DISS. ETH NO. 29282

TOWARDS ROBUST ACTIVE PLANNING FOR AUTONOMOUS AERIAL NAVIGATION

A thesis submitted to attain the degree of

DOCTOR OF SCIENCES of ETH ZURICH

(Dr. sc. ETH Zurich)

presented by

LUCA BARTOLOMEI

M. Sc. ETH in Robotics, Systems and Control (RSC), ETH Zurich

born on May 7, 1993
citizen of Italy

accepted on the recommendation of

Prof. Dr. Margarita Chli, Examiner
Prof. Dr. Konstantinos Alexis, Co-examiner
Prof. Dr. Sebastian Scherer, Co-examiner

2023

Department of Mechanical and Process Engineering
ETH Zurich
Switzerland

© 2023 Luca Bartolomei. All rights reserved.

Abstract

Unmanned Aerial Vehicles (UAVs), more commonly known as drones, have the potential to profoundly impact numerous applications, such as inspection, post-disaster assessment and Search and Rescue (SaR), thanks to their agility, which translates to a unique ability to move freely in 3D space, and their continuously decreasing cost. However, despite the recent technological advances, commercially available drones are either flown manually by an experienced pilot or in semi-autonomous mode largely based on GPS. In an effort to automate the aforementioned tasks using small rotorcraft UAVs, the research community has been focusing on improving their capacity to navigate unknown environments autonomously, while relying on onboard sensing for pose estimation, mapping and path planning. More advanced extensions deploying multiple robots have also been proposed, aiming at boosting the effectiveness of robotic missions further, as efficiency is particularly crucial in time-critical applications, such as rescue operations. However, while accelerating missions massively, the deployment of multiple UAVs entails a series of difficulties in terms of co-localization and coordination of the swarm. Inspired by these challenges, this thesis begins by addressing the problem of multi-robot collaboration for inspection and exploration tasks, while in the second part it concentrates on the problem of perception-aware navigation, also known in the literature as *active planning*.

Aiming at relaxing the assumptions typical in multi-robot planning, such as the availability of the agents' ground-truth poses, as well as of a prior map of the environment, the first approach proposes a centralized multi-robot architecture, encompassing state estimation, dense mapping and multi-agent coordination. The objective is the generation of a consistent 3D map of a large-scale structure of interest using a team of drones flying a pre-defined coarse mission plan. However, although proposing a complete, applicable solution, this approach requires initial guidance from a human operator. Addressing this limitation, in a follow-up approach, a decentralized coordination strategy for automatic exploration of forests is proposed. As this type of scenery is common in SaR, minimizing the time required to complete the coverage of the area of interest is vital; therefore, here, we propose an efficient strategy able to exploit the agility of UAVs, keeping consistently high flight velocities throughout the mission, despite potentially high density of obstacles and large number of occlusions in the environment.

Although exhibiting promising performance, the two approaches rely on the assumption that the poses of the UAV-robots can be estimated directly and accurately by sensors, such as GPS, which can fail in a number of situations, for example, when near to structures or close to buildings. A well-established alternative to GPS-based localization in UAV navigation literature is Visual-Inertial Simultaneous Localization And Mapping (VI-SLAM), where the robot's poses are estimated using sequences of images and high-rate inertial measurements. On the other hand, VI-SLAM is sensitive to the motions of the UAV, and the performance of camera-based state estimation is strongly linked to the visual appearance of the environment in general. Driven by these limitations, the second part of this thesis focuses on the problem of perception-aware path planning, with the objective of minimizing the error in pose estimation by generating camera-aware motions. Inspired by the maturity of semantic segmentation in the Computer Vision literature, partitioning the scene into semantically meaningful clusters of pixels, this thesis addresses the problem of active planning using semantic cues. In contrast to the large body of research carried out in the autonomous driving literature, to the best of our knowledge, this was the first time a semantic-aware active perception approach for UAVs was presented.

Guided by semantic scene annotation, the first proposed approach for active planning encourages the robot to navigate over visually-reliable regions, e.g. solid scene structures, such as buildings, while avoiding perceptually degraded areas, characterized, for example, by high dynamism or reflective surfaces, such as water basins. This concept of using semantics for reliable vision-based navigation is pushed forward in a follow-up approach, where a deep Reinforcement Learning (RL) policy is trained to identify useful areas for VI-SLAM online during mission execution. This allows the UAV to adapt its future trajectory dynamically to the scene in view, improving the performance in terms of robustness and localization error.

Nevertheless, during deployment in real missions, small UAVs are susceptible to a series of possible threats, such as strong wind gusts or sensor faults, potentially leading to crashes. Thus, in these situations it is fundamental to render them capable of finding a suitable landing spot where to land autonomously. In an effort of ensuring the safety of the robot and the landscape, especially in urban areas, this thesis closes with a semantic-aware approach for autonomous emergency landing of multi-rotor UAVs. Here, following a deep-RL paradigm, we demonstrate that semantic information allows to find a landing spot faster by exploiting the high-level spatial associations between semantic classes (e.g. cars and roads). The proposed pipeline can be directly deployed in real-world experiments immediately after policy training, without additional fine-tuning or domain adaptation.

With the focus on multi-robot coordination and perception-aware active planning for UAVs, the approaches and systems presented in this thesis contribute towards autonomous aerial navigation deployable in challenging real-world scenarios. Furthermore, it is demonstrated that the use of semantic segmentation can be extremely beneficial for path-planning purposes during vision-based flights and autonomous emergency landing. This leads to more robust methods, able to succeed where state-of-the-art systems fail, paving the way towards more reliable autonomous navigation of robotic agents.

Sommario

I veicoli aerei senza pilota (Unmanned Aerial Vehicles – UAV), più comunemente noti come droni, possono potenzialmente automatizzare numerose mansioni, come l'ispezione di edifici, le valutazioni di sicurezza in seguito a catastrofi naturali, o le missioni di ricerca e salvataggio (Search-and-Rescue – SaR), grazie alla loro agilità, che si traduce in un'ineguagliabile capacità di muoversi liberamente nello spazio 3D, e al loro costo in continua diminuzione. Tuttavia, nonostante i recenti progressi tecnologici in questo campo, la maggioranza dei droni disponibili sul mercato viene pilotata manualmente da un operatore esperto, oppure viene utilizzata in modalità semi-autonoma, affidandosi principalmente al GPS. Con lo scopo di automatizzare gli utilizzi menzionati precedentemente utilizzando UAV di piccole dimensioni, la comunità scientifica si è concentrata sul potenziamento della capacità di questi robot di navigare autonomamente in ambienti sconosciuti, affidandosi esclusivamente ai dati rilevati in tempo reale dai sensori a bordo del drone per la stima della posizione del robot, la mappatura dell'ambiente circostante e la pianificazione del percorso da seguire. Recentemente sono anche state proposte estensioni più avanzate che prevedono il dispiego simultaneo di più UAV, con lo scopo di migliorare l'efficacia delle missioni in cui l'efficienza rappresenta un aspetto cruciale, come ad esempio le operazioni di salvataggio, dove sono richiesti tempi di completamento brevi. Tuttavia, pur portando ad un'accelerazione massiccia, il dispiegamento di più UAV comporta una serie di difficoltà in termini di co-localizzazione e coordinamento della squadra di robot. Traendo ispirazione da queste difficoltà tecniche, questa tesi si apre affrontando il tema della collaborazione multi-robot per mansioni come l'ispezione di strutture e l'esplorazione di ambienti sconosciuti, mentre nella seconda parte si concentra maggiormente sull'aspetto della navigazione autonoma basata su sensori esterocettivi, come ad esempio telecamere, argomento noto in letteratura anche come *pianificazione attiva* delle traiettorie.

Il primo approccio che viene presentato ha l'obiettivo di rilassare le ipotesi tipiche della letteratura sul coordinamento di più robot, come ad esempio la disponibilità di posizioni accurate degli agenti e di una mappa dell'ambiente a priori, e propone un'architettura multi-robot centralizzata, che comprende la stima dello stato, ovvero della posizione e dell'orientazione di ciascun robot, la capacità di creare mappe dell'area di volo e il coordinamento tra agenti robotici. L'obiettivo è quello di generare una mappa 3D di una struttura di interesse di grandi dimensioni utilizzando una squadra di droni mentre questi seguono un percorso predefinito. Tuttavia, pur proponendo una soluzione completa e direttamente utilizzabile in una varietà di situazioni, questo approccio richiede una guida iniziale da parte di un operatore. Per ovviare a questa limitazione, in un secondo approccio, viene proposta una strategia di coordinamento decentralizzato per l'esplorazione automatizzata di aree forestali. Poiché questo tipo di ambiente è comune in operazioni di salvataggio, è fondamentale ridurre al minimo il tempo necessario per completare l'esplorazione dell'area di interesse; pertanto, in questo lavoro viene proposta una strategia efficiente, in grado di sfruttare l'agilità degli UAV, che mantiene velocità di volo costantemente elevate per tutta la missione, nonostante la potenziale alta densità di ostacoli e l'elevato numero di occlusioni nell'ambiente.

Pur mostrando prestazioni promettenti, i due approcci finora descritti si basano sull'ipotesi che le posizioni degli UAV possano essere stimate direttamente e accuratamente dai dati sensoriali, usando ad esempio il GPS, il quale, però, può fallire in diverse situazioni, ad esempio in prossimità di strutture o edifici. Nella letteratura sulla navigazione di UAV autonomi, un'alternativa consolidata rispetto alla localizzazione basata sul GPS è il Visual-Inertial Simultaneous Localization And Mapping (VI-SLAM), in cui le posizioni e orientazioni del robot sono stimate utilizzando sequenze di immagini e misure inerziali ad alta frequenza. D'altra parte, il VI-SLAM è sensibile ai movimenti dell'UAV e le prestazioni di questo algoritmo, quando basato su telecamere, sono in generale fortemente legate all'aspetto visivo dell'ambiente. La seconda parte di questa tesi si concentra pertanto sul problema della pianificazione del percorso tenendo conto di questi limiti e problemi percettivi, con l'obiettivo di minimizzare l'errore nel calcolo della posizione del robot generando moti che considerano l'utilizzo delle telecamere come principale sensore per la navigazione. Ispirata dalla maturità della segmentazione semantica nella letteratura sulla Computer Vision, in cui lo scopo è quello di suddividere le immagini in gruppi di pixel semanticamente appartenenti ad una stessa categoria, questa tesi affronta il problema della pianificazione attiva utilizzando informazioni semantiche. In contrasto con l'ampio corpus di ricerca condotto sulle auto a guida autonoma, al meglio delle nostre conoscenze, questa è la prima volta che viene presentato un approccio di percezione attiva per gli UAV basato sulla segmentazione semantica.

Utilizzando l'annotazione semantica dell'ambiente di navigazione, il primo approccio che viene proposto incoraggia il robot a navigare su regioni visivamente affidabili, ad esempio strutture solide della scena, come gli edifici, evitando invece aree percettivamente degradate, caratterizzate, ad esempio, da un elevato dinamismo o da superfici riflettenti, come i bacini d'acqua. Questo concetto dell'utilizzo della semantica per migliorare l'affidabilità della navigazione basata sulla visione viene ulteriormente sviluppato in un secondo approccio, in cui una policy basata sull'apprendimento per rinforzo (Reinforcement Learning – RL) viene addestrata per identificare in tempo reale aree utili per il VI-SLAM durante l'esecuzione della missione. Questo permette agli UAV di adattare dinamicamente la propria traiettoria in base all'ambiente, migliorando le prestazioni in termini di robustezza e di errore di localizzazione.

Tuttavia, durante il dispiegamento in missioni reali, gli UAV di piccole dimensioni sono suscettibili a una serie di possibili eventi avversi, come forti raffiche di vento o guasti ai sensori, che possono portare a incidenti potenzialmente catastrofici. Pertanto, in queste situazioni è fondamentale renderli capaci di trovare un punto di atterraggio adeguato dove atterrare autonomamente. Nel tentativo di garantire la sicurezza del robot e dell'ambiente circostante, con particolare attenzione rivolta alle aree urbane, questa tesi si conclude con un approccio per l'atterraggio di emergenza di UAV multirotori autonomi utilizzando informazioni semantiche. Seguendo un paradigma di tipo RL basato sull'intelligenza artificiale, questo lavoro dimostra che le informazioni semantiche permettono di trovare più velocemente un punto di atterraggio, grazie alle associazioni implicite tra diverse classi (ad esempio, automobili e strade). Il sistema proposto può essere impiegato direttamente in esperimenti reali immediatamente dopo l'addestramento della policy, senza ulteriori aggiustamenti o adattamenti rispetto al dominio di utilizzo.

Prestando particolare attenzione alla coordinazione multi-robot e alla pianificazione attiva,

consapevole di limiti dei sensori di cui gli UAV sono equipaggiati, gli approcci e i sistemi presentati in questa tesi rappresentano un passo importante verso una navigazione aerea autonoma che può essere impiegata in complessi scenari reali. Inoltre, si dimostra che l'impiego della segmentazione semantica può essere estremamente vantaggioso per la pianificazione del percorso per i voli basati su dati sensoriali visivi e per l'atterraggio autonomo di emergenza. Questo porta a metodi più robusti, in grado di portare a compimento le missioni assegnate anche dove i sistemi più avanzati nello stato dell'arte falliscono, aprendo la strada a una navigazione autonoma dei robot più affidabile.

Acknowledgements

This doctoral thesis would have not been possible without the help, support and contributions from numerous people. First and foremost, I would like to express my sincere gratitude to Prof. Dr. Margarita Chli, director of the Vision of Robotics Lab at ETH Zurich, who gave me the rare opportunity to conduct my doctoral studies within the lab. Thank you for your guidance during my studies, your trust in my abilities, and for sparking my inspiration. I truly appreciate the freedom you gave me to pursue my own ideas, and your continuous and honest mentoring throughout the years, without which the results presented in this thesis would have not been possible. I would also like to thank Prof. Dr. Marco Hutter for his guidance, and to Prof. Dr. Kostas Alexis and Prof. Dr. Sebastian Scherer for accepting to co-examine this doctoral thesis, and for their contributions to the state of the art in the field of autonomous navigation and path planning, which were a great source of inspiration for my own research.

