified and undistorted to achieve a FoV of $\frac{\pi}{4}$ rad, with aspect ratio 1. The D435 is however RGB and set to 30 Hz, with a FoV of 70° and aspect ratio $\frac{16}{9}$. The image processing (i.e. AruCo markers detection and 6D pose + covariance estimation) takes about 3 to 5 ms per frame, on CPU, using OpenCV [Bradski, 2000].

Appendix B

Implementation Details

Introduction

This appendix is an overview of the N-MPC implementation used in the main body of this thesis. It encompasses a brief review of existing open-source optimization frameworks for robotics, followed by an in-depth presentation of the proposed control algorithm, including some technical considerations. Finally, the supervisor framework is succinctly presented.

The code of the proposed controller will be released open-source on the Open-Robots platform[1] in an off-the-shelf version, which is meant to be as generic as possible w.r.t. the perception objectives. So far, the code used in all the individual experiments and simulations presented along this manuscript are available online, each time with a comprehensive wiki page to guide the installation and the performing of simulations. In includes:

1. the perception-constrained N-MPC from Chapter 5[2];
2. the human-robot handover simulation from Section 5.8[3];
3. the N-MPC for enforced visual state estimation from Chapter 6[4];
4. the N-MPC-AIA framework from Chapter 8[5];
5. the visual servoing scheme for physical interaction from Appendix D.2[6].

N-MPC Software

There exist several open-source libraries for MPC and N-MPC. This section briefly presents some of those. The controller presented in [Kamel, 2017] is available for ROS on Github[7]. It handles linear and nonlinear MPC. Another toolbox for MPC is the Control Toolbox presented in [Giftthaler, 2018]. Additionally, the Perception-Aware MPC from [Falanga, 2018] is also publicly available for ROS[8].

The latter implementation uses the ACADO toolbox [Houska, 2011], one of the most famous MPC software[9]. A new, more complete, version called ACADOS [Verschueren, 2021] is for instance used in the implementation of [Barros Carlos, 2021]. Acados is available online[10]. Another N-MPC toolbox is MATMPC [Chen, 2019], a Matlab framework for N-MPC[11]. Its specificity is to implement the fixed-time block update RTI presented in [Chen, 2017], which performs the linearization of the NLP based on an estimate of a curvature measure of the model function.

One of the most famous solvers, which is used in the aforementioned toolboxes, is qpOASES [Ferreau, 2014], a well-established open source implementation of the Active

[1] https://git.openrobots.org/
[2] https://redmine.laas.fr/projects/perceptive-torque-nmpc
[3] https://redmine.laas.fr/projects/nmpc-handover
[4] https://redmine.laas.fr/projects/nmpc-localization
[5] https://redmine.laas.fr/projects/active-perception-nmpc
[6] https://redmine.laas.fr/projects/visual-physical-control-architecture
[7] https://github.com/ethz-asl/mav_control_rw
[8] https://github.com/uzh-rpg/rpg_mpc
[9] https://acado.github.io/
[10] https://docs.acados.org/
[11] https://github.com/chenyutao36/MATMPC

Set method for QP solving. Another commonly used QP solver is HPIPM [Frison, 2020], which implements the Interior Point method.

To handle symbolic computing, the aforementioned toolboxes make use of CasADi [Andersson, 2019]. Using this framework, the symbolic formula for dynamics and problem differentiation are translated into C++ code and embedded inside the controller. Another solution is to use analytical computation with a fast numerical implementation. An instance of such toolbox for robotics is Pinnocchio [Carpentier, 2019], available online[12]. It is for instance used for optimal control in the scope of contact-based robotics controllers in Crocoddyl [Mastalli, 2020], also available open-source[13], used, e.g., in [Dantec, 2021].

N-MPC Implementation

Software description

The N-MPC implementation is made in GenoM, a component generation framework for robotics, described in Appendix A. It uses CPPMPC[14], the C++ implementation of the aforementioned MATMPC toolbox. It uses the implicit direct multiple-shooting method to generate the NLP, using the RTI solving policy. The subsequent SQPs is solved with qpOASES.

The models of the ARs and all the derivatives required for the definition of the NLP are defined in python3. The discretization is done using a 4^{th} order explicit Runge-Kutta integrator. All the symbolic computations are exported to C++ code, using CasADi. Both these auto-generate model files and the CPPMPC solver code are directly embedded inside the code of the N-MPC component. Therefore, the component is compiled, usually for Pocolibs.

In the model definition, the quaternions are integrated using the policy introduced in [Rucker, 2018]. It consists of using non-unit quaternions along with an adequate mapping from $\mathbb{Q}$ to $SO(3)$, and implements a proper regularization in the derivation and integration formula.

The controller is implemented in two parallel threads, which modify a shared pool of internal variables. The GenoM framework handles concurrent variable access. The first thread is the main control loop. Due to the unpredictable duration of the optimization step, it is asynchronous, i.e. is not blocking w.r.t. the rest of the software stack. Once the controller is enabled, this thread performs a continuous loop that updates the initial state vector $\mathbf{x}_0$, generates and solves the SQP, integrate and convert the N-MPC input $\mathbf{u}_0$ into the rotor velocities $\mathbf{\Omega}$, and send those to the flight controller.

The second thread is synchronous to facilitate the interaction with the other components, namely the state estimator, trajectory generation, and feature detector.

Two additional features are implemented to allow the performing of the task. First, the take-off activity is not possible using the N-MPC controller since the

[12] https://github.com/stack-of-tasks/pinocchio
[13] https://github.com/loco-3d/crocoddyl
[14] https://github.com/chenyutao36/cppmpc

ground force compensating for gravity is not modeled in any way. To palliate this, an open-loop spinning of the propellers is performed until reaching a steady spinning rate close to hovering velocities. From there, the N-MPC is enabled to take over the control of the platform from a valid state.

Second, in order to ensure the safety of the AR flight to possible N-MPC failure, a fail-safe strategy is implemented. A constant monitoring is performed of the return value of the qpOASES solver, which assesses if the SQP was successfully solved. Additionally, the N-MPC state, inputs and cost function values are monitored to check for numerical singularities. In case of such events, the controller goes into an emergency procedure which disables all the extra constraints acting on the system (in particular, perception constraints) as well as all the extra objectives. The N-MPC matrices are reinitialized, and the reference motion is set to hover in place with 0 roll and pitch This procedure ensures that the AR won't crash or behave unexpectedly in case of unsolved SQP.

Computation Time

The frequency of the controller is saturated to 500 Hz, i.e. the controller idles if the optimal control cycle is faster than 2 ms. Otherwise, the controller starts another cycle as soon as the current one finishes.

The computational time of the N-MPC on an onboard computer (Intel NUC with an Intel Core i7-8565U and 8GB of DDR3 RAM) have been recorded during the various experiments. Figure B.1 reports a box plot of the computational time of the N-MPC component during the experiments presented in Section 8.5, with 1 and 2 UAVs performing the tracking of a static or moving feature. It covers a flight time of about 10 min. The horizon length for these experiments is $T = 0.75$ s, sampled in $N = 20$ points. The average computation time is 0.87 ms, and the last percentile of recorded control cycle ranges from 1.74 to 8.6 ms.

We empirically tested that a control period artificially downgraded to 40 ms still allows the platform to fly and perform simple maneuvers. The observed outliers are still far below this value.

This illustrates the capability of the proposed framework to compute the torque-level inputs of the UAV, onboard and in real-time.

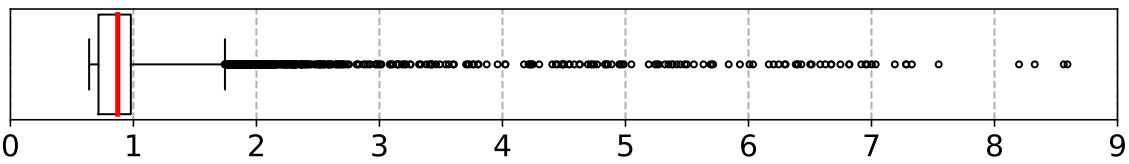


Figure B.1: Box plot of the computation time (in milliseconds) of the onboard N-MPC. The red is the mean, and the black dots are the slowest 1% control cycles recorded.

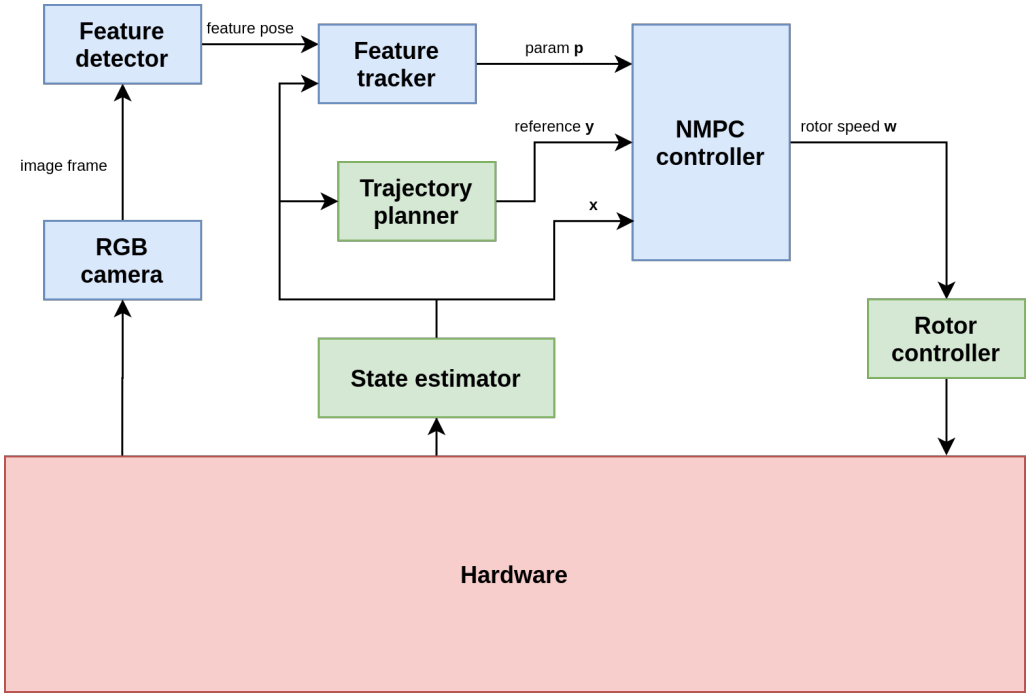


Figure B.2: Architecture of the AR software stack. In blue are the components developed throughout the thesis.

Supervisor Implementation

The interaction with the N-MPC controller and the overall software stack is achieved using Genomix[15], an HTTP server interfacing clients (i.e. a Matlab, Python or TCL script running on the user computer) and the GenoM components, which are running on the AR. The connection between the two is made through regular wifi.

The supervisor is made of a set of scripts sending the desired configurations to the AR, and connecting the inputs and outputs of all components according to the mission requirement. It also handles the interactions with the components through GenoM *services*, i.e. pieces of code that are enabled through the external call of a function that performs a pre-defined action. These are used, e.g,, to change the values of some parameters of the components, or enable some tasks. In particular, the trajectory generator[16] used to define the motion task can receive successive commands to produce the trajectory toward a set of waypoints.

The architecture of the client supervisor in the aforementioned git repositories, which were provided along the submitted papers from Table 1.1, is made of two types of files:

- the parameter files, that collect the various parametric values to configure the components according to the AR and the mission;
- the mission files, which defines successive inputs to send to the AR to complete the experiment (start the motors, take-off, enable the perception constraints, reach successive waypoints. . .).

[15] https://git.openrobots.org/projects/genomix
[16] https://git.openrobots.org/projects/maneuver-genom3

Additionally, some static initialization files are required to perform the in/out connections, and to launch the software on the remote PC.

Figure B.2 depicts the overall software architecture that is running on the onboard computer.

Appendix C

Center of Mass Estimation

Estimation Method

This short appendix describes a simple linear least-square procedure to estimate the CoM position for an AR. Indeed, the model presented in Chapter 3 relies on this quantity, and as mentioned in Section 3.4.1 it is in general hard to obtain.

This procedure assumes that a robust PID controller is available in order to achieve a steady hovering with the AR. The integral action is fundamental to cancel any steady state error, e.g. induced by the CoM offset or any other modeling uncertainty. Then, recording a sufficiently large sample of data (i.e. for 30 sec or 1 min of flight time), Equation (3.21d) can be exploited through least-square estimator to identify ${}^{B}\mathbf{p}_{CoM}$. Those data are in particular the controller-generated body wrench and the body orientation.

We note that the angular acceleration $\dot{\boldsymbol{\omega}}$ of the AR is in general not estimated. Differentiating $\boldsymbol{\omega}$, which is inherently noisy, is in general not a satisfactory solution. Additionally, as previously mentioned, the inertia tensor $\boldsymbol{I}$ is in general not known precisely around O_B. As a consequence, maintaining a steady hovering allows to nullify the transient terms and simplify the estimation process. In these conditions, Equation (3.21d) is written

$$\cancel{\boldsymbol{I}\dot{\boldsymbol{\omega}}} = \cancel{-\boldsymbol{\omega}\times\boldsymbol{I}\boldsymbol{\omega}} + \boldsymbol{\tau}_u + {}^{B}\mathbf{p}_{CoM} \times {}^{B}\mathbf{g}, \tag{C.1}$$

where $\boldsymbol{\tau}_u$ is the torque requested by the PID controller, and ${}^{B}\mathbf{g} = -mg\,{}^{B}\mathbf{R}_{W}\mathbf{z}_{W}$.

Aggregating the collected data over the time samples $t \in \{1,\ N\}$, we denote $\boldsymbol{\tau}$ and $\mathbf{S}_\mathbf{g}$ the quantities defined by

$$\boldsymbol{\tau} = \begin{bmatrix} \boldsymbol{\tau}_u(t=1) \\ \vdots \\ \boldsymbol{\tau}_u(t=N) \end{bmatrix}, \qquad \mathbf{S}_\mathbf{g} = \begin{bmatrix} [{}^{B}\mathbf{g}(t=1)]_\times \\ \vdots \\ [{}^{B}\mathbf{g}(t=N)]_\times \end{bmatrix}. \tag{C.2}$$

Finally, ${}^{B}\mathbf{p}_{CoM}$ can be obtained by solving the linear least-square problem

$$\min_{\mathbf{x}} \|\mathbf{S}_\mathbf{g}\mathbf{x} - \boldsymbol{\tau}\|. \tag{C.3}$$

Remark. For underactuated ARs such as quadrotors, reaching a steady hovering state implies having zero roll and pitch, hence ${}^{B}\mathbf{g} = {}^{W}\mathbf{g}$. Therefore, the third component of ${}^{B}\mathbf{p}_{CoM} = \begin{bmatrix} p_x & p_y & p_z \end{bmatrix}^\top$ along $\mathbf{z}_B$ cannot be observed since it disappears in the cross product in Equation (C.1):

$${}^{B}\mathbf{p}_{CoM} \times \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} = -mg \begin{bmatrix} p_y \\ -p_x \\ 0 \end{bmatrix}. \tag{C.4}$$

Nonetheless, knowing the x and y still allows a meaningful improvement of the trajectory tracking, and allows to cancel the steady state errors when hovering, as reported in the next section. For fully-actuated ARs, such as tilted-propellers hexarotors, hovering in a tilted configuration allows to estimate all three components of ${}^{B}\mathbf{p}_{CoM}$.

Quantitative Evaluation

In Table C.1, we report the position Root Mean Square Error (RMSE) of two trajectories, both with the estimated CoM compensation, and without (i.e., assuming ${}^{B}\mathbf{p}_{CoM} = 0$). The experiment is conducted with a quadrotor with no prior knowledge of the offset. Therefore, only the x and y components are estimated, showing an offset of $\|{}^{B}\mathbf{p}_{CoM}\| = 0.75$ cm.