I would also like to thank the team of the Vision for Robotics Lab, who contributed to this thesis with invaluable inspiration and help with numerous experiments and publications. It was a remarkable experience to be part of such a talented team of researchers, and a pleasure working with all of you. I want to express special thanks to Dr. Lucas Teixeira for being such a great inspiration and friend, and for the great discussions we had, even beyond robotics. A special thank goes also to Dr. Marco Karrer and Dr. Patrik Schmuck, whose presence made demos and ARCHE events way more fun, and to Yves Kompis and Ruben Mascaro for the numerous collaborations we had. Furthermore, I want to express my gratitude to the construction robotics team, specifically Dr. Nitish Kumar, Dr. Abel Gawel, Dr. Timothy Sandy, Dr. Lionel Ott, Dr. Hermann Blum, Dr. Selen Ercan, Johannes Pankert and Florian Kennel-Maushart, for the many hours of fun we spent together preparing for demos. I also want to thank to the teams of the Autonomous System Lab and Robotic Systems Lab for the time we spent together building the robots for the summer schools. Finally, I would like to thank to all the people who reviewed this thesis and for their honest and useful feedback.

Lastly, I would like to express my deepest gratitude to my family and friends, who were always there when I needed them, especially to my mother, Attilia, and my father, Antonio, for their unconditional love, and for always having supported me pursuing my plans. Special thanks to my wife Laura, for your great support, love and understanding even in the toughest days of this journey. Your warmth, advice and help to put things in perspective are impossible to value. Lastly, I would like to thank my son Nicola, without whom the writing of this thesis would have taken a quarter of the effort and half of the time. Your smile reminds us of what is really important, and makes every day brighter.

June 4, 2023 *Luca Bartolomei*

Financial Support

The research leading to the publications and results presented in this thesis was supported by the HILTI Group.

That’s the whole problem with science.
You’ve got a bunch of empiricists trying
to describe things of unimaginable wonder

CALVIN AND HOBBES

Contents

Abstract i

Sommario iii

Acknowledgements vii

Preface 1

List of Acronyms 3

1 Introduction 5
- 1.1 Motivation . . . 7
  - 1.1.1 Robotic Collaboration . . . 8
  - 1.1.2 Aerial Navigation in GPS-Denied Environments . . . 8
  - 1.1.3 Emergency Landing of Multi-rotors . . . 9
- 1.2 Approach and Background . . . 10
  - 1.2.1 Multi-UAV Collaborative Exploration . . . 10
  - 1.2.2 Aerial Vision-based Path Planning using Semantics . . . 11
  - 1.2.3 Semantic-aware Emergency Landing . . . 13

2 Contribution 15
- 2.1 Research Contributions . . . 15
- 2.2 List of Publications . . . 23
  - 2.2.1 Publications Included in this Thesis . . . 23
  - 2.2.2 Other Publications . . . 24
- 2.3 Open-source Software . . . 25
- 2.4 List of Supervised Students . . . 25

3 Conclusion and Outlook 29
- 3.1 Robotic Collaboration . . . 29
- 3.2 Semantic-aware Aerial Navigation . . . 31

Paper I: Multi-robot Coordination with Agent-Server Architecture for Autonomous Navigation in Partially Unknown Environments 33
- 1 Introduction . . . 34
- 2 Methodology . . . 35
- 3 Experimental Results . . . 42

4 Conclusions . . . 45

Paper II: Towards Multi-robot Exploration: A Decentralized Strategy for UAV Forest Exploration 47
1 Introduction . . . 48
2 Related Works . . . 49
3 Methodology . . . 51
4 Experiments . . . 57
5 Conclusion and Future Work . . . 62

Paper III: Perception-aware Path Planning for UAVs using Semantic Segmentation 65
1 Introduction . . . 66
2 Related Works . . . 67
3 Problem Description . . . 68
4 System Overview . . . 69
5 Path Planning . . . 69
6 Experiments . . . 76
7 Conclusion and Future Work . . . 80

Paper IV: Semantic-aware Active Perception for UAVs using Deep Reinforcement Learning 83
1 Introduction . . . 84
2 Related Works . . . 85
3 Method . . . 86
4 Experiments . . . 92
5 Conclusion and Future Work . . . 96

Paper V: Autonomous Emergency Landing for Multicopters using Deep Reinforcement Learning 99
1 Introduction . . . 100
2 Related Works . . . 101
3 Methodology . . . 103
4 Experiments . . . 107
5 Conclusion and Future Work . . . 115

Bibliography 116

Curriculum Vitae 129

Preface

This doctoral thesis is structured as a cumulative thesis consisting of three chapters, while all the relevant papers forming the contributions of this research lie in the appendix. Specifically, Chapter 1 provides the context of this dissertation, introducing the central problems tackled in this work. Chapter 2 summarizes the context and the contributions of each published manuscript comprised in this thesis, detailing the interrelations between them. Chapter 3 presents an overview of our findings and suggests potential future directions of research to build upon this work.

List of Acronyms

CPU Central Processing Unit

FoV Field of View

GPS Global Positioning System

GPU Graphics Processing Unit

IMU Inertial Measurement Unit

LIDAR LIght Detection And Ranging

MPC Model Predictive Control

RL Reinforcement Learning

SaR Search and Rescue

SLAM Simultaneous Localization And Mapping

UAV Unmanned Aerial Vehicle

V-SLAM Visual SLAM

VI-SLAM Visual-Inertial SLAM

Chapter 1

Introduction

The great potential of autonomous robots to serve humanity in a variety of tasks, from infrastructure inspection in hazardous areas to post-disaster damage assessment, for example after a natural catastrophe as in Fig. 1.1, has been a key driving force for research into the automation of robot missions over the past decades. While each application has its own unique challenges, they share a common need for mobile robots to traverse and navigate their environment with some degree of autonomy [42, 142].

The most common approaches in commercially available robots are tele-operation and semi-autonomous navigation; however, these limit the applicability and scalability of robot-based solutions, as the constant attention of expert operators is indispensable. On the other hand, fully autonomous navigation implies addressing a number of challenges, such as building a robust and reliable robotic system with self-localization, mapping and path-planning capabilities, in order to ensure the accomplishment of the mission and the safety of the robot and its surroundings. These requirements become even more critical when robots are equipped with minimal sensor-suites, as in the case of small Unmanned Aerial Vehicles (UAVs) or drones. While autonomous cars or ground robots in general can carry powerful computers and Graphics Processing Units (GPUs), as well as a large number of sensors, including heavy and bulky ones, such as LIght Detection And Ranging (LIDAR) sensors, small drones have a limited payload; as a result, they can carry only lightweight, small lower-power computers and sensors. This factor is particularly impactful on state estimation and path planning, as the algorithms need to run in real time on Central Processing Units (CPUs) or on portable GPUs, under limited sensing capabilities and restricted computational capacity. Nevertheless, the prospect of autonomous aerial platforms, especially small multi-rotors, can potentially enable a broader range of tasks compared to ground robots. However, the UAV solutions available on the market mostly focus on consumer-level applications, such as recording of video footage, and limit the navigation of commercial drones to wide, open spaces, by employing sensing modalities based on Global Positioning System (GPS) receivers and ultrasonic proximity sensors. This prevents *de-facto* the usage of UAVs in more complex, cluttered, and potentially dynamic environments, which are typical in Search and Rescue (SaR) operations and inspection tasks, since GPS-based position estimates become unreliable when navigating in close proximity to general structures or tall buildings, due to the occlusion of the line-of-sight between the satellites and the receivers. Similarly, ultrasonic proximity sensors have limited mapping capabilities, as they can measure distances to an obstacle only in one direction and within a limited range. These shortcomings in the sensing modalities prevent UAVs from achieving full autonomy, as they restrict the robot's mapping and obstacle avoidance capabilities. To overcome these limitations, cameras have become the standard sensor of choice. While most commercially available drones use cameras predominantly to take pictures or filming, mainly due to the complexity of processing visual cues effectively, images carry extremely rich information that can be used to localize a UAV and create a map of its surroundings, which are key components for successful autonomous navigation. Additionally, camera frames can be used to extrapolate a higher-level semantic understanding of the scene [51], that is extraordinarily helpful to identify potential dangers in the vicinity of the robot, for example a moving obstacle.

From a localization standpoint, when using cameras, position estimation in a general environment can be achieved by motion. Using a sequence of overlapping frames, it is possible to estimate the pose of the camera at the time-instant when each image is being captured and the corresponding scene map; this technique is generally known as Simultaneous Localization And Mapping (SLAM). SLAM using visual cues is versatile, and it enables autonomous navigation in GPS-denied environments (e.g. indoors) or in the aforementioned situations where the GPS signal is noisy or unstable. In purely Visual SLAM (V-SLAM), the scene is perceived solely based on the stream of frames; however, this translates to sensitivity of estimates to illumination conditions, as well as to motion blur. To this end, it is now common practice to fuse image information with inertial measurements from Inertial Measurement Units (IMUs), to perform Visual-Inertial SLAM (VI-SLAM). Nonetheless, SLAM is intrinsically prone to drift over time; repetitive scene texture, reflective surfaces or moving objects (e.g. people, cars, trees moving in the wind) can accelerate this degradation process, possibly leading to pose estimation failures. This limitation can be alleviated by proper planning of viewpoints, that avoid directing the camera towards unreliable areas for vision-based localization; this process is commonly known as *active planning*, or *active perception* [8].

To widen the range of applications of VI-SLAM further, in recent years research effort has been focusing on developing SLAM systems able to accommodate for the presence of multiple robots in the scene employed for the same task [71, 123], opening new promising research avenues toward multi-robot (multi-agent) autonomous navigation. The potential of performing SLAM with multiple UAVs and fusing their experiences into a set of unified scene and pose estimates promises a multitude of advantages over the single-robot case. The most obvious benefit is the increased tolerance to single-robot failures (e.g. pose estimation failures; collisions), as well as the capability to cover larger areas of interest within a given time-frame, which is especially valuable in time-critical applications, such as SaR. However, generating a common map from shared experiences, co-localizing the agents in it, and coordinating their motions are rather complex tasks. The literature lacks valid and practical approaches for real-world missions, as the existing systems make unrealistic assumptions, such as the instant availability of data from different agents, unrestricted communication bandwidth and range, or existence of previously computed maps.

This thesis aims at addressing the aforementioned challenges of autonomous navigation, with

Figure 1.1: Illustration taken from [151], showing a UAV inspecting collapsed buildings during a SaR operation after the earthquake happened in Nepal in 2015.

focus on multi-UAV coordination and single-agent vision-based active planning. The first half of this thesis focuses on studying different coordination strategies for robotic teams in the context of exploration of partially or completely unknown environments, targeting applications such as industrial inspection and SaR, while in the second half, the core contributions concentrate on exploiting the power of semantic information to enable active and effective path planning for resilient decision-making during autonomous UAV navigation and emergency landing.

1.1 Motivation

The research conducted over the course of this thesis aims at advancing UAV autonomy in general settings under realistic flight conditions. We focus in particular on three main applications, namely robot collaboration, aerial navigation in GPS-denied environments, and emergency landing of multi-rotor UAVs. While these applications require a wide range of different capabilities in order to realize them, the core capacity of performing real-time planning and decision-making is a common, key requirement across all of them. In the remainder of this section, these three applications are discussed in more detail as to the corresponding challenges and the potential impact in advancing UAV and robotic autonomy in general.

1.1.1 Robotic Collaboration

The deployment of multiple robots in tasks requiring large areas to be covered can lead to massive speed-ups of missions. This aspect is particularly crucial in time-sensitive situations, for example rescue operations, where completion times need to be kept as low as possible, rendering multi-robot systems particularly impactful in such cases. Similarly, aerial inspection and image capturing for 3D reconstruction can greatly benefit from the presence of multiple robots, especially when large-scale structures needs to be inspected or reconstructed. Inspired by the potential offered by agile platforms such as UAVs, a substantial part of the research effort in this thesis has been dedicated to addressing such scenarios in a multi-agent setting, with particular focus on the capacity of drones to perform high-speed flights.

Multi-robot collaboration introduces a series of new challenges for co-localization [123] and planning [6, 25]. From the planning perspective, it is fundamental to develop strategies to avoid collisions between agents and the scene [69], while minimizing duplicated work [89]. To reach such levels of coordination, co-localization, together with a common map built from the shared experiences of the agents, is a key component of effective multi-robot systems, as robots can collaborate more efficiently if they know the current pose and the past trajectories of the components of the team. In the last couple of decades, the research efforts in the path-planning community have been targeting mostly the coordination problem, under the assumption of co-localized robots [20, 150], while research on collaborative SLAM was exclusively focused on state estimation [71, 123]. Recently, solutions addressing these challenges in unified frameworks emerged [157, 164, 166]; however, these systems generally require restrictive assumptions limiting their applicability in the aforementioned scenarios, for example stable long-range communication, or valid visual co-localization between agents, even when they are far away from each other. This does not hold true when navigating cluttered, complex scenes, such as forests, where the line-of-sight between agents can be obstructed frequently. Moreover, in such environments, these methods do not fully exploit the capacity of UAVs to perform high-speed flights, leading to slower and more conservative maneuvers, affecting the overall time performance of the multi-robot system negatively.

1.1.2 Aerial Navigation in GPS-Denied Environments

The prospect of automating tasks such as industrial inspection, cave exploration, 3D reconstruction or goods delivery, implies the need of aerial robots able to reliably localize in a variety scenes. While this can be generally achieved in wide, open outdoor environments using GPS, when flying close to structures or indoors, robot self-localization cannot rely on external global positioning [35, 38]. Similarly, adverse weather and jamming can significantly degrade the GPS signal, leading to poor localization of the UAV and, consequently, endangering the robot and its surroundings. In such cases, the pose of the agent needs to be estimated from other onboard sensor modalities, e.g. via VI-SLAM [23] as described at the beginning of this chapter. This is a well established approach, and it is also adopted in high-end commercial drones, such as the DJI Mavic[1] and the Skydio 2X[2], which are able to perform obstacle avoidance in uncluttered scenes by relying on visual-inertial sensor measurements.

---

[1] `www.dji.com/mavic`

[2] `www.skydio.com`

However, visual-inertial state estimation, despite its flexibility, is not only prone to drift and increasing uncertainty over time, it is also extremely sensitive to the motions of the sensor-suite [98], as well as to the appearance of the environment to traverse [162]. For example, fast movements or in-place rotations can lead to blurry images and accentuate the drift in the pose estimates [44]. Similarly, the accuracy of VI-SLAM degrades significantly in the presence of dynamic objects, or when navigating areas with no texture [33] or textures exhibiting specularities. All these factors can potentially lead to localization failures and, consequently, to early abortions of the mission. These limitations are accentuated in situations such as SaR or inspection of post-disaster scenes, as in these scenarios there is no general *a priori* knowledge of the environment, and the supervision of an expert human pilot is not possible for safety reasons, e.g. when exploring a contaminated area [85]. Adding to these challenges, another important aspect in vision-based navigation is the narrow Field of View (FoV) of the sensors UAVs are commonly equipped with, such as stereo-vision set-ups or RGB-D cameras, providing color and depth information. This poses considerable safety concerns, as the low FoV, together with the limited measurable depth range, implies that most of the scene is out of the sensing distance of the platform. Driven by these limitations, a big part of this thesis is dedicated to addressing these challenges, designing a navigation stack that, thanks to the incorporation of semantic information during planning, enables UAVs to fly reliably in complex scenes, by relying exclusively on visual cues and limited-range depth measurements.

1.1.3 Emergency Landing of Multi-rotors

Despite the wider range of applications compared to ground robots, a disadvantage of UAVs is that they cannot simply stop navigating in case of an emergency, but they need to identify and reach a safe landing spot in their vicinity as swiftly as possible. This limitation is extremely important, as several dangerous events can occur during deployment, for example, strong wind gusts, cold weather, which can unexpectedly shorten the battery life, mechanical malfunctions or localization failures. This aspect is particularly crucial for navigation in urban settings, as not only the integrity of the drone and the landscape needs to be preserved, but also moving cars and crowded spaces must be avoided. Commercial drones, since they are designed to fly over populated areas, remain capable of landing in such critical situations, even with reduced battery voltage or partially working rotors [94]. However, they are generally not able to autonomously detect a landing area, and implement only basic safety features, such as GPS-based return-to-home options, or landing on fiducial markers placed by an operator (Fig. 1.2). On the other hand, these strategies are not suitable for all emergency situations, for example in long-range navigation. In the literature, more advanced solutions estimate a 3D map of the environment by means of depth measurements to identify a safe landing area [46, 87, 90]; nevertheless, given the limited sensing capabilities of UAVs described in the previous section, such as short detection range and narrow FoV of depth sensors, these methods generally underestimate sensor noise and tend to fly the drone dangerously close to obstacles.

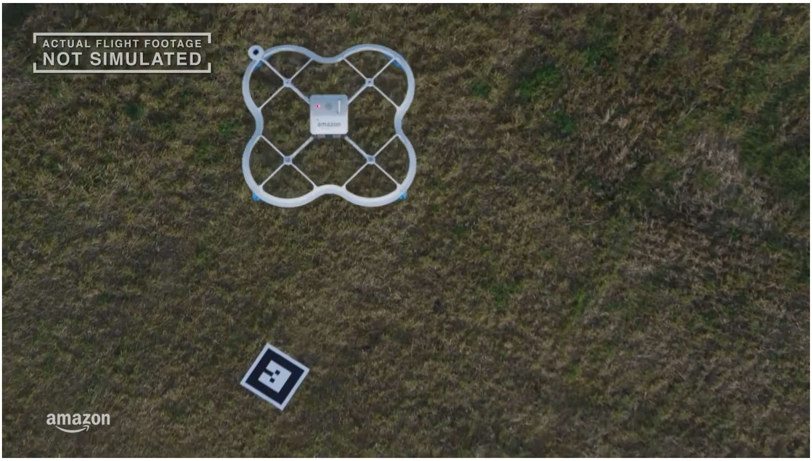


Figure 1.2: Illustration taken from [53], showing a top view of an Amazon drone landing on a fiducial marker placed by the customer.

1.2 Approach and Background

Motivated by the aforementioned application areas and challenges, the research in this thesis strives to advance the state of the art in autonomous aerial navigation. With focus on multi-UAV collaborative exploration, aerial vision-based navigation and emergency landing of multi-rotor UAVs, this section provides a summary of the approach followed, as well as some context on the topics addressed in this thesis.

1.2.1 Multi-UAV Collaborative Exploration

Robotic exploration is the task of mapping unknown environments using one or more mobile robots. The main objective of this process is tightly coupled with the particular application at hand; for example, in SaR missions, time efficiency is the key driving force [28, 37], while other tasks place more emphasis on precise and complete reconstruction [77, 121, 126]. In this context, multi-robot systems have obvious benefits, since deploying multiple robots allows to cover large environments faster, proving at the same time robustness to single-robot failures [14]. On the other hand, coordinating the motions of many agents is greatly challenging from a co-localization and a planning perspective, as described in the previous sections. To tackle these problems, two types of coordination paradigms exist; namely the centralized and decentralized strategies. While in centralized approaches the agents share their experiences with the other robots via a common middle point [20, 40, 43, 92, 157], such as a ground station or a server on the cloud, decentralized solutions are based on direct peer-to-peer communication [31, 130, 156, 164].

Centralized architectures gather all the available information provided by the robots and the environment at one location, the server, translating to better coordination of the robotic team, generally by means of a global planner [40]. This implies the assumptions that the poses of the robots are available at all times and that a consistent, global map can be built from sensor measurements directly [61, 92]. While simplifying the overall problem, these presumptions are restrictive, as in reality all state estimates are subject to drift, which is an important aspect for planning in multi-agent settings, since robots need not only to steer away from obstacles, but also avoid collisions between each other. Tackling these limitations in the context of post-disaster assessment, in *Paper I* we propose a centralized multi-robot architecture able to perform swarm state estimation and coordination, with the objective of building a globally consistent 3D reconstruction of the area of interest.

An important aspect to consider is that, by design, centralized systems suffer from poor scalability, as the amount of data to process increases with the number of agents. Moreover, since planning is performed in the central unit, it is generally assumed that communication with the server is always possible, regardless of the conditions and characteristics of the environment. In this regard, decentralized approaches [156, 164] are more robust, since, given the nature of these methods based on direct peer-to-peer communication, each agent is able to operate independently from the other robots in the team. This advantage comes at the cost of higher difficulty of coordinating the agents, as they have access to local information only. Moreover, each robot needs to plan its next action relying on incomplete knowledge about the future moves of its peers, while at the same time considering additional constraints, for example maximum communication range. Together with these limiting factors, another important aspect that can potentially lead to worse exploratory performance is that the vast majority of exploration strategies do not fully exploit the capacity of aerial robots to fly at high speed [60]. As exploration is an intrinsically dangerous task, where the robots need to navigate in the vicinity of the boundaries between known and unknown space, a cap on the velocity of each agent is imposed for safety reasons; consequently, most path planners, with very few exceptions [28, 165], plan conservative maneuvers, leading to start-and-stop motions [15]. This effect is exacerbated in complex, cluttered and obstacle-dense environments, such as forests, which are typical in SaR scenarios.

Inspired by these challenges, this thesis proposes a multi-robot decentralized exploration strategy for UAVs (*Paper II*) that is capable of fast coverage of forest areas by exploiting the platform's agility, while considering the peer-to-peer communication constraints. The proposed system sets out the scaffolding towards decentralized multi-robot high-speed exploration of cluttered environments.

1.2.2 Aerial Vision-based Path Planning using Semantics

For successful autonomous navigation in unknown and dynamic environments, a robot must map the scene with sufficient accuracy to avoid collisions with obstacles, and at the same time localize in the built map. To this end, VI-SLAM can localize the robot and estimate a map of the environment at once, and represent a valid alternative to GPS-based navigation. However, due to the limitations of visual-inertial SLAM introduced in the previous sections, planning of the motions of the robot (and consequently, of the cameras) is crucial, especially in situations

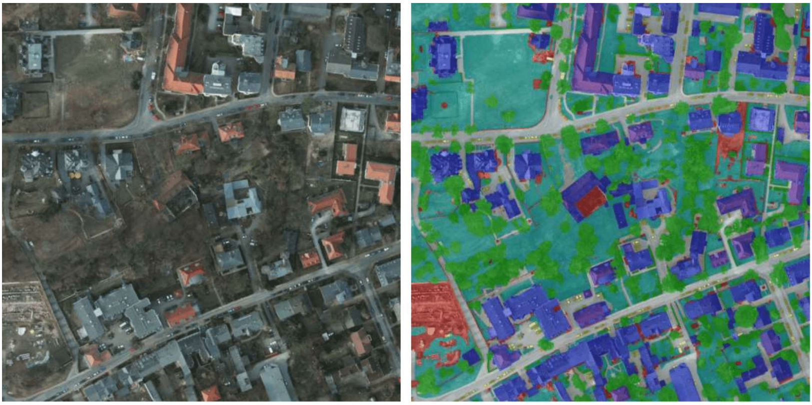


Figure 1.3: Semantic segmentation on a satellite image of an urban area. Given the input RGB image on the left, the pipeline assigns a semantic class to each pixel (e.g. building, road, vegetation). This thesis shows how semantics can be used to fly a UAV over visually informative areas, demonstrating the benefits of using semantic information in the context of active vision-based aerial planning. Illustration taken from [93].

characterized by visual degradation, for example in the presence of moving objects or textureless areas. This problem is at the core of the research field of active planning, which advocates that sensory performance can be improved by proper selection of motion-control actions [8]. While existing works propose solutions integrating planning and perception in unified frameworks [2, 66], the complexity of such systems quickly explodes, motivating the need for more efficient approaches.

Recently, perception-aware planning strategies targeting multi-rotor UAVs emerged [5, 33, 104, 162]. While more sophisticated solutions compute the uncertainty of the poses of the robot using photometric information [33, 104], a common and simpler approach is to plan trajectories using the concentration of map points as estimated by the SLAM system [5, 162]. Nevertheless, these methods, while allowing for more robust vision-based aerial flights, are limited to avoiding textureless areas, not dealing with the complexity in scenery that can be found in a real-world mission. Driven by this limitation, in this thesis we advocate in favor of semantic segmentation for perception-aware path planning (Fig. 1.3), as semantics can be extremely helpful to identify areas of the environment that are unsuitable for robust localization, for example due to dynamic objects, or specularities. In semantic segmentation of visual cues, the objective is to assign a semantic label (e.g. building, car, road, tree) to each pixel of an input image, generally using deep learning techniques [59]. Despite the considerable level of maturity reached by research in the field [51], only few works study the benefits of using semantics for aerial path planning [75, 86, 120, 133], whereas many examples are available for autonomous cars [96, 107, 125]. Here, we aim at filling this gap, by proposing valid, applicable solutions for autonomous flights of multi-rotor UAVs when relying on visual localization (*Paper III*, *Paper IV*). Inspired by the advances in real-time semantic segmentation for small embedded platforms, such as drones [147], *Paper III* proposes a planning scheme encouraging navigation over informative regions of the space to enhance the performance of VI-SLAM, while *Paper IV* extends this idea further, by introducing a data-driven approach based on Reinforcement Learning (RL) pushing for even more robust vision-based navigation systems.

1.2.3 Semantic-aware Emergency Landing

With their potential utility for a wide range of applications, it is fundamental to maintain a high level of safety in order to exploit the full benefits of autonomous UAV operations [152]. Given the large number of possible threats during flights, such as adverse weather conditions, communication losses, or sudden drops in battery level, automating the landing of aerial platforms is crucial, especially when navigating urban areas. However, autonomous landing implies the need of identifying, approaching and landing at a safe site without human intervention. These actions are particularly challenging, especially in the case of small UAVs carrying lightweight sensor rigs.

Driven by the limitations of commercial solutions described in Section 1.1.3, research efforts have mostly been concentrating on how to control a drone to touch the ground safely under uncertain dynamics or under-actuated systems [94, 109, 127], while other works aim at landing a UAV at the take-off location without the need of fiducial markers [111]. Another line of research focuses instead on the detection of a suitable landing site by relying on depth measurements [46, 90], for example, with the objective of identifying free, flat areas. These solutions generally require the creation of a reliable 3D map online, which, however, can be particularly challenging, especially when navigating at high altitudes. More importantly, these strategies reason only about the geometry of the candidate landing sites, and dismiss more general and relevant context, ignoring that, while some areas can be identified as clear spaces, they are unsuitable for landing, such as rooftops and roads. Tackling this limitation, some works propose to discriminate the possible landing spots using binary classification by training convolutional neural networks [58, 137]. Nonetheless, these approaches have been mostly validated in simulations or synthetic datasets, with very few works demonstrated to work in real-world experiments [47].

In this thesis, we aim at tackling these limitations, by proposing a semantic-aware pipeline for autonomous vision-based landing of UAVs. Here we study how to exploit the power semantics to speed up the identification of safe landing areas, by profiting from the high-level spatial relationships between different semantic classes (*Paper V*). The proposed solution, demonstrated to work in real-world scenarios, fills the existing research gap, and represents an important step towards more reliable autonomous UAVs.