The two trajectories considered for this test are:

1. a steady hover for 30 sec,
2. the replay of a given trajectory (waypoints interpolated with B-splines) for a flight time of about 20 sec. The maximum linear and angular velocities and linear acceleration are, respectively, 1.9 m.s^{-1}, 1.5 rad.s^{-1} and 3.2 m.s^{-2}.

The angular RMSE is not reported since the CoM offset does not directly affect the attitude tracking.

	with	without	improvement
hovering	0.0436	0.1166	63%
trajectory tracking	0.2124	0.3188	34%

Table C.1: Position RMSE, in meter, for two different trajectories, with and without CoM offset compensation.

Appendix D

Collaborative Works

Contents

D.1 Participation to MBZIRC 2020 . . . . . **162**
D.1.1 Introduction 162
D.1.2 Brick Detection 163
D.1.3 Data Association and Tracking 165
D.2 Generic Control Scheme for Vision-Based Physical Interaction . **165**
D.2.1 Introduction and Motivation 165
D.2.2 Modeling 166
D.2.3 Control Architecture 167
D.2.4 Experimental Results 169
D.2.5 Conclusion 171

D.1 Participation to MBZIRC 2020

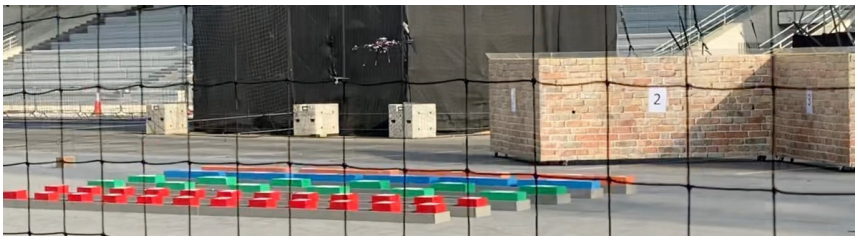


Figure D.1: The arena of the second challenge in MBZIRC 2020.

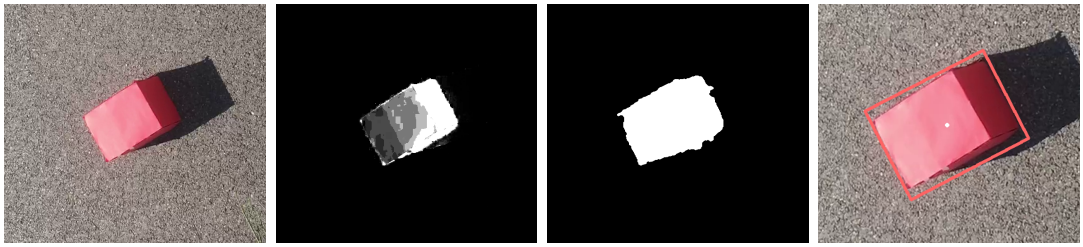
D.1.1 Introduction

The MBZIRC challenge is an international robotic challenge held every two years in Abu Dhabi. It gathers several teams from universities around the world, and aims to be at the state of the art of robotics, while stimulating technical innovations. The competition is not specifically oriented toward aerial robotics, yet it often has a large place therein, most of the time in cooperation among ARs or with other robots.

The LAAS-CNRS team was involved in the second challenge of the 2020 edition, which consisted of autonomous cooperative pick-and-place of several colored bricks on a mock-up wall (see Figure D.1). This challenging task led the LAAS-CNRS team to build a new platform, the FibertHex (see Figure D.2), and a completely customized software stack, including the low-level control.

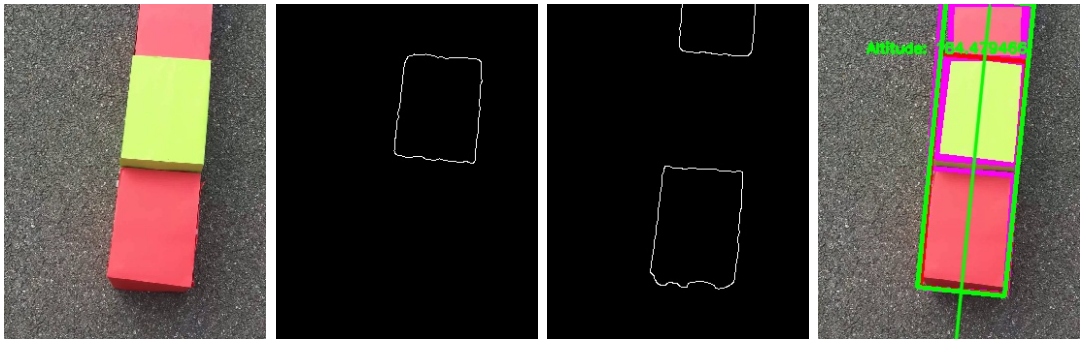
Objectives corollary to the work conducted along this thesis are related to the detection and mapping of the bricks, as well as the mock-up wall. The section briefly presents the two brick detection pipelines implemented, as well as the KF-based mapping policy employed for the competition. The detection algorithms were collaboratively designed and implemented.

Figure D.2: The LAAS-CNRS aerial platform, the FibertHex.



(a) Input image. (b) Histogram back-projection. (c) Segmented area. (d) Final detection.

Figure D.3: Histogram back-projection-based detection of the colored bricks (courtesy of [Dantec, 2019]).



(a) Input image. (b) Green segmentation. (c) Red segmentation. (d) Final output.

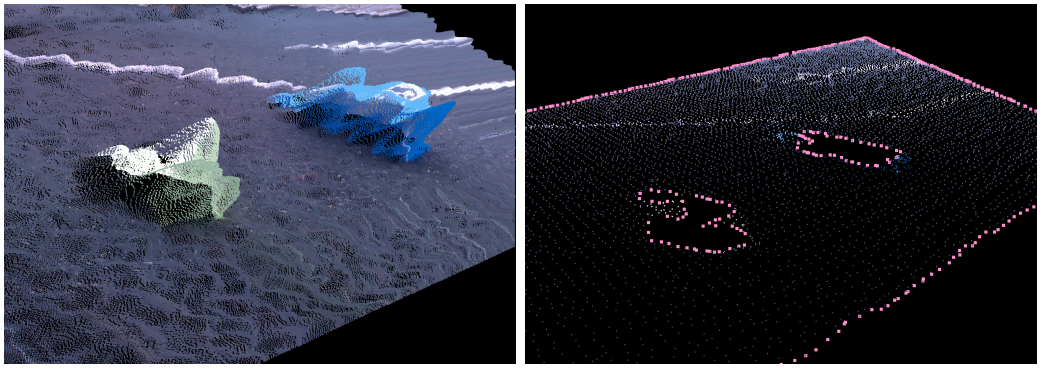
Figure D.4: Histogram back-projection-based detection of the colored bricks (courtesy of [Dantec, 2019]).

D.1.2 Brick Detection

In order to perform the pick-and-place operations, the robot needs to be able to detect and track bricks of different colors and sizes. In fact, since the position of the bricks cannot be known very precisely, the pick-and-place must rely on image features of the brick of interest. In fact, a VS controller scheme is implemented, similar to the one presented in Section D.2.3, which uses the brick centroid and brick orientation along the vertical axis (yaw). The choice of these parameters is essentially due to the nature of the performed operation: a brick to be picked and correctly placed, has to be firstly grasped on its upper face centroid and with a specific orientation. The proposed method relies on the precise knowledge of the brick color and shape, since these are provided. Moreover, all the bricks of a given color have the same size. Given the robot state and a camera image related to that state, the brick detection module plays the role of localizing bricks in the image plane, by computing and outputting their centroid and yaw features. This information is passed to the brick tracking module as input, that plays the role of filtering these measures.

D.1.2.a Monocular Detection

First, a monocular imaging approach is proposed for the brick centroid and yaw assessment. This is achieved using a classical histogram back-projection [Swain, 1992] on the HSV color space, allowing to extract specific color chunks from the image as



(a) Noisy point cloud taken from afar.

(b) Extracted ground surface with holes corresponding to bricks.

Figure D.5: RGBD ground extraction for brick detection (courtesy of [Dantec, 2019]).

binary images. Histogram back-projection is based on the per-pixel comparison with an existing histogram. The probability that a pixel shares the color that appears in the histogram is computed, based on colorimetric distance. These probabilities form a grayscale image, where the "whitest" pixels are more likely to have the expected color. A thresholding is performed, followed by morphological operations to get rid of outliers and fill the obtained chunks. Therefore, the knowledge of the brick shape is exploited, namely the length/width ratio. The chunks are processed and the aspect ratio of the detected hull is computed, and compared to the reference one. A tolerance margin is considered, since the brick can be observed from the side. Figure D.3 illustrates of the proposed detection pipeline.

Once the bricks are detected, a Hough transform [Duda, 1972] (after a Canny edges extraction [Canny, 1986]) is applied to assess the orientation of both pairs of sides. The median of the longest sides for each brick returns the yaw orientation. This technique is also employed to detect the brick wall orientation, see Figure D.4.

D.1.2.b Depth-based Detection

Another approach is based the use of RGBD sensors (namely, the RealSense D435, see Figure A.3b). This camera is used, e.g., to produce 3D point clouds through the use of stereo and infrared distancing. However, point clouds are very noisy, and it is therefore though to perform an accurate processing on those (see Figure D.5a).

However, the impact of the noise is reduced on large-area surfaces, e.g., the ground (which is, indeed, assumed flat). Thereby, a workaround solution is to segment the ground from the input point cloud. From there, the "shadow" cast by the bricks, which are occulting the ground for the camera, can be exploited to retrieve the convex hulls of the bricks. A RANdom SAmpling Consensus (RANSAC) algorithm allows to retrieve the ground plan.

The holes induced by the brick shadows are retrieved using an alpha-shape policy [Trinh, 2015]. It is a generalization of the convex hull of a point distribution, based on a parametric value α defining the radius of spheres that are used to detect the edges by assessing whether or not they contain some points. The prior shape knowledge is also exploited to assess the orientation and type of the brick. Figure D.5

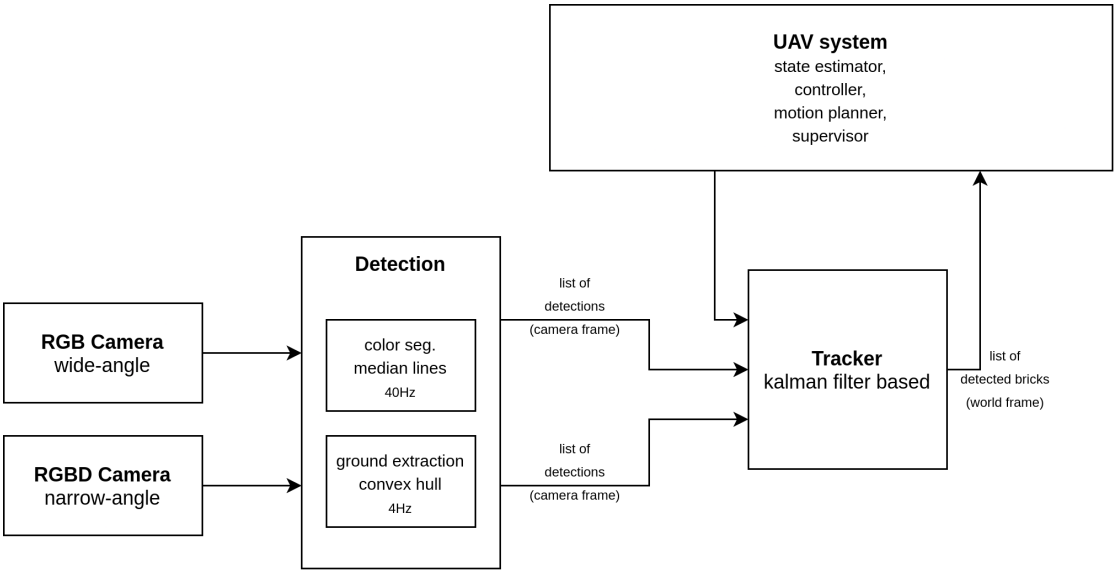


Figure D.6: Software architecture of the perception software stack.

shows the proposed brick shadow detection.

D.1.3 Data Association and Tracking

The detected bricks parameters (position and yaw) are passed to the brick tracking module. This algorithm is used to associate each new detection either to a known brick position, hence refining the knowledge of its position, or to a new instance of brick. Such an algorithm is in fact based on the essential assumption that the bricks are static in the inertial frame. Exploiting the knowledge of the AR position in the reference frame, the detections – which are performed in camera frame $\mathcal{F}_C$ – are rototranslated to $\mathcal{F}_W$. Thereby, a distance is computed (based on 3D position + yaw) between the detection for each known brick to match the new detection to a previous one.

From there, a KF is used for the tracking of each brick, and matching new detections are used as measurements. Given the complexity of the employed detection algorithms, the explicit propagation of the raw sensor uncertainty toward the brick 4D pose knowledge is out of the question (in particular for the RGBD processing). Therefore, an ad hoc Gaussian noise similar to Equation (8.8) is used as measurement covariance. Figure D.6 shows a block diagram of the perception software stack.

D.2 Generic Control Scheme for Vision-Based Physical Interaction

D.2.1 Introduction and Motivation

During our involvement in the MBZIRC challenge, one of the main challenges was the development of a unified framework for vision-based physical interaction. In such tasks, the aim is to interact through contact with the environment, either

for grasping [Augugliaro, 2014] or sensing (e.g. surface inspection [Tognon, 2019]). Therefore, an end-effector is often placed onto a n-DoF robotic arms [Kim, 2013; Baizid, 2017; Tognon, 2017]. This allows to overcome the underactuation, and thus enhances the dexterity of the platform. However, this solution is not free of drawbacks, e.g., the weight of the attached manipulator arm decreases the available payload and the flight time, increasing at the same time the overall mechanical complexity.

To overcome these drawbacks, a novel solution is offered by *the Flying End-Effector* paradigm [Ryll, 2019]. Fully-actuated ARs are used with rigidly attached tools. The extra actuation of the platform allows to control for 6D pose of the tool, according to the task to perform, without the need for redundant DoFs offered by a robotic arm. The research community started to adopt fully-actuated ARs in contact-based applications [Jiang, 2018; Bodie, 2021].

The autonomous performing of the task is most often driven by visual cues. As presented in Section 2.4, vision-based control are widely used in robotics, and can be employed to fulfill the interaction task.

In this section, a generic scheme is proposed to perform physical interaction tasks with fully-actuated ARs. The control is achieved through an HVS scheme. It uses an Admittance Filter (AF) for compliancy at the end-effector, as well as a Wrench Observer (WO) for forces and torques estimation during the contact phase. The control scheme is designed to be autonomous and generic, e.g. only high-level planning is required. A particular implementation is proposed to validate the scheme in an autonomous pick-and-place operation, similar to the context of the MBZIRC challenge. It has been tested in simulations and real experiments. In order to simplify the detection process w.r.t. the complex MBZIRC scenario, the bricks are marked with fiducial markers, as described in Chapter 3.

The contributions of this section are:

- Generic control scheme for Visual Servoing-based physical interactions,
- A fully onboard implementation and its validation in simulations and experiments.

The work presented hereafter led to a publication: [Corsini, 2021].