Chapter 2

Contribution

This chapter details the core contributions of the research conducted during this doctoral thesis. In Section 2.1, we present the context relating every publication to the state of art. A summary of each individual work, as well as the research contributions and the interrelations between the publications are provided. The complete list of publications is reported in Section 2.2, while Section 2.3 reports the open-source software libraries resulting from the papers contributing to this thesis. At the end of this chapter, Section 2.4 provides a list of all student projects supervised in the course of the doctoral studies.

2.1 Research Contributions

Paper I

Luca Bartolomei, Marco Karrer, and Margarita Chli. Multi-robot Coordination with Agent-Server Architecture for Autonomous Navigation in Partially Unknown Environments.
In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

Context

As motivated in Section 1.1.1, multi-UAV collaboration offers appealing advantages in tasks such as 3D reconstruction and exploration. To this end, there are works in the literature studying ways to coordinate multiple robots to complete a given mission within a limited time or resources budget [78, 160, 164], for example, by building coordination diagrams [105, 136] or by considering the joint velocity-time space of the agents [144]. These works concentrate exclusively on the coordination task, and assume the availability of the robots' ground-truth poses [141, 153], ignoring the challenges of co-localization [71, 122, 123] and mapping [81] with multiple robots. However, in order to have effective exploration planning [20, 155] of unknown environments under realistic conditions, such as drifting pose estimates, it is imperative to have a complete framework coupling multi-agent state estimation and coordination [164, 166]. Nonetheless, creating effective multi-UAV systems is far from trivial, since the localization, mapping and planning components of the pipeline need to be designed considering the limitations of aerial platforms introduced in Chapter 1. Aiming at filling in the lack of practical, valid solutions in the state of the art at the time, *Paper I* proposes an architecture composed of a central server and a team of robots, able to perform multi-robot co-localization, mapping and path planning in a unified framework.

Contribution

Aiming at generating 3D digital models of large-scale structures of interest, this paper proposes a centralized multi-robot architecture, composed of a server and a team of UAVs, or agents. Here, we assume that a human operator assigns a series of target positions to each robot using prior knowledge about the area of interest, for example from satellite imagery. Moreover, we assume that each agent is equipped with a GPS receiver, as well as a sensor suite capable of running VI-SLAM [116] and estimating a local, dense 3D map of the environment for obstacle-avoidance purposes [102].

The key contribution of this work is the design of a complete, multi-agent pipeline for inspection and exploration tasks, allowing to drop some of the typical assumptions in the multi-robot planning literature, i.e. the perfect knowledge of the agents' poses and the existence of a prior ground-truth map of the environment. To this end, we collect the experiences of the robots at a central server. Adopting a pose-graph formulation for sensor fusion [55], we obtain the agents' poses in a common GPS-based reference frame, while at the same time building a globally consistent 3D reconstruction of the structure of interest to inspect. This map is generated online on the server using the robots' optimized poses as well as their raw sensor measurements, and is then fed into a hierarchical global planner, which aims to coordinate the swarm and deconflict potentially colliding agents, guiding the UAVs to navigate towards their respective destinations performing obstacle avoidance, locally. The proposed pipeline is tested in simulation with robotic teams composed of up to four drones, reconstructing a large-scale industrial scene by relying on onboard sensor measurements. Furthermore, with the objective of facilitating and accelerating scientific progress, our design is released as open-source code.

Interrelations

Similarly to *Paper II*, this paper studies the multi-robot coordination problem; however, while *Paper II* focuses more on the path-planning aspect, the main contribution here is on the design of a multi-agent system comprehensive of state-estimation and coordination capabilities. Nonetheless, the assumptions of GPS signal availability and the continuous communication between the agents and the server may be sufficient for open spaces, but can potentially limit the applicability of the system in more complex, cluttered scenes, such as forests. Furthermore, the proposed planning approach suffers from the typical scalability issues of centralized architectures, arising when larger team sizes are deployed [72], and requires an initial guidance from an operator. These limitations are addressed in *Paper II*, where a decentralized multi-robot architecture is proposed.

Paper II

Luca Bartolomei, Lucas Teixeira, and Margarita Chli. Towards Multi-robot Exploration: A Decentralized Strategy for UAV Forest Exploration. Submitted to *IEEE Robotics and Automation Letters (RA-L)*, 2023.

Context

In contrast to our previous work, where the objective is large-scale industrial inspection with multiple robots, this paper aims at fully autonomous exploration of forest areas with a small swarm of multi-rotor UAVs, eliminating the need for any guidance from a human operator. Driven by the agility of drones [60], here research effort concentrates on the minimization of the time required to cover a region of interest, exploiting the capacity of these platforms to perform high-speed flights. However, maintaining the safety of the UAV-robot is one of the main challenges in the exploration of unknown environments, since this task requires navigation in the vicinity of frontiers, defined as the boundary between known and unknown space [155]. This effect is exacerbated by the sensing modalities typically used onboard small UAVs, such as stereo or RGB-D cameras, with limited detection range and a narrow FoV. Consequently, in order to avoid getting stuck within a confined space or performing aggressive dodging maneuvers to avoid collisions, most path-planners generate conservative start-and-stop motions, not fully exploiting the agility of these platforms [15, 77, 121].

While there are few works that aim at tackling this limitation [28, 165], a large body of literature mostly focuses on exploring large spaces, such as industrial scenes, driving the exploration process to focus on wide, unexplored areas. When navigating in particularly cluttered environments, such as forests, this strategy is not effective anymore, as many thinner trails of unknown space can be left unexplored, imposing the need for a second sweep of the environment over mostly explored areas. Multi-robot extensions have also been proposed with the objective of mitigating these issues and pushing for faster coverage [32, 92, 140]. Nevertheless, they focus more on the coordination aspect and suffer from the same limitations as the single-agent counterpart. In this paper, we aim at tackling these challenges in the context of single- and multi-UAV forest exploration.

Contribution

The core contribution of this paper is in the proposal of an effective strategy for high-speed exploration of unknown, cluttered environments using autonomous multi-rotor UAVs. The objective is to exploit the agility of these platforms, in order to reach complete coverage, while minimizing the time required to complete the mission. This task is particularly challenging for forests, since these scenes are characterized by a high number of occlusions, caused, for example, by tree trunks, branches and shrubs, which can lead to the generation of many smaller, less informative frontiers. As mentioned above, this effect is exacerbated when a robot is equipped with lightweight depth sensors with limited sensing range, such as stereo or RGB-D cameras.

To overcome these limitations and push towards more agile flight maneuvers to exploit the UAV's agility, even in cluttered environments, this work proposes to balance cautious exploration where it makes most sense, for example when navigating in the vicinity of unknown space, and more aggressive maneuvers when traversing mostly explored areas. This strategy leads to navigation at consistently higher speeds than existing approaches [165], and, consequently, up to 65% shorter completion times for missions in single-agent settings.

Moreover, the proposed approach can be directly extended to multiple robots in a decentralized fashion, assuming that the poses of the agents are available at a constant rate, and that a maximum communication distance between UAV-robots exists. As robots need to exchange information in order to collaborate, such as poses and local maps, our strategy aims at keeping the UAVs within communication range, while encouraging collaboration via online assignment of the aforementioned exploration modes. The proposed approach is demonstrated to work with up to three robots in a large-scale forest, and it is shown to lead to a 40% increase in the explored volume compared to a multi-agent centralized planning solution for same time budget. Thus, this work opens up interesting future research directions for scalable, multi-UAV high-speed exploration of cluttered environments.

Interrelations

As in *Paper I*, this work aims at proposing a multi-robot coordination strategy; however, while *Paper I* concentrates on the design of a complete system, encompassing centralized state-estimation and path-planning capabilities, here the focus strongly shifts toward high-speed exploration. Nevertheless, while the proposed decentralized approach for coordination yields better results than the state of the art, this work relies on the assumption that the UAV-robots can co-localize within the area of interest, and that information can always be exchanged between agents if within communication range, regardless of the presence of obstacles potentially occluding the signal.

Paper III

Luca Bartolomei, Lucas Teixeira and Margarita Chli. Perception-aware Path Planning for UAVs using Semantic Segmentation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

Context

Motivated by the limitations of pose estimation based on GPS, the research community has been actively developing VI-SLAM algorithms, specifically targeting real-time capabilities for usage onboard aerial platforms [24, 116]. Despite the maturity reached by the field, the existing camera-based SLAM solutions suffer from a series of limitations that prevent the large-scale adoption of this type of localization system; namely, the necessity of favorable illumination conditions, a mostly static scene, and the absence of reflective surfaces. When these assumptions do not hold, VI-SLAM is subject to faster drift and the estimated poses can quickly become highly uncertain. In turn, this can lead to localization and, consequently, mission failures. This danger can be greatly mitigated by adopting planning strategies tailored to avoid ill-conditioned estimates, for example, by steering the UAV away from unsuitable areas for localization, such as lakes or roads during peak hours, i.e. areas that violate typical assumptions made in SLAM estimation. In these cases, the path followed by the robot has a crucial impact on the overall performance of the vision-based localization system.

This research field has come to be known as *active perception* [8], and builds on the idea that sensory performance can be improved by careful selection of control actions. To this end, various methods have been proposed over the years [2, 7, 36]. For example, common approaches model the problem as a Partially Observable Markov Decision Process (POMDP) [66] or plan in the robot's belief space (i.e. its state and associated covariance) to reduce the uncertainty on the estimates [18, 65, 114]. However, these methods are generally computationally intensive, and not suitable for real-time path planning and short-term obstacle avoidance. To fill this gap, perception-aware planners recently emerged [1, 33, 162], with the objective of proposing practical alternatives that can run onboard aerial platforms. Nevertheless, despite representing an important step toward reliable vision-based autonomous flights, these works are still immature, as they mostly concentrate on avoiding textureless areas, using as discriminative factors the appearance of the scene [33] or the concentration of map points estimated by a VI-SLAM system [73, 162]. Other approaches, instead, put more focus on online trajectory generation, to avoid, for example, losing track of map points [44] or generating motion blur in the images captured by the camera [98, 154]. In this paper, we aim at tackling these limitations by proposing an active-perception pipeline using semantic information to plan the motions of a UAV during vision-based navigation.

Contribution

Building on the ideas of active perception and perception-aware planning, this paper presents an approach for autonomous aerial navigation when relying on VI-SLAM for pose estimation. The objective is to reduce the drift in the estimates, while avoiding collisions with obstacles and flying the robot towards a predefined goal position in the scenarios, for example, of package delivery or frontier-based exploration. Driven by the maturity reached by semantic segmentation [51], and by relying on the assumption that some areas are more suitable for visual localization than others, this work has its core contribution in the use of semantics for the purpose of active perception. Since today there exists a plethora of algorithms able to detect problematic areas for visual localization, such as water basins [84], busy roads [83] or trees moving in the wind [17], the proposed planner is able to fly the robot over informative regions of space, favoring the triangulation of new map points while avoiding unreliable areas for vision-based SLAM, such as textureless areas or reflective surfaces. Here, thanks to the use of semantics, we address the main limitation of the state of the art in aerial path planning, namely the incapacity of considering high-level visual information about the navigation area. Following this paradigm, in a series of experiments where the robot is tasked to reach a target position, the proposed planner is demonstrated to yield better results in terms of pose error than a reactive strategy aiming at finding the shortest path to the destination, and the approach by [162], representing the state of the art at the time of publication of this work. More importantly, this paper represents one of the first works in semantic-aware planning for UAVs. In fact, this field is still in its infancy, especially when compared to the large body of research done on the incorporation of semantic information in motion planning for autonomous cars [107, 125].

Interrelations

This paper's goal is semantic-aware path planning, as in *Paper IV* and *Paper V*, and lays out the foundations for *Paper IV*. However, this work is limited by the assumptions that semantic classes can be classified with respect to visual localization in a binary fashion, as either trustworthy or unreliable; this limitation is addressed in *Paper IV*. Furthermore, since this work does not make any strict assumption about the environment or the task to perform, it can be potentially used for local planning and obstacle avoidance in the pipelines of *Paper I* and *Paper II*.

Paper IV

Luca Bartolomei, Lucas Teixeira, and Margarita Chli. Semantic-aware Active Perception for UAVs using Deep Reinforcement Learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021.

Context

As in the previous work, this paper also aims at performing real-time active planning using semantic cues. However, while *Paper III* demonstrates that semantic information can be extremely beneficial for avoiding areas unsuitable for vision-based SLAM, its approach requires an *a priori* manual assignment of informativeness scores to each semantic class, depending, for example, on its visual appearance. Despite being very practical, this method requires manual annotation, which can be extremely cumbersome when coping with a large number of classes. More importantly, this assignment of fixed scores does not allow the robot to adapt its behaviour online with respect to changes in the environment, potentially leading to sub-optimal action selection. While other works on planning using semantics have also been proposed [120, 133, 149], these limitations are not addressed. To fill this gap, here we propose a data-driven approach based on RL, capable of dynamically identifying reliable regions for localization by relying exclusively on semantic cues.

In the context of robotics, and path planning in general, RL is used to learn complex control policies [63] via trial-and-error, for example, for locomotion [45] or obstacle avoidance [68], at the cost of the engineering effort of designing suitable reward functions, and creating simulation environments to train the RL agents. With the advent of deep learning, research focus has been aggressively shifting towards *deep RL*, where policies are modeled as deep neural networks, opening up avenues for better generalization and applicability of RL agents to a wider range of applications. This work builds on the demonstrated capacity of these policies to map high-dimensional sensory inputs to actions, taking inspiration from autonomous driving [62, 74] and legged robotics [50, 159]. Nevertheless, despite the recent impressive advances in the field, there is a lack of (deep) RL-based solutions for active planning with UAVs.