D.2.2 Modeling

The fully actuated AR is equipped with an end-effector suited for the desired task, and with a monocular camera, provided along with a software processing stack for the detection of the feature of interest. This work also makes use of the GTMR model from Section 3.4, with the differences that the CoM is assumed coincident with O_B, and the external contact wrench needs to be accounted for. This is achieved by modifying the dynamics equations Equations (3.21c) and (3.21d). This wrench applied at the end-effector, and expressed in the end-effector $\mathcal{F}_E$, is denoted

$$\begin{bmatrix} {}^E\mathbf{f}_E \\ {}^E\boldsymbol{\tau}_E \end{bmatrix} \in \mathbb{R}^6. \tag{D.1}$$

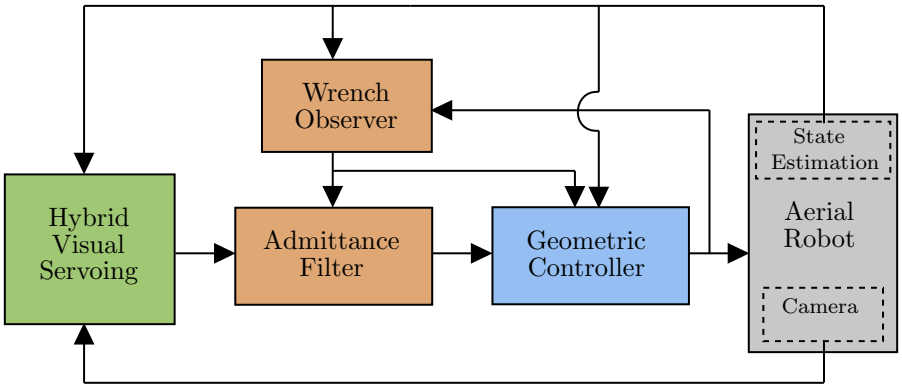


Figure D.7: Generic control architecture for vision-based physical interaction with fully-actuated platforms. In green the vision-based control, in orange the control in charge of the physical interaction, in blue the geometric controller, and in gray the robotic system.

Because the end-effector is attached to a rigid tool, its relative pose to $\mathcal{F}_B$, ${}^B\mathbf{T}_E$, is assumed constant and known.

The modified dynamic equations are then written

$$m\,{}^W\dot{\mathbf{v}}_B = -mg\mathbf{z}_W + {}^W\mathbf{R}_B\mathbf{G}_f\boldsymbol{\gamma} + {}^W\mathbf{R}_E\,{}^E\mathbf{f}_E, \tag{D.2a}$$

$$\mathbf{J}\,{}^B\dot{\boldsymbol{\omega}}_B = -{}^B\boldsymbol{\omega}_B \times \mathbf{J}\,{}^B\boldsymbol{\omega}_B + \mathbf{G}_\tau\boldsymbol{\gamma} + \begin{bmatrix} [{}^B\mathbf{p}_E]_\times\,{}^B\mathbf{R}_E & {}^B\mathbf{R}_E \end{bmatrix} \begin{bmatrix} {}^E\mathbf{f}_E \\ {}^E\boldsymbol{\tau}_E \end{bmatrix}, \tag{D.2b}$$

D.2.3 Control Architecture

The proposed control architecture makes use of 4 blocks, which are depicted in the block diagram Figure D.7. Contrary to the full-state controllers presented in the main parts of the thesis, we employ a cascaded controller, in which a reference trajectory is provided to a geometric controller [Franchi, 2018]. It allows the decoupling of the position and attitude control of fully-actuated ARs. We refer to this article, in which the complete derivation and theoretical proof can be found.

The HVS scheme is detailed hereafter, after [Chaumette, 2007]. First, the visual feature vector $\mathbf{s} \in \mathbb{R}^6$ and its reference $\mathbf{s}_r$ are chosen such that the tracking error can be defined as

$$\mathbf{e} = \mathbf{s} - \mathbf{s}_r. \tag{D.3}$$

In a classical HVS scheme, the feature vector is defined as

$$\mathbf{s} = \begin{bmatrix} \mathbf{x} \\ \log z_M \\ \theta\mathbf{u} \end{bmatrix} \in \mathbb{R}^6, \tag{D.4}$$

where $\mathbf{x} \in \mathbb{R}^2$ is the (x, y) position of the feature which can be defined either in camera frame, normalized camera coordinates, or pixel coordinates, while $z_M > 0$ is the position of the feature along the principal axis of the camera $\mathbf{z}_C$. $\theta\mathbf{u} \in \mathbb{R}^3$ is the

angle-axis representation of the orientation error. In the following, we present the equations for $\mathbf{x}$ being the normalized camera coordinates of the detected feature M, that is

$$\mathbf{x} = \frac{1}{z_{_M}} \begin{bmatrix} x_{_M} \\ y_{_M} \end{bmatrix}. \tag{D.5}$$

The reference vector $\mathbf{s}^r$ has to be chosen in order to align the end-effector with its goal, hence is task-specific. Since the relative pose between the camera and the end-effector is known, the definition of this reference simply implies the knowledge of the desired position of the end-effector w.r.t. the detected visual feature for the interaction process, and can be easily computed analytically using a handful of rototranslations.

The velocity control in $\mathcal{F}_{_C}$ is designed to nullify $\mathbf{e}$. An exponentially decreasing rate is imposed on $\mathbf{e}$

$$\dot{\mathbf{e}} = -\lambda \mathbf{e}, \quad \lambda > 0. \tag{D.6}$$

Thus, the interaction matrices $\mathbf{L}_{\mathbf{v}}$, $\mathbf{L}_{\boldsymbol{\omega}}$ and $\mathbf{L}_{\theta\mathbf{u}} \in \mathbb{R}^{3\times3}$ are defined such that

$$\dot{\mathbf{e}} = \begin{bmatrix} \mathbf{L}_{\mathbf{v}} & \mathbf{L}_{\boldsymbol{\omega}} \\ \mathbf{O}_3 & \mathbf{L}_{\theta\mathbf{u}} \end{bmatrix}, \begin{bmatrix} {}^{C}\mathbf{v}_{_C} \\ {}^{C}\boldsymbol{\omega}_{_C} \end{bmatrix}, \tag{D.7}$$

where ${}^{C}\mathbf{v}_{_C}$ and ${}^{C}\boldsymbol{\omega}_{_C}$ are the desired linear and angular velocities for the camera, expressed in $\mathcal{F}_{_C}$.

Then, the angular velocity control scheme is defined, as in [Chaumette, 2006]. The orientation interaction matrix $\mathbf{L}_{\theta\mathbf{u}}$ is defined as

$$\mathbf{L}_{\theta\mathbf{u}} = \mathbf{I}_3 - \frac{\theta}{2}[\mathbf{u}]_\times + \left(1 - \frac{\sin\theta}{\operatorname{sinc}^2\frac{\theta}{2}}\right)[\mathbf{u}]_\times^2, \tag{D.8}$$

where sinc is the sinus cardinal function.

Remark. Since the determinant of the above matrix is given by

$$\det \mathbf{L}_{\theta\mathbf{u}} = \frac{1}{\operatorname{sinc}^2\frac{\theta}{2}}, \tag{D.9}$$

$\mathbf{L}_{\theta\mathbf{u}}$ is singular only for $\theta = 2k\pi,\ k \neq 0$, which is out of the potential workspace, since $\theta \in [0,\ \pi[$.

Putting together Equations (D.6), (D.7) and (D.9), we have

$${}^{C}\boldsymbol{\omega}_{_C} = -\lambda \mathbf{L}_{\theta\mathbf{u}}^{-1}\theta\mathbf{u}. \tag{D.10}$$

The linear velocity control scheme, following [Chaumette, 2007], is defined define using $\mathbf{L}_{\mathbf{v}}$ and $\mathbf{L}_{\boldsymbol{\omega}}$ as

$$\mathbf{L}_{\mathbf{v}} = \frac{1}{\rho_z z_{_M}^r} \begin{bmatrix} -1 & 0 & x \\ 0 & -1 & y \\ 0 & 0 & -1 \end{bmatrix}, \tag{D.11}$$

$$\mathbf{L}_{\boldsymbol{\omega}} = \begin{bmatrix} xy & -(1+x^2) & y \\ 1+y^2 & -xy & -x \\ -y & x & 0 \end{bmatrix}, \tag{D.12}$$

where $\rho_z = z_{_M}/z^r_{_M}$, $z^r_{_M} > 0$ being the reference for $z_{_M}$, and $x, y \in \mathbb{R}$ are the normalized coordinates of the detected feature. As noted in [Chaumette, 2007], ρ_z can be obtained from a partial pose estimation scheme. It makes the HVS scheme more generic than PBVS since the estimation process is lighter. We also remark that $\mathbf{L_v}$ is singular only when $z_{_M} \to \infty$, making the inversion always feasible. Putting together Equations (D.6), (D.7), (D.11) and (D.12), we obtain

$$ {}^{C}\mathbf{v}_{_C} = -\mathbf{L}_{\mathbf{v}}^{-1} \left(\lambda \begin{bmatrix} x \\ y \\ \log z_{_M} \end{bmatrix} + \mathbf{L}_{\theta\mathbf{u}} {}^{C}\boldsymbol{\omega}_{_C} \right). \tag{D.13} $$

We can now define the desired linear and angular velocities of the AR, which are sent to the geometric controller, using the relations

$$ {}^{W}\boldsymbol{\omega}_{_B} = {}^{W}\boldsymbol{\omega}_{_C} = {}^{W}\mathbf{R}_{_C} {}^{C}\boldsymbol{\omega}_{_C}, \tag{D.14} $$

$$ {}^{W}\mathbf{v}_{_B} = {}^{W}\mathbf{R}_{_C} {}^{C}\mathbf{v}_{_C} - {}^{W}\mathbf{R}_{_B} [{}^{W}\boldsymbol{\omega}_{_B}]_{\times} {}^{W}\mathbf{p}_{_C}. \tag{D.15} $$

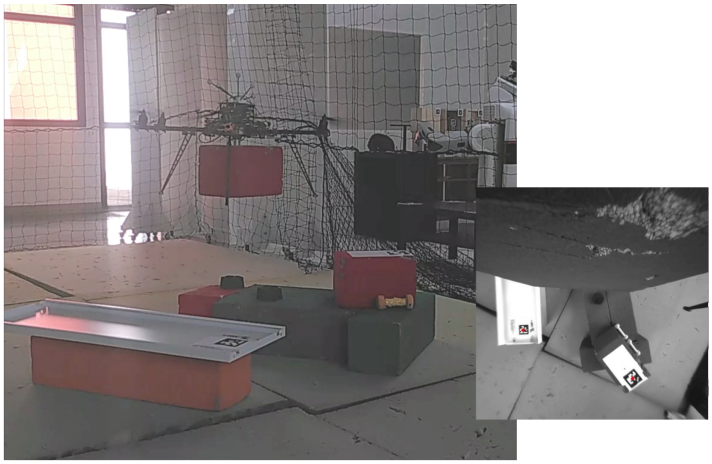
From that point on, the HVS scheme provides a reference trajectory for the body in the inertial frame. This reference drives the end-effector in contact with the object of interest, to perform the task. This task can be, e.g., picking the object, applying a contact force, pushing, maintaining contact, etc, and has to be defined in a high-level planner. However, the object is a priori unknown, thus the controller needs to be compliant w.r.t. the contact wrench applied, e.g. when lifting or pushing. Hence, the reference trajectory provided by the HVS is modulated accordingly. To this end, we make use of two tools: a Wrench Observer (WO) and an Admittance Filter (AF).

The WO proposed in [Tomić, 2017] – namely an hybrid Wrench Observer – is a combination of an acceleration-based estimator for the external forces and a momentum-based one for the external torques. This WO is well suited for flying robots since it exploits only proprioceptive sensors, which are usually available on board, as an IMU. It consists of a computation of the total body wrench based on the acceleration and angular rates retrieved from the IMU, which are compared to the nominal body wrench requested by the controller. The difference provides an estimate of the external wrench applied to the body. We note that this wrench is computed at the body CoM and expressed in $\mathcal{F}_{_B}$, while the external wrench is applied at the end-effector.

The AF is a well-known technique in the literature [Siciliano, 2009]. Its objective is to modulate a reference trajectory to compensate for external disturbances in general in the form of an externally applied wrench. The resulting output trajectory is thereby compliant w.r.t. this external wrench. Because of the fixed rigid transform between $\mathcal{F}_{_B}$ and $\mathcal{F}_{_E}$, the AF can be either applied to the end-effector or to the body CoM, in both cases making the platform's end-effector compliant. We chose to express it at the body CoM for simplicity w.r.t. the WO.

D.2.4 Experimental Results

As a practical use case, we propose to apply this control strategy in the scope of fully autonomous pick-and-place of bricks. Similar to the MBZIRC scenario, the bricks



(a) Clickable image to the video on PeerTube.

(b) QR code to the video.

Video D.1: Experiment of pick and place operation with an hexarotor.

are topped with a metal plate, allowing for magnetic gripping. The robot is tasked to pick designated bricks in a pre-defined order – e.g. computed in a prior planning step. Both the bricks and target locations are unknown. A dummy exploratory behavior is implemented to wander the workspace while looking for the brick of interest. It occurs at a predefined altitude and "scans" the whole workspace in a zigzag motion. No further mapping has been implemented, but any policy, such as the one proposed in Section D.1, could be added. As the detected coordinates are provided to the VS, the servoing starts. The resulting experiment is shown in Video D.1. Around 1 (min)43, the first placed brick flies out because of the wind generated by the propeller thrusts.

Remark. For practical reasons mainly related to the wind produced by the propellers which moves the bricks, the servoing does not bring the magnet in contact with the brick. Rather, the VS aligns the gripper right atop the metal plate, and a vertical picking is performed once the error (D.3) is sufficiently small.

The picking is automatically assessed using the WO. Then, a vertical takeoff is performed to reach the exploratory altitude, and the dummy scanning resumes until the placing location is detected. The placing is performed in the same way as the picking, and also automatically assessed with the WO. The mission goes on like this. Figure D.8 depicts the z coordinates of the robot during the pick-and-place of one brick, along with the estimated external force for the same time window. It can be observed that the observed wrench increases before the contact is established, because of the pushing force of the wind exerted by the propellers. It also explains why the observed force is not equal to 0 before the green phase. This is prominent during the placing phase, since the propellers are spinning faster to compensate for the brick weight. Therefore, the placing is assessed in a short time window. This aspect should definitely be handled in future works, e.g. using a model-based compensation of the propeller-exerted wind in the WO to improve its accuracy [Matus-Vargas, 2021].

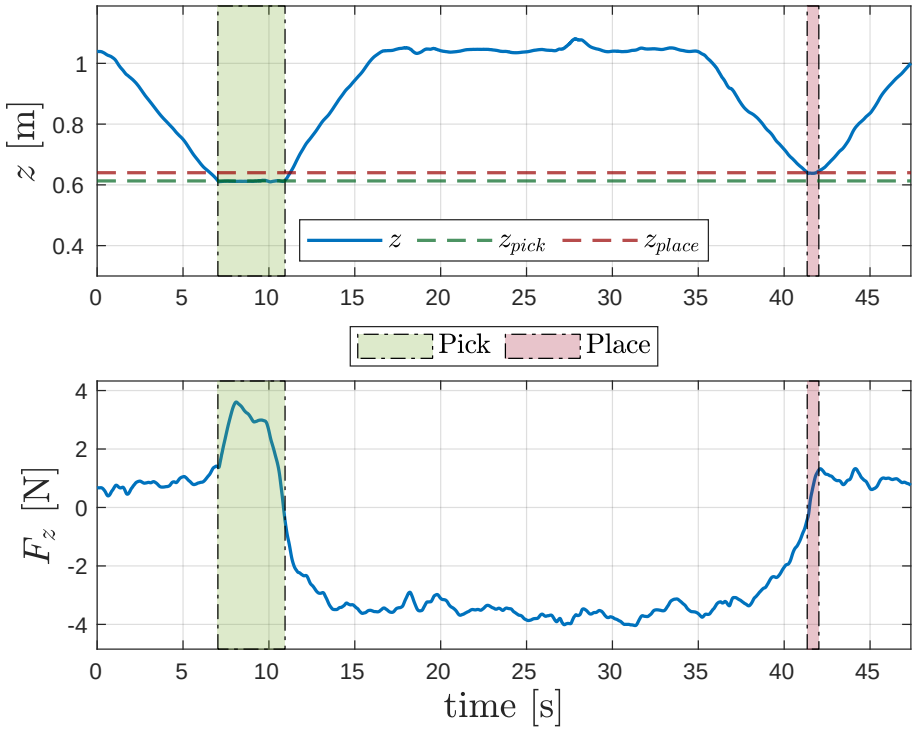


Figure D.8: The altitude of the robot in $\mathcal{F}_W$ (top) and the estimated contact force along $\mathbf{z}_B$ (bottom).

D.2.5 Conclusion

This appendix presents a general control architecture for autonomous physical interaction tasks tailored for fully-actuated ARs. We employ a physical-interaction paradigm called *the Flying End-Effector*, by adopting, in particular, a position/attitude decoupled controller to exploit the 6D motion capability of the fully-actuated platform. To make the architecture autonomous, the environment is monitored using a camera; thus, we implement a classical vision-based trajectory generator called Hybrid Visual Servoing. Finally, the physical interaction is handled by using an onboard model-based Wrench Observer in order to autonomously react to the physical contact, and the generated trajectory is filtered through an Admittance Filter to achieve compliancy. This control architecture is employed, as a proof of concept, in a vision-driven autonomous pick-and-place scenario. The experimental results are provided to demonstrate the coherence and efficiency of the framework.

Bibliography

[Aker, 2017] Cemal Aker and Sinan Kalkan. "Using deep networks for drone detection". In: *14th IEEE Int. Conf. on Advanced Video and Signal Based Surveillance*. 2017 (cit. on p. 109).

[Alcántara, 2020] Alfonso Alcántara, Jesús Capitán, Arturo Torres-González, Rita Cunha, and Aníbal Ollero. "Autonomous execution of cinematographic shots with multiple drones". In: *IEEE Access* 8 (2020), pp. 201300–201316 (cit. on pp. 6, 9, 21).

[Alexis, 2016] Kostas Alexis, Christos Papachristos, Roland Siegwart, and Anthony Tzes. "Robust model predictive flight control of unmanned rotorcrafts". In: *Journal of Intelligent & Robotics Systems* 81.3 (2016), pp. 443–469 (cit. on p. 59).

[Allibert, 2010] Guillaume Allibert, Estelle Courtial, and François Chaumette. "Predictive control for constrained image-based visual servoing". In: *IEEE Trans. on Robotics* 26.5 (2010), pp. 933–939 (cit. on p. 72).

[Anderson, 2020] Jeffery Ryan Anderson and Beshah Ayalew. "A cascaded optimization approach for modeling a professional driver's driving style". In: *ASME Journal on Dynamic Systems, Measurement, and Control* 142.9 (2020) (cit. on p. 51).

[Andersson, 2019] Joel A.E. Andersson, Joris Gillis, Greg Horn, James B. Rawlings, and Moritz Diehl. "CasADi: a software framework for nonlinear optimization and optimal control". In: *Mathematical Programming Computation* 11.1 (2019), pp. 1–36 (cit. on pp. 147, 153).

[Atanasov, 2014a] Nikolay Atanasov, Roberto Tron, Victor M Preciado, and George J Pappas. "Joint estimation and localization in sensor networks". In: *53rd IEEE Conf. on Decision and Control*. 2014, pp. 6875–6882 (cit. on pp. 117, 136).

[Atanasov, 2014b] Nikolay Atanasov, Jerome Le Ny, Kostas Daniilidis, and George J Pappas. "Information acquisition with sensing robots: Algorithms and error bounds". In: *2014 IEEE Int. Conf. on Robotics and Automation*. 2014, pp. 6447–6454 (cit. on pp. 118, 135).

[Atanasov, 2015a] Nikolay Atanasov, Jerome Le Ny, Kostas Daniilidis, and George J Pappas. "Decentralized active information acquisition: Theory and application to multi-robot SLAM". In: *2015 IEEE Int. Conf. on Robotics and Automation*. 2015, pp. 4775–4782 (cit. on pp. 114, 117).

[Atanasov, 2015b] Nikolay Asenov Atanasov. "Active information acquisition with mobile robots". PhD thesis. University of Pennsylvania, 2015 (cit. on pp. 117–120).

[Athans, 1972] Michael Athans. "On the determination of optimal costly measurement strategies for linear stochastic systems". In: *Automatica* 8.4 (1972), pp. 397–412 (cit. on pp. 116, 118).

[Augugliaro, 2012] Federico Augugliaro, Angela P Schoellig, and Raffaello D'Andrea. "Generation of collision-free trajectories for a quadrocopter fleet: A sequential convex programming approach". In: *2012 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. 2012, pp. 1917–1922 (cit. on p. 111).

[Augugliaro, 2014] Frederico Augugliaro, Sergei Lupashin, Michael Hamer, Cason Male, Markus Hehn, Mark W. Mueller, Jan Sebastian Willmann, Fabio Gramazio, Matthias Kohler, and Raffaello D'Andrea. "The Flight Assembled Architecture installation: Cooperative construction with flying machines". In: *IEEE Control Systems Magazine* 34.4 (2014), pp. 46–64 (cit. on p. 166).

[Azzabi, 2017] Ameni Azzabi and Khaled Nouri. "Path planning for autonomous mobile robot using the Potential Field method". In: *2017 IEEE Int. Conf. on Advanced Systems and Electric Technologies*. 2017, pp. 389–394 (cit. on p. 111).

[Baca, 2016] Tomas Baca, Giuseppe Loianno, and Martin Saska. "Embedded model predictive control of unmanned micro aerial vehicles". In: *21st IEEE Int. Conf. on methods and models in automation and robotics*. 2016, pp. 992–997 (cit. on pp. 57, 111).

[Badr, 2016] Sherif Badr, Omar Mehrez, and Abd Elnaby Kabeel. "A novel modification for a quadrotor design". In: *2016 Int. Conf. on Unmanned Aircraft Systems*. 2016, pp. 702–710 (cit. on p. 18).

[Bailon-Ruiz, 2022] Rafael Bailon-Ruiz, Arthur Bit-Monnot, and Simon Lacroix. "Real-time wildfire monitoring with a fleet of UAVs". In: *Robotics and Autonomous Systems* 152 (2022), p. 104071 (cit. on p. 9).

[Baizid, 2017] Khelifa Baizid, Gerardo Giglio, Francesco Pierri, Miguel Angel Trujillo, Gianluca Antonelli, Fabrizio Caccavale, Antidio Viguria, Stefano Chiaverini, and Aníbal Ollero. "Behavioral control of unmanned aerial vehicle manipulator systems". In: *Autonomous Robots* 41.5 (2017), pp. 1203–1220 (cit. on p. 166).

[Balaram, 2021] J. Balaram, MiMi Aung, and Matthew P. Golombek. "The ingenuity helicopter on the perseverance rover". In: *Space Science Reviews* 217.4 (2021) (cit. on p. 6).

[Bar-Shalom, 1988] Yaakov Bar-Shalom, Thomas E. Fortmann, and Peter G. Cable. *Tracking and data association*. Acoustical Society of America, 1988 (cit. on p. 119).

[Barros Carlos, 2021] Bárbara Barros Carlos, Antonio Franchi, and Giuseppe Oriolo. "Towards Safe Human-Quadrotor Interaction: Mixed-Initiative Control via Real-Time NMPC". In: *IEEE Robotics and Automation Letters* 6.4 (2021), pp. 7611–7618 (cit. on pp. 30, 152).

[Baskaya, 2021] Elgiz Baskaya, Mahmoud Hamandi, Murat Bronz, and Antonio Franchi. "A Novel Robust Hexarotor Capable of Static Hovering in Presence of Propeller Failure". In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 4001–4008 (cit. on p. 17).

[Bates, 1980] Douglas M. Bates and Donald G. Watts. "Relative curvature measures of nonlinearity". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 42.1 (1980), pp. 1–16 (cit. on p. 29).

[Beder, 2006] Christian Beder and Richard Steffen. "Determining an initial image pair for fixing the scale of a 3d reconstruction from an image sequence". In: *Joint Pattern Recognition Symposium*. 2006, pp. 657–666 (cit. on pp. 123, 131).

[Bemporad, 2009] Alberto Bemporad, Carlo A Pascucci, and Claudio Rocchi. "Hierarchical and hybrid model predictive control of quadcopter air vehicles". In: *IFAC Proceedings Volumes* 42.17 (2009), pp. 14–19 (cit. on p. 45).

[Bertsekas, 2012] Dimitri Bertsekas. *Dynamic programming and optimal control*. Vol. 1. Athena Scientific, 2012 (cit. on p. 26).

[Bicego, 2020] Davide Bicego, Jacopo Mazzetto, Ruggero Carli, Marcello Farina, and Antonio Franchi. "Nonlinear model predictive control with enhanced actuator model for multi-rotor aerial vehicles with generic designs". In: *Journal of Intelligent & Robotics Systems* 100.3 (2020), pp. 1213–1247 (cit. on pp. 18, 19, 28, 31, 33, 42, 57, 58, 60–62, 143).

[Bloesch, 2015] Michael Bloesch, Sammy Omari, Marco Hutter, and Roland Siegwart. "Robust visual inertial odometry using a direct EKF-based approach". In: *2015 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. 2015, pp. 298–304 (cit. on pp. 21, 90, 93).

[Bock, 1984] Hans Georg Bock and Karl-Josef Plitt. "A multiple shooting algorithm for direct solution of optimal control problems". In: *IFAC Proceedings Volumes* 17.2 (1984), pp. 1603–1608 (cit. on p. 27).

[Bodie, 2021] Karen Bodie, Maximilian Brunner, Michael Pantic, Stefan Walser, Patrick Pfändler, Ueli Angst, Roland Siegwart, and Juan Nieto. "Active Interaction Force Control for Contact-Based Inspection With a Fully Actuated Aerial Vehicle". In: *IEEE Trans. on Robotics* 37.3 (2021), pp. 709–722 (cit. on p. 166).

[Boeuf, 2015] Alexandre Boeuf, Juan Cortés, Rachid Alami, and Thierry Siméon. "Enhancing sampling-based kinodynamic motion planning for quadrotors". In: *2015 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. 2015, pp. 2447–2452 (cit. on p. 146).

[Bourgault, 2004] Frederic Bourgault, Tomonari Furukawa, and Hugh F. Durrant-Whyte. "Process model, constraints, and the coordinated search strategy". In: *2004 IEEE Int. Conf. on Robotics and Automation*. Vol. 5. 2004, pp. 5256–5261 (cit. on p. 117).

[Boyd, 2004] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004 (cit. on p. 26).

[Bradski, 2000] G. Bradski. "The OpenCV Library". In: *Dr. Dobb's Journal of Software Tools* (2000) (cit. on pp. 23, 48, 149).

[Brescianini, 2016] Dario Brescianini and Raffaello D'Andrea. "Design, modeling and control of an omni-directional aerial vehicle". In: *2016 IEEE Int. Conf. on Robotics and Automation*. 2016, pp. 3261–3266 (cit. on p. 45).

[Bury, 2019] Diane Bury, Jean-Baptiste Izard, Marc Gouttefarde, and Florent Lamiraux. "Continuous collision detection for a robotic arm mounted on a cable-driven parallel robot". In: *2019 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. 2019, pp. 8097–8102 (cit. on p. 111).

[Cai, 2021] Xiaoyi Cai, Brent Schlotfeldt, Kasra Khosoussi, Nikolay Atanasov, George J. Pappas, and Jonathan P. How. "Non-Monotone Energy-Aware Information Gathering for Heterogeneous Robot Teams". In: *2021 IEEE Int. Conf. on Robotics and Automation*. 2021, pp. 8859–8865 (cit. on p. 119).

[Campos, 2021] Carlos Campos, Richard Elvira, Juan J. Gómez Rodríguez, José MM Montiel, and Juan D. Tardós. "ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM". In: *IEEE Trans. on Robotics* 37.6 (2021), pp. 1874–1890 (cit. on pp. 21, 104).

[Canny, 1986] John Canny. "A computational approach to edge detection". In: *IEEE Trans. on Pattern Analysis & Machine Intelligence* 8.6 (1986), pp. 679–698 (cit. on p. 164).

[Carino, 2015] Jossué Carino, Hernan Abaunza, and P. Castillo. "Quadrotor quaternion control". In: *2015 Int. Conf. on Unmanned Aircraft Systems*. 2015, pp. 825–831 (cit. on p. 41).

[Carpentier, 2019] Justin Carpentier, Guilhem Saurel, Gabriele Buondonno, Joseph Mirabel, Florent Lamiraux, Olivier Stasse, and Nicolas Mansard. "The Pinocchio C++ library – A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives". In: *2019 Int. Symp. on System Integrations*. 2019 (cit. on pp. 61, 153).

[Carrillo, 2015] Henry Carrillo, Philip Dames, Vijay Kumar, and José A Castellanos. "Autonomous robotic exploration using occupancy grid maps and graph slam based on shannon and rényi entropy". In: *2015 IEEE Int. Conf. on Robotics and Automation*. 2015, pp. 487–494 (cit. on p. 119).

[Chakravarthy, 1998] Animesh Chakravarthy and Debasish Ghose. "Obstacle avoidance in a dynamic environment: A collision cone approach". In: *IEEE Trans. on Systems, Man, & Cybernetics. Part A: Systems & Humans* 28.5 (1998), pp. 562–574 (cit. on p. 110).

[Charrow, 2014] Benjamin Charrow, Vijay Kumar, and Nathan Michael. "Approximate representations for multi-robot control policies that maximize mutual information". In: *Autonomous Robots* 37.4 (2014), pp. 383–400 (cit. on p. 117).

[Chaumette, 1998] François Chaumette. "Potential problems of stability and convergence in image-based and position-based visual servoing". In: *The confluence of vision and control*. Ed. by David J. Kriegman, Gregory D. Hager, and A. Stephen Morse. Springer, 1998, pp. 66–78 (cit. on p. 24).

[Chaumette, 2006] François Chaumette and Seth Hutchinson. "Visual servo control Part I: Basic approaches". In: *IEEE Robotics & Automation Magazine* 13.4 (2006), pp. 82–90 (cit. on p. 168).

[Chaumette, 2007] François Chaumette and Seth Hutchinson. "Visual servo control Part II: Advanced approaches". In: *IEEE Robotics & Automation Magazine* 14.1 (2007), pp. 109–118 (cit. on pp. 24, 167–169).

[Chen, 2016] Jing Chen, Tianbo Liu, and Shaojie Shen. "Tracking a moving target in cluttered environments using a quadrotor". In: *2016 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. 2016, pp. 446–453 (cit. on pp. 117, 120).

[Chen, 2017] Yutao Chen, Davide Cuccato, Mattia Bruschetta, and Alessandro Beghi. "A fast nonlinear model predictive control strategy for real-time motion control of mechanical systems". In: *2017 IEEE/ASME Int. Conf. on Advanced Intelligent Mechatronics*. 2017, pp. 1780–1785 (cit. on pp. 29, 152).

[Chen, 2019] Yutao Chen, Mattia Bruschetta, Enrico Picotti, and Alessandro Beghi. "MATMPC - A MATLAB Based Toolbox for Real-time Nonlinear Model Predictive Control". In: *18th European Control Conference*. 2019, pp. 3365–3370 (cit. on pp. 147, 152).

[Chirikjian, 2011] Gregory S. Chirikjian. *Stochastic models, information theory, and Lie groups, volume 2: Analytic methods and modern applications*. Vol. 2. Springer Science & Business Media, 2011 (cit. on p. 39).

[Chng, 2020] Shin-Fang Ch'ng, Naoya Sogi, Pulak Purkait, Tat-Jun Chin, and Kazuhiro Fukui. "Resolving marker pose ambiguity by robust rotation averaging with clique constraints". In: *2020 IEEE Int. Conf. on Robotics and Automation*. 2020, pp. 9680–9686 (cit. on p. 23).