Contribution

As in *Paper III*, this approach aims at reducing the uncertainty in the robot's poses estimated by VI-SLAM, while flying towards a destination. While the objective of encouraging navigation over well textured, static regions is the same, a key difference to *Paper III* is the adoption of a deep-RL policy as the main discriminator to identify reliable areas for visual localization.

Therefore, the main contribution of this work is in the proposal of the first, semantic-aware active perception planner for aerial navigation adopting deep-learning techniques.

Using semantically labelled images as input, the policy is tasked to assign online an informativeness value to each semantic class, thus addressing the main limitation of our previous work. These scores are used in an optimization-based path planner to generate trajectories in real time, effectively guiding the UAV towards its destination. The proposed method achieves great improvements in success rates and state estimation errors compared to a reactive planning strategy and the state-of-the-art approaches proposed in *Paper III* and [162]. A remarkable aspect of this work is that, thanks to the use of mid-level representations, such as semantic masks, the policy is able to reach convergence faster during training [27], as well as to generalize to previously unseen environments, despite being trained exclusively in non-photorealistic scenes. To this end, the proposed architecture decouples the problems of semantic segmentation and path planning, since learning from raw camera data directly requires an implicit semantic extraction step, which would take extremely long training time in an RL fashion.

Interrelations

This paper extends the semantic-aware planning solution proposed in *Paper III* to a system able to adapt the navigation mode online during the execution of a mission, depending on the semantic classes in view. Thanks to the use of deep-learning techniques, this work establishes a data-driven approach for active perception, which can also be used as a local planning strategy for the multi-robot systems of *Paper I* and *Paper II*. Furthermore, the method introduced here is similar to the approach of *Paper V*, where a RL-based system for semantic-aware path planning is also proposed. However, *Paper V* does not address the active perception problem, but instead focuses on the design of a landing pipeline for multi-rotor UAVs.

Paper V

Luca Bartolomei, Yves Kompis, Lucas Teixeira, and Margarita Chli. Autonomous Emergency Landing for Multicopters using Deep Reinforcement Learning. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

Context

Deploying small multi-rotor UAVs inherently implies a safety hazard since, despite the recent technological advances in autonomous flights, these platforms are still susceptible to conditions that can potentially endanger the robot and its surroundings; for example, adverse atmospheric conditions, cold weather, mechanical or localization failures. In these situations, existing autonomous drone systems attempt to remain capable of landing; however, they opt for simplistic strategies, for example, GPS-based return-to-home maneuvers or vertical, slow descent using proximity sensors to detect the ground, albeit without being able of assessing the suitability of the attempted landing spot. As a result, these naive approaches can severely compromise the safety and the autonomy of a drone. Driven by this limitation, researchers have devoted their efforts to providing safer and more widely applicable solutions to detect and approach a landing area by relying on onboard sensory information. Nonetheless, due to its criticality, emergency landing still remains a challenging task, since a large number of factors need to be accounted for, such as uncertainties in state estimation, the platform's dynamics as well as the presence of obstacles.

These issues have been previously addressed by proposing advanced control strategies [127, 146] based, for example, on Model Predictive Control (MPC) [91] or on model-based RL [109]. However, while these methods have been demonstrated to effectively control the drone during the descending maneuvers, they assume that the landing spot is given and is directly reachable; thus, they do not deal with the problem of detecting a suitable location to land.

To this end, the most common approach is to rely on fiducial markers [112, 113]; however, this requires a human operator to assign the landing area to the UAV, violating the autonomy of the drone and thus compromising the applicability of such an approach, especially considering emergency situations. Aiming to overcome this limitation, another family of approaches use more sophisticated vision-based pipelines [111], built on geometrical [158] or deep-learning techniques [58], relying on depth and, in some cases, semantic images [80, 137]. For example, the work in [46] generates an elevation map of the environment relying on short-range depth completion, while in [90] they identify a flat area to land on by relying on depth measurements directly. A concerning characteristic of these methods is the general assumption that the 3D information obtained from onboard depth sensors, such as stereo or RGB-D cameras, is accurate. This, however, does not generally hold in real flight conditions at medium-high altitudes, as the measurable depth range is limited to a few meters, and the uncertainty in the measurements grows quadratically with the distance [99]. To address this shortcoming, another line of research has been emerging using neural networks to classify candidate landing spots as safe or dangerous based on single images [115], but only few of these approaches have been validated on real robots [47].

Contribution

Inspired by the aforementioned limitations and driven by the lack of valid, practical solutions in the state of the art, this paper aims at proposing an emergency landing pipeline for small multirotor UAVs applicable in real-world missions. Following the paradigm introduced in *Paper IV*, this work designs a deep-RL approach using semantic and depth cues in order to identify and reach a safe landing spot in urban scenarios. To this end, we harvest the now well-known power of semantic information , enabling the policy to learn the underlying associations between different semantic classes (e.g. cars and roads), effectively accelerating the process of detecting valid landing spots, compared, for example, to other methods using binary classification to rigidly discriminate safe from hazardous areas.

Using semantics and depth, the learnt policy selects an action from a discrete set of possible movements, including lateral and vertical displacements, and communicates this to the MPC controlling the drone. To train the policy, the reward function is designed to encourage the UAV to fly on top of *safe* areas (e.g. grass, terrain), as well as to reach the ground as quickly as possible, by penalizing long flights over potentially dangerous regions, such as busy roads or crowded spaces. The RL agent is then deployed on a real platform directly after training in simulation, without the need of additional fine-tuning.

While the pipeline is agnostic to the semantic segmentation algorithm, in this work long-range depth information is obtained using depth completion methods [139], aiming at more reliable estimation of the geometry of the space beneath the robot. This design choice is dictated by the aforementioned limitations of range sensors, and allows to rely on a minimal sensor set-up composed of a single down-looking RGB camera. However, since UAVs are extremely constrained platforms in terms of payload and computational power, we propose to outsource depth completion and semantic segmentation to the cloud, as these are demanding tasks; nevertheless, the pipeline is designed in a modular fashion, such that the cloud can be easily replaced by a portable GPU carried onboard the drone.

The real-world experiments demonstrate that the proposed approach is capable of landing a drone safely on the ground in a real urban scenario, where other commercial[1] and state-of-the-art solutions [90] fail. Thus, this work represents a significant step towards more secure and reliable autonomous UAVs, as our approach is shown to land a drone respecting the necessary safety constraints in multiple challenging scenarios. Furthermore, we release the simulator, the training architecture and the policy design as open-source code with the aim of boosting future research in the field.

Interrelations

Similarly to *Paper IV*, this work proposes a navigation system using semantic information and following a deep-RL paradigm. However, in contrast to previous work, where RL is used in combination with a trajectory optimization scheme considering the presence of moving obstacles, here the problem of emergency landing is addressed using an end-to-end deep learning approach in a mostly static scene. Nevertheless, the pipeline proposed in this paper is a widely applicable framework, that can be used as safety feature, for example, in more complex multi-robot systems as in *Paper I*, where communication with a central server with high computational power, potentially able to run semantic segmentation and depth completion pipelines, is already required.

2.2 List of Publications

During the doctoral studies, the following publications were achieved, with invaluable contributions from the co-authors. Furthermore, the author had the opportunity to present some of these works at international conferences. We list the publications in chronological order, and provide links to the videos of the presentations when available.

2.2.1 Publications Included in this Thesis

- **L. Bartolomei**, M. Karrer, and M. Chli. Multi-robot Coordination with Agent-Server Architecture for Autonomous Navigation in Partially Unknown Environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.
  Video presentation.

- **L. Bartolomei**, L. Teixeira, and M. Chli. Perception-aware Path Planning for UAVs using Semantic Segmentation. In *IEEE/RSJ International Conference on Intelligent Robots*

---

[1] `https://px4.io/`

and Systems (IROS), 2020. Video presentation.

- **L. Bartolomei**, L. Teixeira, and M. Chli. Semantic-aware Active Perception for UAVs using Deep Reinforcement Learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021. Video presentation.

- **L. Bartolomei**, Y. Kompis, L. Teixeira, and M. Chli. Autonomous Emergency Landing for Multicopters using Deep Reinforcement Learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022. Video presentation.

- **L. Bartolomei**, L. Teixeira, and M. Chli. Towards Multi-robot Exploration: A Decentralized Strategy for UAV Forest Exploration. *arXiv preprint*, 2023.

2.2.2 Other Publications

- A. Gawel, H. Blum, J. Pankert, K. Krämer, **L. Bartolomei**, S. Ercan, F. Farshidian, M. Chli, F. Gramazio, R. Siegwart, M. Hutter, and T. Sandy. A Fully-Integrated Sensing and Control System for High-Accuracy Mobile Robotic Building Construction. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.

- H. Blum, S. Rohrbach, M. Popovic, **L. Bartolomei**, and R. Siegwart. Active Learning for UAV-based Semantic Mapping. In *RSS - Workshop on Informative Path Planning and Adaptive Sampling*, 2019.

- Y. Kompis, **L. Bartolomei**, R. Mascaro, L. Teixeira, and M. Chli. Informed Sampling Exploration Path Planner for 3D Reconstruction of Large Scenes. *IEEE Robotics and Automation Letters (RA-L)*, pages 7893–7900, 2021.

- Y. Kompis, **L. Bartolomei**, and M. Chli. Fully Autonomous Live 3D Reconstruction with an MAV: Hardware- and Software-Setup. In *International Conference on 3D Vision (3DV)*, 2021.

- M. Hüppi, **L. Bartolomei**, R. Mascaro, and M. Chli. T-PRM: Temporal Probabilistic Roadmap for Path Planning in Dynamic Environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

- D. Morilla-Cabello, **L. Bartolomei**, L. Teixeira, E. Montijano, and M. Chli. Sweep-Your-Map: Efficient Coverage Planning for Aerial Teams in Large-Scale Environments. *IEEE Robotics and Automation Letters (RA-L)*, pages 10810–10817, 2022.

- Y. Pan, Y. Kompis, **L. Bartolomei**, R. Mascaro, C. Stachniss, and M. Chli. Voxfield: Non-Projective Signed Distance Fields for Online Planning and 3D Reconstruction. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

2.3 Open-source Software

Some of the works developed within this thesis have been released publicly for free use by the research community. Namely, these are:

- The multi-robot coordination with agent-server architecture as described in *Paper I*.
  `https://github.com/VIS4ROB-lab/multi_robot_coordination`

- The semantic-aware emergency landing pipeline for multi-rotor UAVs as described in *Paper V*.
  `https://github.com/VIS4ROB-lab/multirotors_landing`

2.4 List of Supervised Students

During his doctoral studies, the author had the opportunity to supervise the following Master and Bachelor students at ETH Zurich. For projects that resulted in publications, the citation is given.

Master Theses

Master students, 6-month projects, full-time

- Proni, Federico (Spring 2018): “Autonomous Path Planning for 3D Reconstruction”

- Zuidema, Christoph (Fall 2019): “Autonomous Radiation Mapping using Multiple UAVs”

- Pensotti, Sara (Spring 2019): “Autonomous Multi-Robot Exploration of Partially Unknown Environments”

- Kuan, Benson (Spring 2019): “Semantic-based Path Planning for Autonomous Robot Navigation”

- Kompis, Yves (Fall 2020): “Informed Sampling Exploration Path Planner for 3D Reconstruction of Large Scenes” [77]

- Blass, Lukas (Fall 2020): “Centralized Collaborative Exploration for UAVs with Unknown Initial Pose”

- Selmonaj, Ardian (Fall 2020): “Reinforcement Learning for High-Accuracy Localization”

- Liechti, Tim (Spring 2021): “Traversability Analysis for Autonomous Robots”

- Kebler, Nils (Spring 2021): “Environment-aware Active Localization Using Reinforcement Learning”

- Pan, Yue (Fall 2021): “Voxfield: non-Projective Signed Distance Fields for Online Planning and 3D Reconstruction” [103]
- Morilla-Cabello, David (Fall 2021): “Sweep-Your-Map: Efficient Coverage Planning for Aerial Teams in Large-Scale Environments” [92]
- Hüppi, Matthias (Fall 2021): “T-PRM: Temporal Probabilistic Roadmap for Path Planning in Dynamic Environments” [64]
- Asikainen, Elias (Spring 2022): “Panoptic Segmentation and Volumetric Mapping in Dynamic Scenes”
- Bone, Sean (Fall 2022): “Subterranean Multi-Robot Exploration with Monte Carlo Tree Search”

Semester Theses

Master students, 3-4-month projects, part-time

- Sempertegui, Emilk (Spring 2019): “Optimization-based Path Planning for Autonomous Robot Navigation”
- Tearle, Benjamin (Spring 2019): “Collaborative UAV SLAM”
- Di Biase Troccoli, Giancarlo (Spring 2019): “Sampling-based Path Planning for Robot Navigation in Dynamic Environments”
- Mambelli, Davide (Spring 2020): “Multi-agent Path Planning for Active Sensings”
- Erne, Simon (Spring 2020): “Assisted Human Coordination for Robotics Teams”
- Andreae, Leonardo (Spring 2020): “Formalization and Implementation of a Path Planning Algorithm for Robot”
- Kämpf, Nadja (Fall 2020): “Dynamic Object Tracking for Semantic-based Path Planning”
- Colagiorgio, Aurora (Spring 2021): “Path-Planning for Autonomous Robot Navigation in Dynamic Environments”
- Bhardwaj, Arjun (Spring 2021): “Reinforcement Learning based Active Localization for Construction Robots”
- Regenass, Flavio (Fall 2021): “Improving Goal Selection for Next-Best-View Path Planning”
- Loi, Nicola (Spring 2022): “Autonomous Multi-Robot Exploration of Partially Unknown Environments”
- Wang, Han (Spring 2022): “Temporal Sampling-Based Algorithm for Motion Planning in Dynamic Environments”

Bachelor Theses and Studies on Mechatronics

Bachelor students, 3-month projects, full-time

- Weng, Zidong (Spring 2018): “Calibration of a Pan-Tilt Camera for Robotic Vision Tasks”
- Brunner, Maurice (Spring 2020): “Vision and LIDAR based SLAM for mobile robots”
- Wakonig, Theresa (Spring 2020): “Feature-based Target Tracking”
- Bossard, Anna (Fall 2020): “Autonomous Navigation between Fields with an Agricultural Robot”
- Lieberherr, Pascal (Fall 2020): “Autonomous Navigation between Fields with an Agricultural Robot”

Seminar in Computational Science and Engineering (CSE) in Robotics

Master students, 3-month projects, part-time

- Bone, Sean (Spring 2021): “Multi-Robot Exploration”

Chapter 3

Conclusion and Outlook

Driven by the potential impact of drones on a variety of tasks, such as aerial inspection and SaR, this doctoral thesis presents a set of contributions addressing autonomous aerial navigation of small multi-rotor UAVs. The focus is set on real-time path-planning approaches, as these are fundamental in enabling robots to traverse and navigate the environment with some degree of autonomy. To this end, the work in this thesis investigates multi-robot coordination with an explicit focus on rotorcraft UAVs (*Paper I*, *Paper II*), semantic-aware active perception for vision-based flights (*Paper III*, *Paper IV*) and emergency landing of multi-rotors (*Paper V*). While this research introduces novel concepts and is demonstrated to push the state of the art in autonomous aerial navigation significantly, several open questions remain before the proposed concepts can be widely deployed in practice. This chapter provides a brief summary of the conducted research and recapitulates the main insights and contributions of this thesis, presenting a discussion on the challenges and prominent avenues for future research.