[Choi, 2020] Daegyun Choi, Kyuman Lee, and Donghoon Kim. "Enhanced potential field-based collision avoidance for unmanned aerial vehicles in a dynamic environment". In: *AIAA Scitech 2020 Forum*. 2020 (cit. on p. 111).

[Chung, 2004] Timothy H. Chung, Vijay Gupta, Joel W. Burdick, and Richard M. Murray. "On a decentralized active sensing strategy using mobile sensor platforms in a network". In: *43rd IEEE Conf. on Decision and Control*. Vol. 2. 2004, pp. 1914–1919 (cit. on pp. 84, 122, 123, 130, 135).

[Chung, 2006] Timothy H Chung, Joel W Burdick, and Richard M Murray. "A decentralized motion coordination strategy for dynamic target tracking". In: *2006 IEEE Int. Conf. on Robotics and Automation*. 2006, pp. 2416–2422 (cit. on pp. 118, 119).

[Ciarfuglia, 2014] Thomas A. Ciarfuglia, Francesco Crocetti, Antonio Ficola, and Paolo Valigi. "A preliminary experimental analysis of V-tail quad-rotor dynamics". In: *2014 IEEE Int. Conf. on Modelling, Identification & Control*. 2014, pp. 277–282 (cit. on p. 18).

[Collins, 2014] Toby Collins and Adrien Bartoli. "Infinitesimal plane-based pose estimation". In: *Int. Journ. of Computer Vision* 109.3 (2014), pp. 252–286 (cit. on pp. 22, 23).

[Conticelli, 1999] Fabio Conticelli, Allotta Benedetto, and Carlo Colombo. "Hybrid visual servoing: A combination of nonlinear control and linear vision". In: *Robotics and Autonomous Systems* 29.4 (1999), pp. 243–256 (cit. on p. 24).

[Corke, 2001] Peter I. Corke and Seth Hutchinson. "A new partitioned approach to image-based visual servo control". In: *IEEE Trans. on Robotics and Automation* 17.4 (2001), pp. 507–515 (cit. on p. 24).

[Corsini, 2021] Gianluca Corsini, Martin Jacquet, Antonio Enrique Jimenez-Cano, Amr Afifi, Daniel Sidobre, and Antonio Franchi. "A General Control Architecture for Visual Servoing and Physical Interaction Tasks for Fully-actuated Aerial Vehicles". In: *1st Work. on Aerial Robotic Systems Physically Interacting with the Environment*. 2021 (cit. on pp. 14, 166).

[Corsini, 2022] Gianluca Corsini, Martin Jacquet, Hemjyoti Das, Amr Afifi, Daniel Sidobre, and Antonio Franchi. "Nonlinear Model Predictive Control for Human-Robot Handover with Application to the Aerial Case". In: *2022 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. 2022 (cit. on pp. 14, 66).

[Cortes, 2004] Jorge Cortes, Sonia Martinez, Timur Karatas, and Francesco Bullo. "Coverage control for mobile sensing networks". In: *IEEE Trans. on Robotics and Automation* 20.2 (2004), pp. 243–255 (cit. on p. 117).

[Crassidis, 2003] John L Crassidis and F Landis Markley. "Unscented filtering for spacecraft attitude estimation". In: *Journal of guidance, control, and dynamics* 26.4 (2003), pp. 536–542 (cit. on p. 147).

[DAlfonso, 2013] Luigi D'Alfonso, Emanuele Garone, Pietro Muraca, and Paolo Pugliese. "P3P and P2P Problems with known camera and object vertical directions". In: *21st Mediterranean Conf. on Control and Automation*. 2013, pp. 444–451 (cit. on p. 22).

[Dames, 2017] Philip Dames, Pratap Tokekar, and Vijay Kumar. "Detecting, localizing, and tracking an unknown number of moving targets using a team of mobile robots". In: *The International Journal of Robotics Research* 36.13-14 (2017), pp. 1540–1553 (cit. on p. 135).

[Dames, 2020] Philip M. Dames. "Distributed multi-target search and tracking using the PHD filter". In: *Autonomous Robots* 44.3 (2020), pp. 673–689 (cit. on pp. 104, 117, 136).

[Dantec, 2019] Ewen Dantec. "Reconnaissance et localisation d'objets par vision embarquée sur drone pour des tâches de prise et de dépose". MA thesis. ISAE-Supaéro, 2019 (cit. on pp. 163, 164).

[Dantec, 2021] Ewen Dantec et al. "Whole body model predictive control with a memory of motion: Experiments on a torque-controlled talos". In: *2021 IEEE Int. Conf. on Robotics and Automation*. 2021, pp. 8202–8208 (cit. on pp. 29, 153).

[Darivianakis, 2014] Georgios Darivianakis, Kostas Alexis, Michael Burri, and Roland Siegwart. "Hybrid predictive control for aerial robotic physical interaction towards inspection operations". In: *2014 IEEE Int. Conf. on Robotics and Automation*. 2014, pp. 53–58 (cit. on pp. 7, 57, 59).

[Davison, 2003] Andrew J. Davison. "Real-time simultaneous localisation and mapping with a single camera". In: *9th IEEE Int. Conf. on Computer Vision*. Vol. 3. 2003, pp. 1403–1403 (cit. on p. 90).

[Delmerico, 2018] Jeffrey Delmerico and Davide Scaramuzza. "A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots". In: *2018 IEEE Int. Conf. on Robotics and Automation*. 2018, pp. 2502–2509 (cit. on pp. 21, 104).

[Diebel, 2006] James Diebel. *Representing attitude: Euler angles, unit quaternions, and rotation vectors*. Tech. rep. 2006 (cit. on p. 40).

[Diehl, 2002] Moritz Diehl, H Georg Bock, Johannes P. Schlöder, Rolf Findeisen, Zoltan Nagy, and Frank Allgöwer. "Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations". In: *Journal of Process Control* 12.4 (2002), pp. 577–585 (cit. on p. 29).

[Diehl, 2005] Moritz Diehl, Hans Georg Bock, and Johannes P. Schlöder. "A real-time iteration scheme for nonlinear optimization in optimal feedback control". In: *SIAM Journal on Control and Optimization* 43.5 (2005), pp. 1714–1736 (cit. on p. 29).

[Duda, 1972] Richard O. Duda and Peter E. Hart. "Use of the Hough transformation to detect lines and curves in pictures". In: *Communications of the ACM* 15.1 (1972), pp. 11–15 (cit. on p. 164).

[Eidenberger, 2010] Robert Eidenberger and Josef Scharinger. "Active perception and scene modeling by planning with probabilistic 6d object poses". In: *2010 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. 2010, pp. 1036–1043 (cit. on p. 117).

[Engel, 2012] Jakob Engel, Jürgen Sturm, and Daniel Cremers. "Camera-based navigation of a low-cost quadrocopter". In: *2012 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. 2012, pp. 2815–2821 (cit. on p. 90).

[Falanga, 2017] Davide Falanga, Elias Mueggler, Matthias Faessler, and Davide Scaramuzza. "Aggressive quadrotor flight through narrow gaps with onboard sensing and computing using active vision". In: *2017 IEEE Int. Conf. on Robotics and Automation*. 2017, pp. 5774–5781 (cit. on pp. 17, 25).

[Falanga, 2018] Davide Falanga, Philipp Foehn, Peng Lu, and Davide Scaramuzza. "PAMPC: Perception-Aware Model Predictive Control for Quadrotors". In: *2018 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. 2018 (cit. on pp. 25, 32, 33, 57, 68, 89, 96, 104, 152).

[Feron, 2008] Eric Feron and Eric N. Johnson. "Aerial Robotics". In: *Springer Handbook of Robotics*. Ed. by Bruno Siciliano and Oussama Khatib. Springer, 2008, pp. 1009–1029 (cit. on pp. 4, 5, 7).

[Ferreau, 2014] Hans Joachim Ferreau, Christian Kirches, Andreas Potschka, Hans Georg Bock, and Moritz Diehl. "qpOASES: A parametric active-set algorithm for quadratic programming". In: *Mathematical Programming Computation* 6.4 (2014), pp. 327–363 (cit. on p. 152).

[Fliess, 1995] Michel Fliess, Jean Lévine, Philippe Martin, and Pierre Rouchon. "Flatness and defect of non-linear systems: introductory theory and examples". In: *International Journal of Control* 61.6 (1995), pp. 1327–1361 (cit. on p. 16).

[Foehn, 2018] Philipp Foehn and Davide Scaramuzza. "Onboard state dependent LQR for agile quadrotors". In: *2018 IEEE Int. Conf. on Robotics and Automation*. 2018, pp. 6566–6572 (cit. on p. 59).

[Foehn, 2021] Philipp Foehn, Angel Romero, and Davide Scaramuzza. "Time-optimal planning for quadrotor waypoint flight". In: *Science Robotics* 6.56 (2021) (cit. on p. 28).

[Fong, 2008] WT Fong, SK Ong, and AYC Nee. "Methods for in-field user calibration of an inertial measurement unit without external equipment". In: *Measurement Science and technology* 19.8 (2008) (cit. on p. 90).

[Fourmy, 2019] Mederic Fourmy, Dinesh Atchuthan, Nicolas Mansard, Joan Solà, and Thomas Flayols. "Absolute humanoid localization and mapping based on IMU Lie group and fiducial markers". In: *19th IEEE/RAS Int. Conf. on Humanoid Robots*. 2019, pp. 237–243 (cit. on pp. 48, 49).

[Fourmy, 2022] Mederic Fourmy. "State estimation and localization of legged robots: a tightly-coupled approach based on a-posteriori maximization". PhD thesis. Institut national des sciences appliquées de Toulouse, 2022 (cit. on p. 50).

[Franchi, 2012] Antonio Franchi, Carlo Masone, Volker Grabe, Markus Ryll, Heinrich H Bülthoff, and Paolo Giordano. "Modeling and control of UAV bearing formations with bilateral high-level steering". In: *The International Journal of Robotics Research* 31.12 (2012), pp. 1504–1525 (cit. on p. 109).

[Franchi, 2018] Antonio Franchi, Ruggero Carli, Davide Bicego, and Markus Ryll. "Full-pose tracking control for aerial robotic systems with laterally bounded input force". In: *IEEE Trans. on Robotics* 34.2 (2018), pp. 534–541 (cit. on pp. 18, 45, 167).

[Fresk, 2013] Emil Fresk and George Nikolakopoulos. "Full quaternion based attitude control for a quadrotor". In: *2013 European Control Conference*. 2013, pp. 3864–3869 (cit. on p. 41).

[Frison, 2020] Gianluca Frison and Moritz Diehl. "HPIPM: a high-performance quadratic programming framework for model predictive control". In: *IFAC-PapersOnLine* 53.2 (2020), pp. 6563–6569 (cit. on p. 153).

[Furgale, 2013] Paul Furgale, Joern Rehder, and Roland Siegwart. "Unified temporal and spatial calibration for multi-sensor systems". In: *2013 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. 2013, pp. 1280–1286 (cit. on p. 23).

[Gao, 2003] Xiao-Shan Gao, Xiao-Rong Hou, Jianliang Tang, and Hang-Fei Cheng. "Complete solution classification for the perspective-three-point problem". In: *IEEE Trans. on Pattern Analysis & Machine Intelligence* 25.8 (2003), pp. 930–943 (cit. on p. 22).

[Garrido-Jurado, 2014] Sergio Garrido-Jurado, Rafael Muñoz-Salinas, Francisco José Madrid-Cuevas, and Manuel Jesús Marín-Jiménez. "Automatic generation and detection of highly reliable fiducial markers under occlusion". In: *Pattern Recognition* 47.6 (2014), pp. 2280–2292 (cit. on p. 48).

[Geisert, 2016] Mathieu Geisert and Nicolas Mansard. "Trajectory generation for quadrotor based systems using numerical optimal control". In: *2016 IEEE Int. Conf. on Robotics and Automation*. 2016, pp. 2958–2964 (cit. on pp. 45, 59).

[Giftthaler, 2018] Markus Giftthaler, Michael Neunert, Markus Stäuble, and Jonas Buchli. "The control toolbox – An open-source C++ library for robotics, optimal and model predictive control". In: *2018 Int. Conf. on Simulation, Modeling, and Programming for Autonomous Robots*. 2018, pp. 123–129 (cit. on p. 152).

[Greeff, 2020] Melissa Greeff, Timothy D Barfoot, and Angela P. Schoellig. "A perception-aware flatness-based model predictive controller for fast vision-based multirotor flight". In: *IFAC-PapersOnLine* 53.2 (2020), pp. 9412–9419 (cit. on pp. 33, 89).

[Grimm, 2005] Gene Grimm, Michael J. Messina, Sezai Emre Tuna, and Andrew R. Teel. "Model predictive control: for want of a local control Lyapunov function, all is not lost". In: *IEEE Trans. on Automatic Control* 50.5 (2005), pp. 546–558 (cit. on pp. 28, 142).

[Grocholsky, 2002] Ben Grocholsky. "Information-theoretic control of multiple sensor platforms". PhD thesis. University of Sydney, 2002 (cit. on p. 119).

[Gros, 2020] Sébastien Gros, Mario Zanon, Rien Quirynen, Alberto Bemporad, and Moritz Diehl. "From linear to nonlinear MPC: bridging the gap via the real-time iteration". In: *International Journal of Control* 93.1 (2020), pp. 62–80 (cit. on p. 29).

[Grüne, 2010] Lars Grüne, Jürgen Pannek, Martin Seehafer, and Karl Worthmann. "Analysis of unconstrained nonlinear MPC schemes with time varying control horizon". In: *SIAM Journal on Control and Optimization* 48.8 (2010), pp. 4938–4962 (cit. on p. 28).

[Hamandi, 2020] Mahmoud Hamandi, Kapil Sawant, Marco Tognon, and Antonio Franchi. "Omni-plus-seven (O7+): An omnidirectional aerial prototype with a minimal number of unidirectional thrusters". In: *2020 Int. Conf. on Unmanned Aircraft Systems*. 2020, pp. 754–761 (cit. on pp. 9, 17, 18).

[Hamandi, 2021] Mahmoud Hamandi, Federico Usai, Quentin Sablé, Nicolas Staub, Marco Tognon, and Antonio Franchi. "Design of multirotor aerial vehicles: A taxonomy based on input allocation". In: *The International Journal of Robotics Research* 40.8-9 (2021), pp. 1015–1044 (cit. on p. 16).

[Hattenberger, 2022] Gautier Hattenberger, Titouan Verdu, Nicolas Maury, Pierre Narvor, Fleur Couvreux, Murat Bronz, Simon Lacroix, Grègoire Cayez, and Gregory C. Roberts. "Field report: deployment of a fleet of drones for cloud exploration". In: *Int. Journ. of Micro Air Vehicles* 14 (2022) (cit. on pp. 6, 9).

[He, 2021] Dongjiao He, Wei Xu, and Fu Zhang. "Kalman Filters on Differentiable Manifolds". 2021 (cit. on p. 92).

[Houska, 2011] Boris Houska, Hans Joachi Ferreau, and Moritz Diehl. "ACADO Toolkit – An Open Source Framework for Automatic Control and Dynamic Optimization". In: *Optimal Control Applications and Methods* 32.3 (2011), pp. 298–312 (cit. on p. 152).

[Hua, 2015] Minh-Duc Hua, Tarek Hamel, Pascal Morin, and Claude Samson. "Control of VTOL vehicles with thrust-tilting augmentation". In: *Automatica* 52 (2015), pp. 1–7 (cit. on pp. 9, 17, 18).

[Huang, 2019] Sunan Huang, Rodney Swee Huat Teo, and Kok Kiong Tan. "Collision avoidance of multi unmanned aerial vehicles: A review". In: *Annual Reviews in Control* 48 (2019), pp. 147–164 (cit. on p. 110).