3.1 Robotic Collaboration

In this thesis, multi-UAV collaboration is investigated, with the objective of improving the efficiency of inspection and SaR tasks. Aiming at the generation a 3D dense map of large-scale structures of interest, *Paper I* addresses swarm coordination by designing a centralized architecture for state estimation, mapping and path planning, and navigates the robots towards user-defined destinations, providing a complete, practical multi-agent pipeline. The proposed system allows to relax common assumptions in the multi-robot literature, for instance, the access to the UAVs' ground-truth poses, demonstrating the effectiveness of this approach in simulation with up to four UAV-agents. Nevertheless, even though this approach has also been tested in preliminary real-world experiments with two drones, a thorough analysis of the performance in real settings with large team sizes is subject to future work. Furthermore, due to its dependence on the presence of a reliable GPS signal, the pipeline cannot be employed in cluttered environments, where obstacles can potentially occlude the line of sight between the receiver and the satellites, leading to poor pose estimates. To overcome this limit, vision-based solutions can be adopted [71, 123], rendering the system more flexible and applicable to a wider range of scenes. To this end, to increase the level of autonomy further, the inspection process can be guided by automatic map splitting algorithms as in [106] or the necessity of pre-planned missions can be dropped altogether, opting, for example, for autonomous exploration solutions [20]. Nonetheless, coordination strategies in a centralized fashion as proposed in *Paper I* hardly generalize to large team sizes, since these methods suffer from poor scalability. As more robots imply bigger volume of data to process in the central unit, the computational power of the server represents the main bottleneck, while the number of agents does not affect the computational load of the single robots. This effect becomes particularly evident during global path planning, as the movements of the individual members of the swarm are all planned in a single, computationally demanding step, considering the simultaneous presence of all the UAVs. This is a well-known disadvantage of centralized architectures, and recent advances in the field of multi-robot coordination suggest valid alternatives able to tackle this challenge, proposing, for example, efficient decentralized deconfliction strategies [141, 153].

Similarly to our previous work in *Paper I*, *Paper II* also addresses multi-UAV navigation; however, instead of concentrating on system design, here research efforts focuses on collaborative high-speed exploration of complex, cluttered forest areas, with the objective of minimizing the time required to complete coverage missions. While forests are very common in SaR scenarios, they still represent a particularly difficult challenge for autonomous robotic exploration, since, due to the large number of obstacles and occlusions caused by trees and their branches, classical frontier-based strategies are inefficient, continuously switching between accelerations and decelerations of the robot, and requiring multiple passes over the same area to explore it entirely. In an effort to address these limitations, the proposed approach attempts to balance cautious exploration of large portions of unknown space, and aggressive exploitation of already explored regions to profit from the agility of a UAV-robot and boost the efficiency of the mission. Thanks to the high speeds consistently reached by the robot flying along smooth trajectories, this approach is shown to lead to better performance in map-coverage tasks than the state of the art, and it can be extended to accommodate for the presence of multiple UAVs, coordinated in a decentralized fashion. Additionally, an important side effect of this strategy is the improvement in battery life compared to other planners characterized by start-and-stop motions, as these quickly drain the battery due to the high energy required to accelerate and decelerate the platform.

Nevertheless, the proposed approach assumes the UAVs are co-localized in a common reference frame, and expects the agents to stay within communication range, in order to exchange information and coordinate their motions. This last requirement poses restrictions to the movements of the robots, especially in cluttered scenes as forests, since the large number of obstacles, such as trees, bushes or branches, might cause the UAVs to diverge from each other, degrading the overall exploration performance. Consequently, even though the proposed method is demonstrated to scale up to three agents, this indicates that high-speed exploration of cluttered environments under communication constraints is a particularly difficult task to scale to large team sizes. In a next big step, aiming at the generalization of the approach towards a larger fleet, the coordination and message-passing strategies require deeper investigation, and the applicability of the exploration strategy in a real scene, using, for example, collaborative VI-SLAM for multi-robot co-localization, needs to be put to the test.

3.2 Semantic-aware Aerial Navigation

Despite the recent technological advances towards autonomous navigation of multi-rotor UAVs, state-of-the-art and commercial solutions lack the robustness required for a large-scale adoption of these robots as the standard choice for tasks, such as inspection and goods delivery. This thesis aims at addressing this issue with a strong focus on vision-based flights, where the robot's pose is estimated via VI-SLAM. While this localization paradigm allows to drop the dependence on GPS, autonomous navigation exclusively relying on visual cues is still an open challenge, as (a) cameras have a fixed rate, so aggressive motions can cause motion blur, and (b) the performance of VI-SLAM strongly depends on the appearance of the environment to navigate. With these shortcomings in mind, this thesis aims to mitigate them within the path-planning pipeline, pushing perception-aware planning for boosting robotic autonomy.

With the objective of limiting the drift in pose estimates for a UAV navigating towards a pre-defined destination, *Paper III* and *Paper IV* propose a new concept of path planners, using semantic information extracted from images to identify suitable areas for vision-based localization. While the idea of semantic-aware motion planning is a well-studied problem in autonomous driving, this is largely underexplored for autonomous flying platforms; therefore, these works represent some of the first approaches in this field. We identify the reasons for this gap in the facts that (a) path planning for ground vehicles is a simplified version of the problem, as cars must follow the navigation rules of the road traffic, and that (b) the number of semantic classes relevant for autonomous cars are limited to a few examples (vehicles, bicycles, pedestrians), while for aerial navigation this number is potentially unbounded and changes depending on the scenario, as, for instance, semantic classes in man-made environments are different from the ones found in natural scenes. To this end, *Paper III* proposes a perception-aware planner for natural outdoor scenes where an optimization-based planner generates trajectories towards the destination, steering the UAV away from potentially degrading areas for VI-SLAM, such as reflective surfaces (e.g. lakes), while avoiding obstacles and respecting the platform's dynamic limits. In terms of localization error, this method, despite its simplicity, greatly outperforms a classical planning approach and the state of the art by manually assigning an informativeness score to each semantic class, encouraging the robot to navigate over visually reliable regions. *Paper IV* extends this concept further, by using a deep-RL policy to assign these scores to semantic classes online. This renders the UAVs able to adapt dynamically to the scene in view, and to balance more aggressive flights towards the destination and more cautious maneuvers over visually trustworthy areas, improving the robustness of vision-based flights compared to *Paper III* and [162]. These results are a strong indication that semantics are an extremely valuable resource for autonomous, reliable, vision-based navigation.

This aspect is also underlined in *Paper V*, addressing the problem of emergency landing of a small multi-rotor UAV in urban scenarios. In this work, similarly to *Paper IV*, a data-driven approach is adopted; however, here we propose an end-to-end deep-learning pipeline, where an RL policy maps semantic and depth images to high-level command actions, which are then utilized by an MPC framework to control the movements of the UAV. Thanks to the use of semantic information, our method reaches in landing tasks double success rate than the state of the art, and accelerates the search of a safe spot by up to 150%. More importantly, the proposed strategy, despite being trained exclusively in simulation, is shown to be directly deployed successfully in real-world experiments, where other commercial and state-of-the-art solutions fail.

Despite the promising results presented in this thesis, developing a general semantic-aware navigation pipeline is still an open challenge that limits the adoption of these solutions in consumer-level platforms. This limitation is evident in the RL-based approaches of *Paper IV* and *Paper V*, as, since these works target urban scenarios, the policies are exposed to a limited set of semantic classes during training, namely, buildings, roads, cars, vegetation and terrain. This renders the proposed solutions potentially incapable of generalizing to other types of environments, implying the need for re-training the RL agents. To address this issue, policies can be trained incrementally by exposing them to a wider set of environments over time. However, this approach, commonly known as curriculum learning in the RL literature, implies longer training times, depending on the number of scenarios of interest. To this end, a possible approach is continual learning [79], known also as Incremental Learning or Life-long Learning, where the deep neural network underlying the policy is first trained in a large number of scenes sequentially, and is then adapted to new environments online during deployment. However, in the context of aerial navigation, this method raises some safety concerns, as it might not be possible to fine-tune the policy to generalize it to every scene, potentially leading to unpredictable decision-making. This problem can be addressed from a safe reinforcement learning perspective [57], where the objective is to ensure reasonable system performance, while respecting safety constraints during the learning and the deployment phases. Finally, to further improve the robustness of semantic-aware planning solutions, research effort can also focus on the incorporation of the uncertainty scores associated to each label in the semantic masks. However, for an RL-based approach, a per-pixel uncertainty value can enlarge the state space tremendously, rendering the process of learning a valid policy extremely hard.

In conclusion, while there remains a multitude of open questions before complete autonomy in aerial navigation can be achieved, to this end, this thesis presents methods and insights pushing the state of the art in multi-robot coordination and perception-aware active planning for UAVs forward, contributing towards autonomous aerial navigation deployable in challenging real-world scenarios. In the subsequent chapters, the papers discussing the main contributions of this thesis appear.

Paper

Multi-robot Coordination with Agent-Server Architecture for Autonomous Navigation in Partially Unknown Environments

Luca Bartolomei, Marco Karrer and Margarita Chli

Abstract

In this work, we present a system architecture to enable autonomous navigation of multiple agents across user-selected global interest points in a partially unknown environment. The system is composed of a server and a team of agents, here small aircrafts. Leveraging this architecture, computation-ally demanding tasks, such as global dense mapping and global path planning can be outsourced to a potentially powerful central server, limiting the onboard computation for each agent to local pose estimation using Visual-Inertial Odometry (VIO) and local path planning for obstacle avoidance. By assigning priorities to the agents, we propose a hierarchical multi-robot global planning pipeline, which avoids collisions amongst the agents and computes their paths towards the respective goals. The resulting global paths are communicated to the agents and serve as reference input to the local planner running onboard each agent. In contrast to previous works, here we relax the common assumption of a previously mapped environment and perfect knowledge about the state, and we show the effectiveness of the proposed approach in photo-realistic simulations with up to four agents operating in an industrial environment.

Published in:
IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2020
DOI: 10.1109/IROS45743.2020.9341367

1 Introduction

The growing popularity of the use of Unmanned Aerial Vehicles (UAVs) in tasks, such as exploration of unsafe areas, inspection, and search-and-rescue missions has been driving research towards their autonomous navigation. As aerial perception and path planning have become increasingly robust, small UAVs have been demonstrated to successfully plan their path and fly autonomously in some scenarios (e.g. [138]). However, since computational power and payload become the limiting factors when planning a mission with small UAVs, crucial choices regarding the sensor suite to be carried, as well as the type of algorithms that can be run onboard, need to be made in order to remain within the constraints of the platforms, such as battery lifetime. As a result, multi-robot collaboration is often considered, aiming to coordinate multiple agents to complete the mission within a limited time or resources budget, for example in search-and-rescue, investigation of spatio-temporal phenomena and inspection of hazardous areas [78].

For multi-robot collaboration to be effective, co-localization of these robots in a common map and coordination of their motion to avoid collisions are imperative. While multi-robot path planning has been receiving considerable attention in the literature with dedicated studies dating back at least three decades, there still is a lack of practical approaches usable in real scenarios as most existing methods rely on overly optimistic assumptions. For example, in reactive path planning [101] the pose of the robot is often assumed to be not subject to uncertainty, while map-based methods such as [22, 143] rely on known global maps and attempt to generate optimal paths through local maps. However, in real missions the map of the environment is most often unknown *a priori* and all state estimates are subject to drift. In [22], the authors tackle this issue by building a globally consistent map of the environment of interest using bundle adjustment. Nonetheless, their approach is limited to a single robot and requires an initial, manually piloted flight to construct the map used for planning and re-localization.

The impact of having unknown or only partially known maps and state estimation drift grows noticeably in multi-robot applications, where the agents should not only steer away from obstacles, but also need to avoid collisions between them. The coordination of multiple robots and the assignment of the flight area in a multi-UAV mission is not trivial, as with an increasing number of agents the state grows quickly. A typical approach from the literature is based on coordination diagrams, which are searched for paths by minimizing a global performance cost function [105, 136]. In [19] different priorities are set for the robots, while more complex approaches [144] explicitly consider the velocity-time space for coordinating robot motions. Other approaches use mixed integer programming models to encode the interactions between robots [54], while others make use of grid search, roadmaps [108] and sampling-based planning [131]. Recent works [78] show promising results in multi-robot coverage planning, but they do not deal with the localization of multiple agents in a common map.