[Huynh, 2009] Du Q. Huynh. "Metrics for 3D rotations: Comparison and analysis". In: *Journal of Mathematical Imaging and Vision* 35.2 (2009), pp. 155–164 (cit. on pp. 41, 60).

[Jacquet, 2020] Martin Jacquet, Gianluca Corsini, Davide Bicego, and Antonio Franchi. "Perception-constrained and Motor-level Nonlinear MPC for both Underactuated and Tilted-propeller UAVs". In: *2020 IEEE Int. Conf. on Robotics and Automation*. 2020, pp. 4301–4306 (cit. on pp. 14, 66).

[Jacquet, 2021] Martin Jacquet and Antonio Franchi. "Motor and Perception Constrained NMPC for Torque-Controlled Generic Aerial Vehicles". In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 518–525 (cit. on pp. 14, 66).

[Jacquet, 2022a] Martin Jacquet, Max Kivits, Hemjyoti Das, and Antonio Franchi. "Motor-level N-MPC for Cooperative Active Perception with Multiple Heterogeneous UAVs". In: *IEEE Robotics and Automation Letters* 7.2 (Apr. 2022), pp. 2063–2070 (cit. on pp. 14, 116).

[Jacquet, 2022b] Martin Jacquet and Antonio Franchi. "Enforcing Vision-Based Localization using Perception Constrained N-MPC for Multi-Rotor Aerial Vehicles". In: *2022 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. Oct. 2022 (cit. on pp. 14, 88).

[Jang, 2020] Dohyun Jang, Jaehyun Yoo, Clark Youngdong Son, Dabin Kim, and H. Jin Kim. "Multi-robot active sensing and environmental model learning with distributed Gaussian process". In: *IEEE Robotics and Automation Letters* 5.4 (2020), pp. 5905–5912 (cit. on p. 117).

[Jiang, 2018] Guangying Jiang, Richard M Voyles, and Jae Jung Choi. "Precision Fully-Actuated UAV for Visual and Physical Inspection of Structures for Nuclear Decommissioning and Search and Rescue". In: *2018 IEEE Int. Symp. on Safety, Security and Rescue Robotics*. 2018, pp. 1–7 (cit. on p. 166).

[Joshi, 2008] Siddharth Joshi and Stephen Boyd. "Sensor selection via convex optimization". In: *IEEE Trans. on Signal Processing* 57.2 (2008), pp. 451–462 (cit. on p. 117).

[Julier, 1997] Simon J Julier and Jeffrey K Uhlmann. "New extension of the Kalman filter to nonlinear systems". In: *Signal processing, sensor fusion, and target recognition VI*. Vol. 3068. 1997, pp. 182–193 (cit. on p. 146).

[Kamel, 2015] Mina Kamel, Kostas Alexis, Markus Achtelik, and Roland Siegwart. "Fast nonlinear model predictive control for multi-copter attitude tracking on so (3)". In: *2015 IEEE Conf. on Control Applications*. 2015, pp. 1160–1166 (cit. on pp. 29, 31).

[Kamel, 2017] Mina Kamel, Michael Burri, and Roland Siegwart. "Linear vs nonlinear MPC for trajectory tracking applied to rotary wing micro aerial vehicles". In: *IFAC-PapersOnLine* 50.1 (2017), pp. 3463–3469 (cit. on pp. 28, 152).

[Kataoka, 2011] Yasuyuki Kataoka, Kazuma Sekiguchi, and Mitsuji Sampei. "Nonlinear control and model analysis of trirotor UAV model". In: *IFAC Proceedings Volumes* 44.1 (2011), pp. 10391–10396 (cit. on p. 16).

[Khatib, 1987] Oussama Khatib. "A unified approach for motion and force control of robot manipulators: The operational space formulation". In: *IEEE Journal on Robotics and Automation* 3.1 (1987), pp. 43–53 (cit. on p. 30).

[Kiefer, 1974] Jack Kiefer. "General equivalence theory for optimum designs (approximate theory)". In: *The annals of Statistics* 2.5 (1974), pp. 849–879 (cit. on p. 119).

[Kim, 2013] Suseong Kim, Seungwon Choi, and H Jin Kim. "Aerial manipulation using a quadrotor with a two DOF robotic arm". In: *2013 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. 2013, pp. 4990–4995 (cit. on p. 166).

[Kim, 2021] Kyunam Kim, Patrick Spieler, Elena-Sorina Lupu, Alireza Ramezani, and Soon-Jo Chung. "A bipedal walking robot that can fly, slackline, and skateboard". In: *Science Robotics* 6.59 (2021) (cit. on p. 8).

[Kostadinov, 2020] Dimche Kostadinov and Davide Scaramuzza. "Online weight-adaptive nonlinear model predictive control". In: *2020 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. 2020, pp. 1180–1185 (cit. on p. 69).

[Kragic, 2002] Danica Kragic and Henrik I. Christensen. *Survey on Visual Servoing for Manipulation*. Tech. rep. 2002 (cit. on p. 24).

[Krajník, 2014] Tomáš Krajník, Matías Nitsche, Jan Faigl, Petr Vaněk, Martin Saska, Libor Přeučil, Tom Duckett, and Marta Mejail. "A practical multirobot localization system". In: *Journal of Intelligent & Robotics Systems* 76.3 (2014), pp. 539–562 (cit. on pp. 48, 109).

[Kueng, 2016] Beat Kueng, Elias Mueggler, Guillermo Gallego, and Davide Scaramuzza. "Low-latency visual odometry using event-based feature tracks". In: *2016 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. 2016, pp. 16–23 (cit. on p. 20).

[Kuffner, 2004] James J. Kuffner. "Effective sampling and distance metrics for 3D rigid body path planning". In: *2004 IEEE Int. Conf. on Robotics and Automation*. Vol. 4. 2004, pp. 3993–3998 (cit. on p. 61).

[Kuipers, 1999] Jack B. Kuipers. *Quaternions and rotation sequences: a primer with applications to orbits, aerospace, and virtual reality*. Princeton university press, 1999 (cit. on p. 40).

[Kumar, 2004] Vijay Kumar, Daniela Rus, and Sanjiv Singh. "Robot and sensor networks for first responders". In: *IEEE Pervasive computing* 3.4 (2004), pp. 24–33 (cit. on p. 117).

[Larsen, 1998] Thomas Dall Larsen, Nils A Andersen, Ole Ravn, and Niels Kjølstad Poulsen. "Incorporation of time delayed measurements in a discrete-time Kalman filter". In: *37th IEEE Conf. on Decision and Control*. Vol. 4. 1998, pp. 3972–3977 (cit. on p. 114).

[Le Ny, 2009] Jerome Le Ny and George J Pappas. "On trajectory optimization for active sensing in Gaussian process models". In: *48th IEEE Conf. on Decision and Control*. 2009, pp. 6286–6292 (cit. on p. 117).

[Le Ny, 2010] Jerome Le Ny, Eric Feron, and Munther A Dahleh. "Scheduling continuous-time Kalman filters". In: *IEEE Trans. on Automatic Control* 56.6 (2010), pp. 1381–1394 (cit. on p. 117).

[Lee, 2020] Keuntaek Lee, Jason Gibson, and Evangelos A. Theodorou. "Aggressive perception-aware navigation using deep optical flow dynamics and pixelmpc". In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 1207–1214 (cit. on p. 32).

[Lenz, 2020] Christian Lenz, Max Schwarz, Andre Rochow, Jan Razlaw, Arul Selvam Periyasamy, Michael Schreiber, and Sven Behnke. "Autonomous wall building with a UGV-UAV team at MBZIRC 2020". In: *2020 IEEE Int. Symp. on Safety, Security and Rescue Robotics*. 2020, pp. 189–196 (cit. on p. 5).

[Lepetit, 2009] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. "EPnP: An accurate O(n) solution to the PnP problem". In: *Int. Journ. of Computer Vision* 81.2 (2009), pp. 155–166 (cit. on p. 22).

[Li, 2021] Guanrui Li, Alex Tunchez, and Giuseppe Loianno. "PCMPC: Perception-Constrained Model Predictive Control for Quadrotors with Suspended Loads using a Single Camera and IMU". In: *2021 IEEE Int. Conf. on Robotics and Automation*. 2021 (cit. on pp. 32, 89).

[Li, 2022] Guanrui Li, Alex Tunchez, and Giuseppe Loianno. "Learning Model Predictive Control for Quadrotors". In: *2022 IEEE Int. Conf. on Robotics and Automation*. 2022 (cit. on p. 30).

[Liu, 2017] Chang Liu and J Karl Hedrick. "Model predictive control-based target search and tracking using autonomous mobile robot with limited sensing domain". In: *2017 American Control Conference*. 2017, pp. 2937–2942 (cit. on pp. 96, 117, 120–122, 135).

[Lu, 2019] Guozheng Lu and Fu Zhang. "IMU-based attitude estimation in the presence of narrow-band noise". In: *IEEE/ASME Trans. on Mechatronics* 24.2 (2019), pp. 841–852 (cit. on p. 92).

[Lu, 2021] Guozheng Lu, Wei Xu, and Fu Zhang. "Model Predictive Control for Trajectory Tracking on Differentiable Manifolds". 2021 (cit. on p. 30).

[Lundberg, 2018] Cody Lee Lundberg, Hakki Erhan Sevil, and Aditya Das. "A VisualSfM based Rapid 3-D Modeling Framework using Swarm of UAVs". In: *2018 Int. Conf. on Unmanned Aircraft Systems*. 2018, pp. 22–29 (cit. on p. 20).

[Lunni, 2017] Dario Lunni, Angel Santamaria-Navarro, Roberto Rossi, Paolo Rocco, Luca Bascetta, and Juan Andrade-Cetto. "Nonlinear model predictive control for aerial manipulation". In: *2017 Int. Conf. on Unmanned Aircraft Systems*. 2017, pp. 87–93 (cit. on p. 30).

[Madyastha, 2011] Venkatesh Kattigari Madyastha, Vijaysai Prasad, and Venkatram Mahendraker. "Reduced order model monitoring and control of a membrane bioreactor system via delayed measurements". In: *Water Science and Technology* 64.8 (2011), pp. 1675–1684 (cit. on pp. 90, 91).

[Mahler, 2003] Ronald PS Mahler. "Multitarget Bayes filtering via first-order multitarget moments". In: *IEEE Trans. on Aerospace and Electronic System* 39.4 (2003), pp. 1152–1178 (cit. on p. 117).

[Mahony, 2012] Robert Mahony, Vijay Kumar, and Peter Corke. "Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor". In: *IEEE Robotics & Automation Magazine* 19.3 (2012), pp. 20–32 (cit. on p. 16).

[Malis, 1999] Ezio Malis, François Chaumette, and Sylvie Boudet. "2D 1/2 Visual Servoing". In: *IEEE Trans. on Robotics* 15.2 (1999), pp. 238–250 (cit. on p. 24).

[Mallet, 2010] Anthony Mallet, Cédric Pasteur, Matthieu Herrb, Séverin Lemaignan, and Félix Ingrand. "GenoM3: Building middleware-independent robotic components". In: *2010 IEEE Int. Conf. on Robotics and Automation*. 2010, pp. 4627–4632 (cit. on p. 146).

[Mansard, 2009] Nicolas Mansard, Oussama Khatib, and Abderrahmane Kheddar. "A unified approach to integrate unilateral constraints in the stack of tasks". In: *IEEE Trans. on Robotics* 25.3 (2009), pp. 670–685 (cit. on p. 142).

[Mansard, 2018] Nicolas Mansard, Andrea DelPrete, Mathieu Geisert, Steve Tonneau, and Olivier Stasse. "Using a memory of motion to efficiently warm-start a nonlinear predictive controller". In: *2018 IEEE Int. Conf. on Robotics and Automation*. 2018, pp. 2986–2993 (cit. on pp. 29, 30).

[Mao, 2022] Jeffrey Mao, Stephen Nogar, Christopher Kroninger, and Giuseppe Loianno. "Robust Active Visual Perching with Quadrotors on Inclined Surfaces". 2022 (cit. on p. 25).

[Martí-Saumell, 2022] Josep Martí-Saumell, Joan Solà, Angel Santamaria-Navarro, and Juan Andrade-Cetto. "Full-Body Torque-Level Nonlinear Model Predictive Control for Aerial Manipulation". Submitted to IEEE Trans. on Robotics. 2022 (cit. on p. 30).

[Martínez, 2006] Sonia Martínez and Francesco Bullo. "Optimal sensor placement and motion coordination for target tracking". In: *Automatica* 42.4 (2006), pp. 661–668 (cit. on p. 118).

[Martinez, 2017] Julieta Martinez, Michael J Black, and Javier Romero. "On human motion prediction using recurrent neural networks". In: *2017 IEEE Conf. on Computer Vision and Pattern Recognition*. 2017, pp. 2891–2900 (cit. on p. 51).

[Mastalli, 2020] Carlos Mastalli, Rohan Budhiraja, Wolfgang Merkt, Guilhem Saurel, Bilal Hammoud, Maximilien Naveau, Justin Carpentier, Ludovic Righetti, Sethu Vijayakumar, and Nicolas Mansard. "Crocoddyl: An efficient and versatile framework for multi-contact optimal control". In: *2020 IEEE Int. Conf. on Robotics and Automation*. 2020, pp. 2536–2542 (cit. on p. 153).

[Matus-Vargas, 2021] Antonio Matus-Vargas, Gustavo Rodriguez-Gomez, and Jose Martinez-Carranza. "Ground effect on rotorcraft unmanned aerial vehicles: a review". In: *Intelligent Service Robotics* 14.1 (2021), pp. 99–118 (cit. on p. 170).

[Mayne, 2000] David Q. Mayne, James B. Rawlings, Christopher V. Rao, and Pierre O.M. Scokaert. "Constrained model predictive control: Stability and optimality". In: *Automatica* 36.6 (2000), pp. 789–814 (cit. on p. 28).

[Meier, 1967] Lewis Meier, John Peschon, and Robert Dressler. "Optimal control of measurement subsystems". In: *IEEE Trans. on Automatic Control* 12.5 (1967), pp. 528–536 (cit. on pp. 116, 118).

[Michieletto, 2018] Giulia Michieletto, Markus Ryll, and Antonio Franchi. "Fundamental Actuation Properties of Multirotors: Force-Moment Decoupling and Fail-Safe Robustness". In: *IEEE Trans. on Robotics* 34.3 (2018), pp. 702–715 (cit. on pp. 18, 42, 43).

[Mihaylova, 2002] Lyudmila Mihaylova, Tine Lefebvre, Herman Bruyninckx, Klaas Gadeyne, and Joris De Schutter. "A comparison of decision making criteria and optimization methods for active robotic sensing". In: *Int. Conf. on Numerical Methods and Applications*. 2002 (cit. on p. 119).

[Milijas, 2021] Robert Milijas, Lovro Markovic, Antun Ivanovic, Frano Petric, and Stjepan Bogdan. "A comparison of lidar-based slam systems for control of unmanned aerial vehicles". In: *2021 Int. Conf. on Unmanned Aircraft Systems*. 2021, pp. 1148–1154 (cit. on p. 20).

[Mistler, 2001] V. Mistler, Abdelaziz Benallegue, and Nacer Kouider M'sirdi. "Exact linearization and noninteracting control of a 4 rotors helicopter via dynamic feedback". In: *10th IEEE Int. Symp. on Robots and Human Interactive Communications*. 2001, pp. 586–593 (cit. on p. 16).

[Mitrokhin, 2018] Anton Mitrokhin, Cornelia Fermüller, Chethan Parameshwara, and Yiannis Aloimonos. "Event-based moving object detection and tracking". In: *2018 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. 2018, pp. 1–9 (cit. on p. 20).