In this work, we take inspiration from existing approaches aforementioned, combining them in a new architecture composed of a central server and a team of robots (i.e. agents). The server collects the experiences of all robots and performs optimization-based sensor fusion to obtain their poses in a common GPS-based reference frame building up a joint, global dense map of the environment. This map is then used to generate global paths in a hierarchical fashion in order to coordinate the agents.

The objective of the proposed pipeline is to provide a practical solution for applications in search-and-rescue scenarios. Here, we assume that prior knowledge about the area of interest is

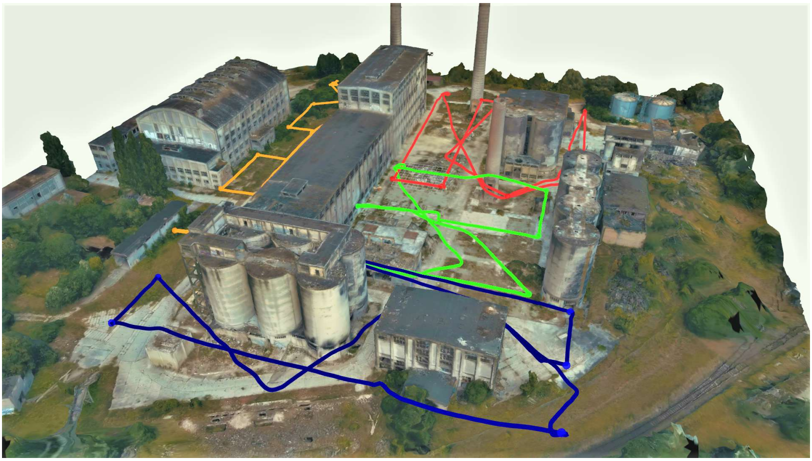


Figure 4.1: 3D-view of the proposed system guiding safe and successful navigation of a team of four UAV-agents (with trajectories shown in different colors) in a photo-realistic simulation of an industrial site, despite that the user-defined waypoints for the UAVs correspond to partially overlapping regions. Via hierarchical planning, the proposed system is able to safely plan the UAVs' paths, running state estimation and local obstacle avoidance onboard each UAV, and sensor fusion and multi-robot global planning on the server to collect all UAVs' experiences in a joint, optimized map.

available (e.g. from satellite images) and that a human operator assigns a list of goal positions to each robot in the team.

In brief, the contributions of this work are the following:

- the design of a centralized agent-server architecture, enabling merging of data from multiple agents (here UAVs), reducing the computational load of each agent,
- the design of a hierarchical planning strategy for multi-agent coordination in partially unknown space,
- an extensive evaluation of the proposed system in photo-realistic simulations with up to four UAV-agents, and
- the source code of the proposed system.

2 Methodology

The proposed system architecture is composed of a central server and a team of robotic agents, as illustrated in Fig. 4.2, with the aim of coordinating this team to explore an area of in-

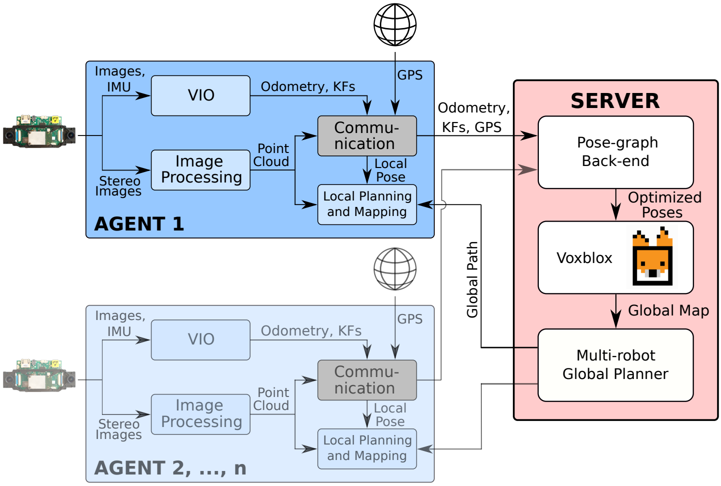


Figure 4.2: The proposed architecture, composed of a set of agents $\{1, \cdots, n\}$ and a server. The agents process the sensor inputs (images, IMU and GPS measurements), perform state estimation in the VIO module and send the information alongside the odometry and the keyframes (KFs) to the central server. The server fuses these measurements in a pose-graph from which globally consistent poses in a common reference can be obtained. These poses, together with the compressed stereo point clouds, are fed to the Voxblox framework [102] to build a dense representation of the explored area, enabling the multi-robot global planner to compute paths which are sent back to the agents for execution.

terest. Each agent is assumed to be capable of estimating its egomotion running keyframe-based Visual-Inertial Odometry (VIO) onboard – in our experiments, an adaptation of VINS-Mono [116] is used. As agents here are small rotorcraft UAVs, in order to follow a global path and avoid smaller obstacles, we employ the local planning approach proposed by [143], while the required local occupancy map is built up using a front-looking stereo camera onboard each agent. To limit the bandwidth requirement of communicating dense scene information to the server, these stereo point clouds are compressed by fusing the ones captured from subsequent frames via voxel filtering. The resulting point clouds are then referenced with respect to an anchor keyframe and are sent together with the keyframe information (including pose, keypoints and the 3D landmarks visible in the keyframe) and the GPS readings corresponding to that keyframe to the server.

The server performs some of the computationally more expensive tasks, such as optimization-based sensor fusion, mapping and global path planning. In particular, the VIO keyframe poses get fused with the GPS information in order to estimate each UAV's trajectory in a global reference frame, estimating the drift of the VIO of each agent. Using the result from the sensor fusion, the compressed point clouds are used to build up a global dense map of the environment

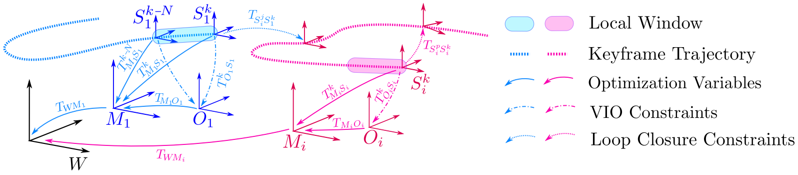


Figure 4.3: Schematic depicting the transformation chains used in the proposed system. The world frame W denotes the GPS-reference frame and is shared amongst all agents, while every agent i has a map with origin M_i and an origin O_i for the corresponding VIO estimate. The body of agent i is indicated with the IMU frame S_i. The drift of the VIO of each agent is encoded in the transformation $\mathbf{T}_{M_iO_i}$. Every time a new keyframe gets inserted in the estimation of an agent, local optimization is triggered operating over the last N keyframes (indicated by the shaded region).

using Voxblox [102]. Based on the resulting dense map, a hierarchical global path-planning strategy is employed to coordinate the mission, such that each agent reaches safely its goals in the previously unmapped area. In the following, the individual parts of the system are described.

2.1 Notation

In this paper, capital letters denote coordinate frames (e.g. A), bold capitals ($\mathbf{A}$) denote matrices and bold small letters ($\mathbf{a}$) vectors. Rigid body transformations from coordinate frame B to A are denoted as $\mathbf{T}_{AB}$, while the translational part of any transformation $\mathbf{T}$ is denoted by $\mathbf{p}$ and the rotational part as $\mathbf{R}$. For notational brevity, we use $\mathbf{T}_{AB} \cdot \mathbf{v}$ to denote the transformation of the vector $\mathbf{v}$ from B to A.

2.2 Pose-graph Back-end

In order to establish the relationships between the different types of measurements as well as amongst multiple agents, we use four different types of coordinate frames as illustrated in Fig. 4.3. The world frame W represents the common GPS reference frame and is unique across all agents. For every agent i we have a map frame M_i, representing the origin of that agent's drift-corrected map, which relates to W by a fixed transformation $\mathbf{T}_{WM_i}$. Finally, the frame O_i denotes the origin of the local VIO estimates describing the pose of the IMU (frame S_i). The drift of VIO is expressed as a time-varying transformation $\mathbf{T}_{M_iO_i}(t)$, where t denotes the time. Since in VIO systems the roll and pitch angles are observable, as proposed in [116], only the position and the yaw angles of each keyframe are optimized. For the sake of readability, in the following we drop the agent index i when referring to transformations within a single agent.

Parameterization and Residuals

In order to optimize the desired transformations, as indicated in Fig. 4.3, we need to be able to express the measurements in terms of these transformations to form residuals. The first type of residual used is a prior, which we define as

$$\mathbf{r}_{AB} = \mathbf{T}_{AB} \boxminus \tilde{\mathbf{T}}_{AB}, \tag{4.1}$$

where $\tilde{\mathbf{T}}_{AB}$ represents the prior knowledge of the transformation $\mathbf{T}_{AB}$. The notation $\boxminus$ indicates a generalized subtractions, which in case of the used 4-DoF parameterization corresponds to

$$\mathbf{T}_1 \boxminus \mathbf{T}_2 := \begin{bmatrix} \mathbf{R}_2^T(\mathbf{p}_1 - \mathbf{p}_2) & \phi(\text{yaw}(\mathbf{T}_1) - \text{yaw}(\mathbf{T}_2)) \end{bmatrix}^T, \tag{4.2}$$

where $\text{yaw}(\mathbf{T})$ represents the yaw angle encoded in the transformation $\mathbf{T}$ and the function ϕ unwraps the yaw angles to lie within the range $[-\pi, \pi)$. The second type of residuals used are relative odometry constraints, representing the error between the measured relative pose of the keyframes j and p (estimated by the VIO) and the predicted transformation based on the state variables:

$$\mathbf{r}_{rel}^{j,p} = (\mathbf{T}_{MS}^{j})^{-1}\mathbf{T}_{MS}^{p} \boxminus (\mathbf{T}_{OS}^{j})^{-1}\mathbf{T}_{OS}^{p} \,. \tag{4.3}$$

Furthermore, we define a residual to relate the current state of $\mathbf{T}_{MO}$ together with the keyframe poses $\mathbf{T}_{MS}^{j}$ to the VIO poses ($\mathbf{T}_{OS}^{j}$) as follows:

$$\mathbf{r}_{o}^{j} = (\mathbf{T}_{MO})^{-1}\mathbf{T}_{MS}^{j} \boxminus \mathbf{T}_{OS}^{j} \,. \tag{4.4}$$

Initial GPS Alignment

In order to bootstrap the estimation of the transformation $\mathbf{T}_{WM}$ between the VIO map of one of the agents and the GPS reference frame, we align the VIO poses to the GPS measurements using least squares. The obtained transformation then gets refined in a non-linear optimization using Gauss-Newton. To decide whether the initialization was successful or not, we compute the covariance of the obtained transformation. As the translational part of $\mathbf{T}_{WS}$ can be estimated even without motion, we consider the system initialized only if

$$\sigma_{\text{yaw}} < \text{threshold} \tag{4.5}$$

holds, while σ_{yaw} is the marginal uncertainty of the estimated yaw angle.

Local Optimization

In the local optimization for agent i the poses of the most recent N keyframes (i.e. $\mathbf{T}_{M_iS_i}$), as well as the transformation between the Map and the GPS-reference frame $\mathbf{T}_{WM_i}$ are refined. Furthermore, the current drift of the VIO (i.e. $\mathbf{T}_{M_iO_i}$) is estimated. The local optimization runs independently for every agent and in our implementation is executed in separate threads. The objective of the local optimization is given by

$$\begin{aligned} \mathcal{X}_i^k = \operatorname{argmin} \sum_{j=k-N}^{k} \Big(\|\mathbf{r}_{o_i}^{j}\|^2_{\Sigma_{o_i}} + \sum_{p \in \mathcal{N}_j, p<j} \|\mathbf{r}_{rel}^{j,p}\|^2_{\Sigma_{rel}} + e_{GPS}^{j} \Big) \\ + \|\mathbf{r}_{WM_i}^{k}\|^2_{\Sigma_{WM_i}} + \|\mathbf{r}_{M_iO_i}\|^2_{\Sigma_{M_iO_i}}, \end{aligned} \tag{4.6}$$

where $\mathcal{X}_i^k$ represents the set of involved transformations:

$$\mathcal{X}_i^k := \begin{bmatrix} \mathbf{T}_{M_i S_i}^{k-N} & \cdots & \mathbf{T}_{M_i S_i}^{k} & \mathbf{T}_{W M_i} & \mathbf{T}_{M_i O_i} \end{bmatrix}. \tag{4.7}$$

The notation $\|\cdot\|_\Sigma^2$ denotes the squared Mahalanobis distance with covariance Σ. The set $\mathcal{N}_j$ denotes all connected neighbours of the keyframe j, which in our case corresponds to a fixed number of temporal neighbours. The terms $\mathbf{r}_{WM_i}^k$ and $\mathbf{r}_{M_i O_i}$ are the prior residuals associated to $\mathbf{T}_{WM_i}$ and $\mathbf{T}_{M_i O_i}$, respectively, while the error term $e_{GPS_i}^j$ summarizes all n_j GPS measurements associated to keyframe j based on the temporal proximity. In order to have a finer temporal resolution for the data association, we utilize the VIO pose of frame f relative to keyframe j to associate the GPS measurements to the keyframe poses:

$$e_{GPS_i}^j \;=\; \sum_{g=1}^{n_j} \|\mathbf{T}_{WM_i} \mathbf{T}_{M_i S_i}^j (\mathbf{T}_{O_i S_i}^j)^{-1} \mathbf{T}_{O_i S_i}^f \mathbf{p}_{S_i U} \;-\; \mathbf{p}_{GPS_i}^g \|_{\Sigma_{GPS_i}}^2, \tag{4.8}$$

where $\mathbf{p}_{S_i U}$ is the offset of the GPS antenna expressed in the IMU frame S_i and $\mathbf{p}_{GPS_i}^g$ corresponds to the GPS measurement g. After the optimization we recover the marginal covariances Σ_{WM_i} and $\Sigma_{M_i O_i}$ corresponding to $\mathbf{T}_{WM_i}$ and $\mathbf{T}_{M_i O_i}$, respectively, and use these as priors in the next optimization step.