[Mohta, 2018] Kartik Mohta, Ke Sun, Sikang Liu, Michael Watterson, Bernd Pfrommer, James Svacha, Yash Mulgaonkar, Camillo Jose Taylor, and Vijay Kumar. "Experiments in fast, autonomous, GPS-denied quadrotor flight". In: *2018 IEEE Int. Conf. on Robotics and Automation*. 2018, pp. 7832–7839 (cit. on p. 19).

[Morbidi, 2013] Fabio Morbidi and Gian Luca Mariottini. "Active Target Tracking and Cooperative Localization for Teams of Aerial Vehicles". In: *IEEE Trans. on Control Systems Technology* 21.5 (2013), pp. 1694–1707 (cit. on pp. 84, 98, 119, 122, 123, 135).

[Morin, 2015] Pascal Morin. "Modeling and control of convertible micro air vehicles". In: *10th IEEE Int. Work. on Robot Motion and Control*. 2015, pp. 188–198 (cit. on p. 8).

[Mueller, 2020] Karsten Mueller, Michael Fennel, and Gert F. Trommer. "Model predictive control for vision-based quadrotor guidance". In: *2020 IEEE/ION Position, Location and Navigation Symp.* 2020, pp. 50–61 (cit. on p. 33).

[Mur-Artal, 2015] Raúl Mur-Artal, Jose Maria Martinez Montiel, and Juan D. Tardos. "ORB-SLAM: a versatile and accurate monocular SLAM system". In: *IEEE Trans. on Robotics* 31.5 (2015), pp. 1147–1163 (cit. on p. 21).

[Nan, 2022] Fang Nan, Sihao Sun, Philipp Foehn, and Davide Scaramuzza. "Nonlinear MPC for Quadrotor Fault-Tolerant Control". In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 5047–5054 (cit. on pp. 30, 46).

[Neunert, 2016] Michael Neunert, Cédric De Crousaz, Fadri Furrer, Mina Kamel, Farbod Farshidian, Roland Siegwart, and Jonas Buchli. "Fast nonlinear model predictive control for unified trajectory optimization and tracking". In: *2016 IEEE Int. Conf. on Robotics and Automation*. 2016, pp. 1398–1404 (cit. on p. 30).

[Newcome, 2004] Laurence R. Newcome. *Unmanned aviation: a brief history of unmanned aerial vehicles*. American Institute of Aeronautics and Astronautics, 2004 (cit. on pp. 4, 8).

[Nguyen, 2021] Huan Nguyen, Mina Kamel, Kostas Alexis, and Roland Siegwart. "Model predictive control for micro aerial vehicles: A survey". In: *2021 European Control Conference*. 2021, pp. 1556–1563 (cit. on p. 29).

[Nousi, 2019] Paraskevi Nousi, Danai Triantafyllidou, Anastasios Tefas, and Ioannis Pitas. "Joint lightweight object tracking and detection for unmanned vehicles". In: *2019 IEEE Int. Conf on Image Processing*. 2019, pp. 160–164 (cit. on p. 21).

[Olfati-Saber, 2007] Reza Olfati-Saber. "Distributed Kalman filtering for sensor networks". In: *46th IEEE Conf. on Decision and Control*. 2007, pp. 5492–5498 (cit. on p. 114).

[Ollero, 2018] Anibal Ollero, Guillermo Heredia, Antonio Franchi, Gianluca Antonelli, Konstantin Kondak, Alberto Sanfeliu, Antidio Viguria, J Ramiro Martinez-de Dios, Francesco Pierri, Juan Cortés, et al. "The AEROARMS project: Aerial robots with advanced manipulation capabilities for inspection and maintenance". In: *IEEE Robotics & Automation Magazine* 25.4 (2018), pp. 12–23 (cit. on p. 7).

[Ollero, 2021] Anibal Ollero, Marco Tognon, Alejandro Suarez, Dongjun Lee, and Antonio Franchi. "Past, present, and future of aerial robotic manipulators". In: *IEEE Trans. on Robotics* (2021) (cit. on p. 7).

[Olson, 2011] Edwin Olson. "AprilTag: A robust and flexible visual fiducial system". In: *2011 IEEE Int. Conf. on Robotics and Automation*. 2011, pp. 3400–3407 (cit. on p. 48).

[Paneque, 2022] Julio L. Paneque, Jose Ramiro Martinez-de-Dios, Anibal Ollero, Drew Hanover, Sihao Sun, Angel Romero, and Davide Scaramuzza. "Perception-Aware Perching on Powerlines with Multirotors". In: *IEEE Robotics and Automation Letters* 7.2 (2022) (cit. on p. 32).

[Papaioannou, 2019] Savvas Papaioannou, Panayiotis Kolios, Theocharis Theocharides, Christos G Panayiotou, and Marios M. Polycarpou. "Jointly-optimized searching and tracking with random finite sets". In: *IEEE Trans. on Mobile Computing* 19.10 (2019), pp. 2374–2391 (cit. on p. 120).

[Park, 1995] Frank C Park. "Distance metrics on the rigid-body motions with applications to mechanism design". In: *ASME Journal on Mechanical Design* (1995), pp. 48–54 (cit. on p. 60).

[Park, 1997] Frank C Park and Bahram Ravani. "Smooth invariant interpolation of rotations". In: *ACM Transactions on Graphics* 16.3 (1997), pp. 277–295 (cit. on p. 60).

[Patil, 2014] Sachin Patil, Yan Duan, John Schulman, Ken Goldberg, and Pieter Abbeel. "Gaussian belief space planning with discontinuities in sensing domains". In: *2014 IEEE Int. Conf. on Robotics and Automation*. 2014, pp. 6483–6490 (cit. on p. 120).

[Penin, 2017] Bryan Penin, Riccardo Spica, Paolo Robuffo Giordano, and François Chaumette. "Vision-based minimum-time trajectory generation for a quadrotor UAV". In: *2017 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. 2017, pp. 6199–6206 (cit. on pp. 31, 57, 70).

[Penin, 2018] Bryan Penin, Paolo Robuffo Giordano, and François Chaumette. "Vision-Based Reactive Planning for Aggressive Target Tracking While Avoiding Collisions and Occlusions". In: *IEEE Robotics and Automation Letters* 3.4 (2018), pp. 3725–3732 (cit. on pp. 31, 32, 57, 67, 72, 89).

[Pereira, 2021] Jean Carlos. Pereira, Valter J.S. Leite, and Guilherme V. Raffo. "Nonlinear model predictive control on SE(3) for quadrotor aggressive maneuvers". In: *Journal of Intelligent & Robotics Systems* 101.3 (2021), pp. 1–15 (cit. on p. 29).

[Petracek, 2020] Pavel Petracek, Vit Kratky, and Martin Saska. "Dronument: System for Reliable Deployment of Micro Aerial Vehicles in Dark Areas of Large Historical Monuments". In: *IEEE Robotics and Automation Letters* 5 (2020), pp. 2078–2085 (cit. on p. 6).

[Pimenta, 2008] Luciano C.A. Pimenta, Vijay Kumar, Renato C. Mesquita, and Guilherme A.S. Pereira. "Sensing and coverage for a network of heterogeneous robots". In: *47th IEEE Conf. on Decision and Control*. 2008, pp. 3947–3952 (cit. on p. 117).

[Pounds, 2010] Paul Pounds, Robert Mahony, and Peter Corke. "Modelling and control of a large quadrotor robot". In: *Control Engineering Practice* 18.7 (2010), pp. 691–699 (cit. on p. 16).

[Powers, 2015] Caitlin Powers, Daniel Mellinger, and Vijay Kumar. "Quadrotor kinematics and dynamics". In: *Handbook of Unmanned Aerial Vehicles*. Ed. by Kimon P. Valavanis and George J. Vachtsevanos. Springer, 2015, pp. 307–328 (cit. on p. 16).

[Qin, 2018] Tong Qin, Peiliang Li, and Shaojie Shen. "VINS-Mono: A robust and versatile monocular visual-inertial state estimator". In: *IEEE Trans. on Robotics* 34.4 (2018), pp. 1004–1020 (cit. on p. 21).

[Qin, 2021] Yongming Qin, Makoto Kumon, and Tomonari Furukawa. "Estimation of a Human-Maneuvered Target Incorporating Human Intention". In: *Sensors* 21.16 (2021), p. 5316 (cit. on p. 51).

[Queralta, 2020] Jorge Pena Queralta, Jussi Taipalmaa, Bilge Can Pullinen, Victor Kathan Sarker, Tuan Nguyen Gia, Hannu Tenhunen, Moncef Gabbouj, Jenni Raitoharju, and Tomi Westerlund. "Collaborative Multi-Robot Search and Rescue: Planning, Coordination, Perception, and Active Vision". In: *IEEE Access* 8 (2020) (cit. on pp. 21, 22).

[Rajappa, 2015] Sujit Rajappa, Markus Ryll, Heinrich H Bülthoff, and Antonio Franchi. "Modeling, control and design optimization for a fully-actuated hexarotor aerial vehicle with tilted propellers". In: *2015 IEEE Int. Conf. on Robotics and Automation*. 2015, pp. 4006–4013 (cit. on pp. 9, 17, 45, 148).

[Rao, 2009] Anil V. Rao. "A survey of numerical methods for optimal control". In: *Advances in the Astronautical Sciences*. Vol. 135. Univelt, 2009, pp. 497–528 (cit. on p. 27).

[Redmon, 2016] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. "You only look once: Unified, real-time object detection". In: *2016 IEEE Conf. on Computer Vision and Pattern Recognition*. 2016, pp. 779–788 (cit. on p. 22).

[Ren, 2012] Beibei Ren, Shuzhi Sam Ge, Chang Chen, Cheng-Heng Fua, and Tong Heng Lee. *Modeling, control and coordination of helicopter systems*. Springer Science & Business Media, 2012 (cit. on p. 9).

[Rodrigues, 2020] Rômulo T Rodrigues, Pedro Miraldo, Dimos V Dimarogonas, and A Pedro Aguiar. "Active depth estimation: Stability analysis and its applications". In: *2020 IEEE Int. Conf. on Robotics and Automation*. 2020, pp. 2002–2008 (cit. on pp. 20, 22).

[Rosten, 2008] Edward Rosten, Reid Porter, and Tom Drummond. "Faster and better: A machine learning approach to corner detection". In: *IEEE Trans. on Pattern Analysis & Machine Intelligence* 32.1 (2008), pp. 105–119 (cit. on p. 21).

[Rouček, 2019] Tomáš Rouček et al. "DARPA subterranean challenge: Multi-robotic exploration of underground environments". In: *Int. Conf. on Modelling and Simulation for Autonomous Systems*. 2019, pp. 274–290 (cit. on p. 5).

[Roumeliotis, 1999] Stergios I. Roumeliotis, Gaurav S. Sukhatme, and George A. Bekey. "Circumventing dynamic modeling: Evaluation of the error-state kalman filter applied to mobile robot localization". In: *1999 IEEE Int. Conf. on Robotics and Automation*. 1999, pp. 1656–1663 (cit. on pp. 90, 91).

[Roumeliotis, 2000] Stergios I Roumeliotis and George A Bekey. "Collective localization: A distributed kalman filter approach to localization of groups of mobile robots". In: *2000 IEEE Int. Conf. on Robotics and Automation*. 2000, pp. 2958–2965 (cit. on p. 114).

[Rublee, 2011] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. "ORB: An efficient alternative to SIFT or SURF". In: *2011 IEEE Int. Conf. on Computer Vision*. 2011, pp. 2564–2571 (cit. on p. 21).

[Rucker, 2018] Caleb Rucker. "Integrating rotations using nonunit quaternions". In: *IEEE Robotics and Automation Letters* 3.4 (2018), pp. 2979–2986 (cit. on pp. 41, 153).

[Ryll, 2015] Markus Ryll, Heinrich H Bülthoff, and Paolo Robuffo Giordano. "A novel overactuated quadrotor unmanned aerial vehicle: Modeling, control, and experimental validation". In: *IEEE Trans. on Control Systems Technology* 23.2 (2015), pp. 540–556 (cit. on pp. 18, 45).

[Ryll, 2016] Markus Ryll, Davide Bicego, and Antonio Franchi. "Modeling and control of FAST-Hex: A fully-actuated by synchronized-tilting hexarotor". In: *2016 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. 2016, pp. 1689–1694 (cit. on p. 18).

[Ryll, 2017] Markus Ryll, Giuseppe Muscio, Francesco Pierri, Elisabetta Cataldi, Gianluca Antonelli, Fabrizio Caccavale, and Antonio Franchi. "6D physical interaction with a fully actuated aerial robot". In: *2017 IEEE Int. Conf. on Robotics and Automation*. 2017, pp. 5190–5195 (cit. on p. 19).

[Ryll, 2019] Markus Ryll, Giuseppe Muscio, Francesco Pierri, Elisabetta Cataldi, Gianluca Antonelli, Fabrizio Caccavale, Davide Bicego, and Antonio Franchi. "6D Interaction Control with Aerial Robots: The Flying End-Effector Paradigm". In: *The International Journal of Robotics Research* 38.9 (2019), pp. 1045–1062 (cit. on p. 166).

[Salaris, 2019] Paolo Salaris, Marco Cognetti, Riccardo Spica, and Paolo Robuffo Giordano. "Online optimal perception-aware trajectory generation". In: *IEEE Trans. on Robotics* 35.6 (2019), pp. 1307–1322 (cit. on p. 119).

[Schlotfeldt, 2018] Brent Schlotfeldt, Dinesh Thakur, Nikolay Atanasov, Vijay Kumar, and George J. Pappas. "Anytime Planning for Decentralized Multirobot Active Information Gathering". In: *IEEE Robotics and Automation Letters* 3.2 (2018), pp. 1025–1032 (cit. on pp. 119, 120, 136).

[Schlotfeldt, 2019] Brent Schlotfeldt, Nikolay Atanasov, and George J. Pappas. "Maximum information bounds for planning active sensing trajectories". In: *2019 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. 2019, pp. 4913–4920 (cit. on pp. 117, 118).

[Şenkul, 2014] Fatih Şenkul and Erdinç Altuğ. "Adaptive control of a tilt-roll rotor quadrotor UAV". In: *2014 Int. Conf. on Unmanned Aircraft Systems*. 2014, pp. 1132–1137 (cit. on p. 18).

[Shahidian, 2017] Seyyed Ali Asghar Shahidian and Hadi Soltanizadeh. "Single-and multi-UAV trajectory control in RF source localization". In: *Arabian Journal for Science and Engineering* 42.2 (2017), pp. 459–466 (cit. on p. 119).

[Sheckells, 2016] Matthew Sheckells, Gowtham Garimella, and Marin Kobilarov. "Optimal visual servoing for differentially flat underactuated systems". In: *2016 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. 2016, pp. 5541–5548 (cit. on p. 25).

[Shi, 1994] Jianbo Shi et al. "Good features to track". In: *1994 IEEE Conf. on Computer Vision and Pattern Recognition*. 1994, pp. 593–600 (cit. on p. 21).

[Siciliano, 2008] Bruno Siciliano, Oussama Khatib, and Torsten Kröger. *Springer Handbook of Robotics*. Springer, 2008 (cit. on p. 5).

[Siciliano, 2009] Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, and Giuseppe Oriolo. *Robotics: Modelling, Planning and Control*. Springer, 2009 (cit. on pp. 5, 38, 39, 43, 169).

[Singh Bal, 2019] Baljinder Singh Bal. "Cooperative target tracking for heterogeneous robots". MA thesis. Università di Bologna, 2019 (cit. on p. 120).

[Sinopoli, 2004] Bruno Sinopoli, Luca Schenato, Massimo Franceschetti, Kameshwar Poolla, Michael I Jordan, and Shankar S Sastry. "Kalman filtering with intermittent observations". In: *IEEE Trans. on Automatic Control* 49.9 (2004), pp. 1453–1464 (cit. on pp. 120–122).

[Smeur, 2018] Ewoud J. J. Smeur, Guido C.H.E. de Croon, and Qiping Chu. "Cascaded incremental nonlinear dynamic inversion for MAV disturbance rejection". In: *Control Engineering Practice* 73 (2018), pp. 79–90 (cit. on p. 28).

[Solà, 2017] Joan Solà. *Quaternion kinematics for the error-state Kalman filter*. Tech. rep. 2017 (cit. on pp. 38, 41, 42, 90, 91).

[Solà, 2018] Joan Solà, Jeremie Deray, and Dinesh Atchuthan. *A micro Lie theory for state estimation in robotics*. Tech. rep. 2018 (cit. on pp. 39, 50, 91, 93).

[Sommerlade, 2008] Eric Sommerlade and Ian Reid. "Information-theoretic active scene exploration". In: *2008 IEEE Conf. on Computer Vision and Pattern Recognition*. 2008, pp. 1–7 (cit. on p. 117).

[Spasojevic, 2020] Igor Spasojevic, Varun Murali, and Sertac Karaman. "Perception-aware time optimal path parameterization for quadrotors". In: *2020 IEEE Int. Conf. on Robotics and Automation*. 2020, pp. 3213–3219 (cit. on p. 25).

[Spica, 2013] Riccardo Spica, Paolo Robuffo Giordano, Markus Ryll, Heinrich H Bülthoff, and Antonio Franchi. "An open-source hardware/software architecture for quadrotor UAVs". In: *2nd IFAC Work. on Res., Educ. and Develop. of Unmanned Aerial Systems*. 2013 (cit. on p. 80).

[Staub, 2018] Nicolas Staub, Davide Bicego, Quentin Sablé, Victor Arellano, Subodh Mishra, and Antonio Franchi. "Towards a flying assistant paradigm: the OTHex". In: *2018 IEEE Int. Conf. on Robotics and Automation*. 2018, pp. 6997–7002 (cit. on pp. 7, 17, 19).

[Strabala, 2013] Kyle Strabala, Min Kyung Lee, Anca Dragan, Jodi Forlizzi, Siddhartha S Srinivasa, Maya Cakmak, and Vincenzo Micelli. "Toward seamless human-robot handovers". In: *Journal of Human-Robot Interaction* 2.1 (2013), pp. 112–132 (cit. on p. 83).

[Sun, 2021] Sihao Sun, Xuerui Wang, Qiping Chu, and Coen de Visser. "Incremental nonlinear fault-tolerant control of a quadrotor with complete loss of two opposing rotors". In: *IEEE Trans. on Robotics* 37.1 (2021), pp. 116–130 (cit. on p. 30).

[Sun, 2022] Sihao Sun, Angel Romero, Philipp Foehn, Elia Kaufmann, and Davide Scaramuzza. "A comparative study of nonlinear MPC and differential-flatness-based control for quadrotor agile flight". In: *IEEE Trans. on Robotics* (2022) (cit. on pp. 28, 143).

[Swain, 1992] Michael J. Swain and Dana H. Ballard. "Indexing via color histograms". In: *Active perception and robot vision*. Ed. by Arun K. Sood and Harry Wechsler. Springer, 1992, pp. 261–273 (cit. on p. 163).

[Syed, 2007] Zainab F. Syed, P Aggarwal, C Goodall, X Niu, and N El-Sheimy. "A new multi-position calibration method for MEMS inertial navigation systems". In: *Measurement science and technology* 18.7 (2007), p. 1897 (cit. on p. 90).

[Tallamraju, 2019] Rahul Tallamraju, Eric Price, Roman Ludwig, Kamalakar Karlapalem, Heinrich H Bülthoff, Michael J. Black, and Aamir Ahmad. "Active perception based formation control for multiple aerial vehicles". In: *IEEE Robotics and Automation Letters* 4.4 (2019), pp. 4491–4498 (cit. on pp. 118, 130, 135).

[Tedaldi, 2014] David Tedaldi, Alberto Pretto, and Emanuele Menegatti. "A robust and easy to implement method for IMU calibration without external equipments". In: *2014 IEEE Int. Conf. on Robotics and Automation*. 2014, pp. 3042–3049 (cit. on pp. 90, 147).

[Thomas, 2017] Justin Thomas, Jake Welde, Giuseppe Loianno, Kostas Daniilidis, and Vijay Kumar. "Autonomous Flight for Detection, Localization, and Tracking of Moving Targets With a Small Quadrotor". In: *IEEE Robotics and Automation Letters* 2.3 (2017), pp. 1762–1769 (cit. on pp. 20, 22, 31, 33, 117).

[Tognon, 2017] Marco Tognon, Burak Yüksel, Gabriele Buondonno, and Antonio Franchi. "Dynamic Decentralized Control for Protocentric Aerial Manipulators". In: *2017 IEEE Int. Conf. on Robotics and Automation*. 2017, pp. 6375–6380 (cit. on pp. 19, 166).

[Tognon, 2018] Marco Tognon. "Theory and Applications for Control and Motion Planning of Aerial Robots in Physical Interaction with particular focus on Tethered Aerial Vehicles". PhD thesis. Institut national des sciences appliquées de Toulouse, 2018 (cit. on p. 7).

[Tognon, 2019] Marco Tognon et al. "A Truly Redundant Aerial Manipulator System with Application to Push-and-Slide Inspection in Industrial Plants". In: *IEEE Robotics and Automation Letters* 4.2 (2019), pp. 1846–1851 (cit. on pp. 19, 166).

[Tokekar, 2014] Pratap Tokekar, Volkan Isler, and Antonio Franchi. "Multi-target visual tracking with aerial robots". In: *2014 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. 2014, pp. 3067–3072 (cit. on pp. 117, 120).

[Tomić, 2017] Teodor Tomić, Christian Ott, and Sami Haddadin. "External Wrench Estimation, Collision Detection, and Reflex Reaction for Flying Robots". In: *IEEE Trans. on Robotics* 33.6 (2017), pp. 1467–1482 (cit. on p. 169).

[Tordesillas, 2022] Jesus Tordesillas and Jonathan P. How. "PANTHER: Perception-aware trajectory planner in dynamic environments". In: *IEEE Access* 10 (2022), pp. 22662–22677 (cit. on p. 25).

[Torrente, 2021] Guillem Torrente, Elia Kaufmann, Philipp Föhn, and Davide Scaramuzza. "Data-driven MPC for quadrotors". In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 3769–3776 (cit. on p. 30).

[Tremblay, 2018] Jonathan Tremblay, Thang To, Balakumar Sundaralingam, Yu Xiang, Dieter Fox, and Stan Birchfield. "Deep Object Pose Estimation for Semantic Robotic Grasping of Household Objects". In: *2nd Conf. on Robot Learning*. 2018, pp. 306–316 (cit. on pp. 23, 47, 142).

[Trinh, 2015] Trong Hai Trinh, Manh Ha Tran, et al. "Hole boundary detection of a surface of 3D point clouds". In: *2015 IEEE Int. Conf. on Advanced Computing and Applications*. 2015, pp. 124–129 (cit. on p. 164).

[Tron, 2016] Roberto Tron, Justin Thomas, Giuseppe Loianno, Kostas Daniilidis, and Vijay Kumar. "A distributed optimization framework for localization and formation control: Applications to vision-based measurements". In: *IEEE Robotics & Automation Magazine* 36.4 (2016), pp. 22–44 (cit. on p. 109).

[Trujillo, 2019] Miguel Ángel Trujillo, José Ramiro Martínez-de Dios, Carlos Martín, Antidio Viguria, and Aníbal Ollero. "Novel aerial manipulator for accurate and robust industrial NDT contact inspection: A new tool for the oil and gas inspection industry". In: *Sensors* 19.6 (2019), pp. 1305–1329 (cit. on p. 19).

[Tzoumanikas, 2020] Dimos Tzoumanikas, Qingyue Yan, and Stefan Leutenegger. "Nonlinear MPC with motor failure identification and recovery for safe and aggressive multicopter flight". In: *2020 IEEE Int. Conf. on Robotics and Automation*. 2020, pp. 8538–8544 (cit. on p. 30).

[Urban, 2016] S. Urban, J. Leitloff, and S. Hinz. "MLPnP - A Real-Time Maximum Likelihood Solution to the Perspective-n-Point Problem". In: *ISPRS Ann. of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 3.3 (2016), pp. 131–138 (cit. on p. 48).

[Varotto, 2022] Luca Varotto, Angelo Cenedese, and Andrea Cavallaro. "Active Sensing for Search and Tracking: A Review". Submitted to IEEE Trans. on Cybernetics. 2022 (cit. on p. 9).

[Verschueren, 2021] Robin Verschueren, Gianluca Frison, Dimitris Kouzoupis, Jonathan Frey, Niels van Duijkeren, Andrea Zanelli, Branimir Novoselnik, Thivaharan Albin, Rien Quirynen, and Moritz Diehl. "acados – a modular open-source framework for fast embedded optimal control". In: *Mathematical Programming Computation* 14 (2021), pp. 147–183 (cit. on p. 152).

[Vitus, 2012] Michael P Vitus, Wei Zhang, Alessandro Abate, Jianghai Hu, and Claire J Tomlin. "On efficient sensor scheduling for linear dynamical systems". In: *Automatica* 48.10 (2012), pp. 2482–2493 (cit. on p. 117).

[Walter, 2018] Viktor Walter, Martin Saska, and Antonio Franchi. "Fast mutual relative localization of uavs using ultraviolet led markers". In: *2018 Int. Conf. on Unmanned Aircraft Systems*. 2018, pp. 1217–1226 (cit. on p. 109).

[Watanabe, 2006] Yoko Watanabe, Anthony Calise, Eric Johnson, and Johnny Evers. "Minimum-effort guidance for vision-based collision avoidance". In: *AIAA atmospheric flight mechanics conference and exhibit*. 2006 (cit. on p. 110).

[Wofk, 2019] Diana Wofk, Fangchang Ma, Tien-Ju Yang, Sertac Karaman, and Vivienne Sze. "Fastdepth: Fast monocular depth estimation on embedded systems". In: *2019 IEEE Int. Conf. on Robotics and Automation*. 2019, pp. 6101–6108 (cit. on pp. 20, 22).

[Xiao, 2021] Shengjie Xiao, Kai Hu, Binxiao Huang, Huichao Deng, and Xilun Ding. "A Review of Research on the Mechanical Design of Hoverable Flapping Wing Micro-Air Vehicles". In: *Journal of Bionic Engineering* (2021), pp. 1235–1254 (cit. on p. 7).

[Yakushiji, 2020] Koki Yakushiji, Hiroshi Fujita, Mikio Murata, Naoki Hiroi, Yuuichi Hamabe, and Fumiatsu Yakushiji. "Short-range transportation using unmanned aerial vehicles (UAVS) during disasters in Japan". In: *Drones* 4.4 (2020), p. 68 (cit. on p. 6).

[Yang, 2007] Peng Yang, Randy A. Freeman, and Kevin M. Lynch. "Distributed Cooperative Active Sensing Using Consensus Filters". In: *2007 IEEE Int. Conf. on Robotics and Automation*. 2007, pp. 405–410 (cit. on pp. 98, 117–119, 122, 123).

[Yang, 2017] Chao Yang, Jiangying Zheng, Xiaoqiang Ren, Wen Yang, Hongbo Shi, and Ling Shi. "Multi-sensor Kalman filtering with intermittent measurements". In: *IEEE Trans. on Automatic Control* 63.3 (2017), pp. 797–804 (cit. on pp. 120–122).

[Yasin, 2020] Jawad N Yasin, Sherif AS Mohamed, Mohammad-Hashem Haghbayan, Jukka Heikkonen, Hannu Tenhunen, and Juha Plosila. "Unmanned aerial vehicles (UAVS): Collision avoidance systems and approaches". In: *IEEE Access* 8 (2020), pp. 105139–105155 (cit. on p. 110).

[Yiğit, 2021] Arda Yiğit, Miguel Arpa Perozo, Mandela Ouafo, Loïc Cuvillon, Sylvain Durand, and Jacques Gangloff. "Aerial Manipulator Suspended from a Cable-Driven Parallel Robot: Preliminary Experimental Results". In: (2021), pp. 9662–9668 (cit. on pp. 8, 19).

[Zeilinger, 2010] Melanie N Zeilinger, Colin N Jones, and Manfred Morari. "Robust stability properties of soft constrained MPC". In: *49th IEEE Conf. on Decision and Control*. 2010, pp. 5276–5282 (cit. on p. 72).

[Zemas, 2017] Nicola Roberto Zemas, Enrico Natalizio, and Evsen Yanmaz. "An Unmanned Aerial Vehicle Network for Sport Event Filming with Communication Constraints". In: *1st Int. Balkan Conf. on Communications and Networking*. 2017 (cit. on pp. 9, 135).

[Zhang, 2015] Xu Zhang, Bin Xian, Bo Zhao, and Yao Zhang. "Autonomous flight control of a nano quadrotor helicopter in a GPS-denied environment using on-board vision". In: *IEEE Trans. on Industrial Electronics* 62.10 (2015), pp. 6392–6403 (cit. on p. 16).

[Zhang, 2019] Pengyi Zhang, Yunxin Zhong, and Xiaoqiong Li. "SlimYOLOv3: Narrower, faster and better for real-time UAV applications". In: *2020 IEEE/CVF Int. Conf. on Computer Vision*. 2019 (cit. on p. 22).

[Zhao, 2018] Moju Zhao, Tomoki Anzai, Fan Shi, Xiangyu Chen, Kei Okada, and Masayuki Inaba. "Design, modeling, and control of an aerial robot dragon: A dual-rotor-embedded multilink robot with the ability of multi-degree-of-freedom aerial transformation". In: *IEEE Robotics and Automation Letters* 3.2 (2018), pp. 1176–1183 (cit. on p. 8).

[Zhu, 2018] Pengfei Zhu, Longyin Wen, Dawei Du, Xiao Bian, Haibin Ling, Qinghua Hu, Qinqin Nie, Hao Cheng, Chenfeng Liu, Xiaoyu Liu, et al. "Visdrone-DET2018: The vision meets drone object detection in image challenge results". In: *2018 IEEE/CVF European Conf. on Computer Vision*. 2018 (cit. on p. 22).

Résumé - Abstract

Les robots aériens sont de plus en plus présents dans de nombreuses applications. On peut citer par exemple la création photo ou video ou bien diverses activités d'observation dans des endroits difficiles d'accès. Le déploiement de robots autonomes nécessite néanmoins des outils de perception de l'environnement. En particulier, pour le cas des robots multi-rotors, les impératifs de perception, tels que le maintien de repères visuels dans le champ de vision, entre souvent en conflit avec le mouvement du drone ou la tâche à accomplir.

Cette thèse propose des méthodes de contrôle qui permettent de mettre en concordance les contraintes de perception et les contraintes d'actionnement afin de produire des mouvements en conséquence. Ces méthodes s'appuient sur des techniques mathématiques d'optimisation sous contraintes. Elles utilisent des modèles mathématiques pour prédire l'évolution du système et en calculer les commandes optimales à générer pour accomplir les différents objectifs.

Mots clefs - Perception, Contrôle Prédictif, Robots Aériens, Systèmes Multi-Robots

---

Aerial robots are increasingly used in various applications, e.g. photo or video production, or monitoring in complex environments. However, employing autonomous robots in such uncontrolled environments requires perception of the surrounding, assessing the workspace and potential hazards. In particular, the need for perception often collides with motion or task-related constraints.

This thesis proposes control methods accounting for both perception and actuation limitations, in order to produce suitable motions. These methods are based on mathematical constrained optimization techniques, using various models to predict the system evolution over time. The optimal inputs are computed to fulfill the required tasks under the aforementioned constraints.

Keywords - Perception, Predictive Control, Aerial Robots, Multi-Robot Systems