Loop-Closure Detection

In order to establish additional constraints within and across the agents' trajectories, we perform visual loop-closure detection in a similar fashion as in [116] using the bag of binary words DBoW2 [49]. Since we want to enable loop detection also across the agents, a single database of words shared amongst all agents is maintained. New loop-closures are detected by first querying the database for visually similar candidates and the best Q candidates are subjected to descriptor-based 2D-2D brute force correspondence search. Any matches get checked for their associated 3D landmarks, which are reprojected from the query frame into the candidate frame and vice-versa, followed by a 3D-2D RANSAC outlier rejection. If sufficient inliers are found, the relative pose, e.g. indicated by $\mathbf{T}_{S_i^j S_1^k}$ and $\mathbf{T}_{S_i^p S_i^k}$ in Fig. 4.3, in case of loop detection between different agents or within the same trajectory, respectively, is optimized by minimizing the reprojection error of the established correspondences.

Global Optimization

Upon detection of a loop-closure, both within one agent's trajectory and across multiple agents' trajectories, we perform a global optimization refining all poses and map transformations that are connected. For example, in a 3-agent team (with labels $1, 2, 3$) let agents 1 and 3 have loop-closures between them. If an additional loops get detected within either 1 or 3's trajectory, all transformations associated with 1 and 3 get optimized, while the transformations within 2

remains independent. The objective of this global optimization step, is given by

$$\mathcal{X} = \operatorname{argmin} \sum_{i \in \mathcal{M}} \Big(\sum_{j \in \mathrm{M}_i} e^{j}_{GPS_i} + \sum_{p \in N_j, p<j} \|\mathbf{r}^{j,p}_{rel_i}\|^2_{\Sigma_{rel_i}} + \sum_{p=k-N}^{k} \|\mathbf{r}^{p}_{o_i}\|^2_{\Sigma_{o_i}} \Big) + \sum_{k_i, p_j \in \mathcal{L}} \|\mathbf{r}^{k_i,p_j}_{lc}\|^2_{\Sigma_{lc}}, \quad (4.9)$$

where $\mathcal{M}$ denotes the set of all previously connected maps, $\mathcal{L}$ denotes the set of loop closure constraints between keyframe k of agent i and keyframe p of agent j. The optimization variables $\mathcal{X}$ are given by

$$\mathcal{X} := \begin{bmatrix} \mathcal{X}_1 & \mathcal{X}_2 & \cdots & \mathcal{X}_{|\mathcal{M}|} \end{bmatrix}, \quad (4.10)$$

where

$$\mathcal{X}_i := \begin{bmatrix} \mathbf{T}_{WM_i} & \mathbf{T}_{M_i O_i} & \mathbf{T}_{M_i S_i^1} & \cdots & \mathbf{T}_{M_i S_i^k} \end{bmatrix} \ \forall i \in \mathcal{M}, \quad (4.11)$$

while the loop closure residual $\mathbf{r}^{k_i,p_j}_{lc}$ is calculated as

$$\mathbf{r}^{k_i,p_j}_{lc} = (\mathbf{T}_{WM_i}\mathbf{T}_{M_i S_i^k})^{-1}\mathbf{T}_{WM_j}\mathbf{T}_{M_j S_j^p} \boxminus \tilde{\mathbf{T}}_{S_i^k S_j^p}, \quad (4.12)$$

where $\tilde{\mathbf{T}}_{S_i^k S_j^p}$ is the relative pose obtained as outlined in Section 2.2.

2.3 Mapping and Multi-robot Global Path Planning

The results of the optimization of the pose-graph in the back-end are used to create a global, dense map of the area in the world reference frame by means of the Voxblox framework [102]. In order to construct the map, we use the compressed point clouds as communicated by the agents together with the optimized pose of the corresponding anchor keyframe.

The updated information about the space is utilized by the multi-robot global path planner in order to coordinate the movements of the UAVs. At the beginning of a mission, every agent is assigned an area to cover as an ordered sequence of GPS waypoints, which have to be reached as quickly as possible. The planner computes the paths for all the agents in the global map and communicates them to the respective agents. Given the current map, any unreachable waypoints (e.g. placed inside an obstacle) are discarded by the planner. Upon completion of the exploration mission, each UAV returns to its respective home position (e.g. where the planning started) by planning only in explored and known space.

To compute the paths, we employ the standard version of RRT* [70] implemented in the OMPL library [135]. The global planner optimizes the path lengths and is optimistic, i.e. it considers unknown space to be free and uses the Truncated Signed Distance Field (TSDF) information available in the Voxblox map. Nonetheless, when a robot has to return to the starting position, the global planner adopts a pessimistic behavior, i.e. planning happens only in known free space. To avoid collisions amongst the robots, we propose a hierarchical approach for multi-robot coordination. At startup, we arbitrarily assign a priority level to each agent, considering that in the team there cannot be two agents with the same priority. The planning happens hierarchically, meaning that the possible flight area of an agent is constrained by both the global occupancy map and the global paths of the agents with higher priority levels. In other words, given the total space V_{TOT}, we remove the occupied parts V_o to obtain the available obstacle-free volume $V_f = V_{TOT} \setminus V_o$. Given n agents, the available space for an agent i is the free space minus the area of flight of all agents with higher priority than i's, i.e.

$$V_i = V_f \setminus \bigcup_{j \in higher\ priority} V_j, \qquad i \in \{0, \cdots, n\} . \tag{4.13}$$

During the planning, we consider a safety bound of dimension d around the paths, while the agents are modeled as spheres with radius r, with $d > r$. The planner starts by computing the trajectory for the agent with highest priority, whose flight space is subjected solely to obstacles. The planner then sequentially proceeds to plan for lower priority levels, until all agents have estimated a valid global path. In order to be reactive to changes in the global map, the global planner checks all the paths for collisions with the updated map at a fixed rate. We consider a path to be invalid if it is in collision with an obstacle or with another robot with higher priority. In case a path for one of the UAVs is invalid, the global planner re-plans the corresponding trajectory and performs collision checking on the paths of the other lower priority UAVs; all the invalid paths will be re-computed.

2.4 Local Path Planning and Obstacle Avoidance

On each agent, we run a local planner adapted from the planning strategy of [143]. The trajectories are represented as quintic Uniform B-Splines allowing to ensure smoothness up to the snap, while to perform obstacle avoidance a local occupancy model of the environment centered at robot position is maintained. The shape of the B-Spline of order k is locally determined by a set of $k+1$ control points $\mathbf{p}_i,\ i \in [0, \cdots k]$. Using a set of equitemporal control-points, the authors of [143] pose the problem of local trajectory re-planning as an optimization problem with the following cost function:

$$E_{total} = E_{ep} + E_c + E_q + E_l \ , \tag{4.14}$$

where E_{ep} is an endpoint cost function penalizing position and velocity deviations at the end of the optimized trajectory segment from the desired ones from the global path; E_c is a collision cost function, E_q the cost of the integral over the squared derivatives (acceleration, jerk, snap) and E_l is a soft limit on the norm of the derivatives over the trajectory. In order to initialize the control points, we utilize the trajectory samples of the global path. However, as the global path is expressed in W, while the reference frame for the controller of agent i is O_i, the global trajectory is transformed using the most recent received estimate of $\mathbf{T}_{WO_i}$ as computed in the sensor fusion back-end. In order to allow for changes in the global paths when a global re-planning action is triggered on the server, we adopt a receding-horizon approach, by planning at a fixed distance along the global path from the current agent's position.

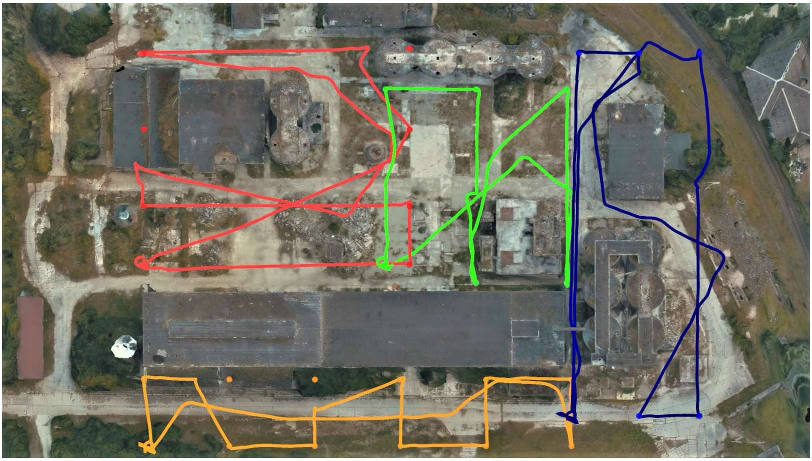


Figure 4.4: Top view of the simulation with four agents. Four different regions were selected and each arbitrarily assigned to one of the agents. As the waypoints were chosen to form a lawnmower pattern, some of them lie inside obstacles (e.g. buildings). The global planner successfully identified and skipped these positions while reaching the accessible remaining points of interest.

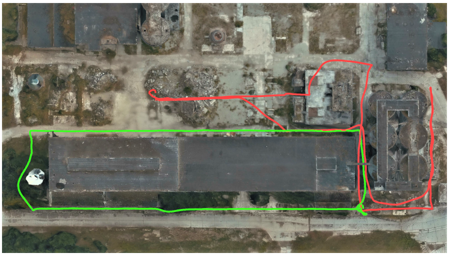
3 Experimental Results

3.1 Benchmarking of Pose-graph Back-end

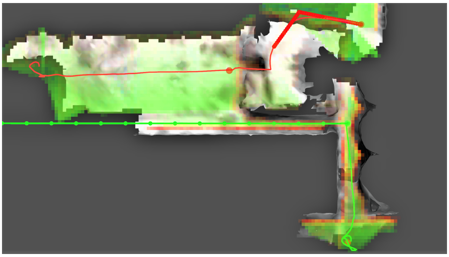
Dataset	VIO only RMSE $[m]$	Optimized RMSE $[m]$	Optimized* RMSE $[m]$
MH01_easy	0.146	0.061	0.073
MH02_easy	0.238	0.082	0.095
MH03_medium	0.210	0.098	0.104
MH04_difficult	0.330	0.092	0.097
MH05_difficult	0.305	0.115	0.118
MH01-MH05	-	0.099	0.101

Table 4.1: ATEs of the keyframe poses averaged over three runs as they enter the proposed system using VIO only, the resulting output of the proposed back-end optimization, and marked with a '*' the ATEs using the estimated $\mathbf{T}_{WM}$.

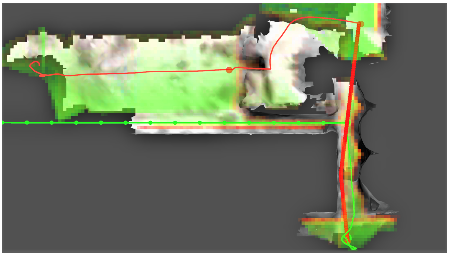
To the best of our knowledge, there is no available dataset containing high-quality visual-



(a) Top-view of the experiment with the paths followed by the agents.



(b) Map before global planning step for Agent 2 (in red).



(c) Planning result for Agent 2.

Figure 4.5: Experiment with two agents assigned to overlapping areas of interest. In (a) a top-view of the experiment is shown, where the paths executed by Agents 1 and 2 are shown in green and red, respectively. (b) shows the situation before the re-planning step of Agent 2. The colored field indicates the traversable space computed by Voxblox, where the color changes from green to red in the direction towards obstacles. (c) shows the re-use of the map created by Agent 1 enabling Agent 2 to plan a shorter path through the same alley.

Message Type	Mean Bandwidth	Std Deviation
Keyframes	24.91 KB/s	16.05 KB/s
Point clouds	68.78 KB/s	35.35 KB/s

Table 4.2: Bandwidth consumption statistics for the communication between one agent and the central server in the experiment described in 3.2. The network traffic necessary to send GPS information to the server and to communicate the paths computed by the global planner to the agents is negligible.

inertial data, GPS data and ground truth, so we evaluate the proposed system on the Machine Hall (MH) sequences of EuRoC dataset [21], simulating GPS measurements by disturbing the available global position measurements with Gaussian noise of a standard deviation of $0.15m$. As these position measurements are expressed in the ground-truth frame, this allows us to evaluate the quality of the estimated map-to-world transformation as well.

The resulting Absolute Trajectory Errors (ATEs) of the keyframe poses are shown in Table 4.1. From the reported values, it can be seen that the proposed sensor fusion approach is successfully able to eliminate the drift in the estimation. In order to evaluate the quality of the estimated transformations between the map(s) and the world frame, the last column in Table 4.1 reports the ATE of the estimated trajectories aligned to the ground-truth using the final estimate of the corresponding transformations $\mathbf{T}_{WM_i}$. As it can be seen, the ATE is only marginally increased when compared to the ATE using least squares trajectory alignment, indicating a high accuracy of the estimated transformations.

3.2 Experiments in Photo-realistic Simulations

The proposed system has been extensively tested in photo-realistic simulations using the Gazebo and RotorS frameworks [48]. The environment used is a reconstruction of a chemical plant in Rüdersdof, Berlin[1]. A 3D-view of the model, spanning an area of $120m \times 120m$, is shown in Fig. 4.1. In order to demonstrate the main capabilities of the proposed system, three different experiments have been carried out:

1. Autonomous navigation along user-defined waypoints with four agents inside the 3D model,
2. Map re-use for two agents with overlapping areas of interest, and
3. Hierarchical planning for a team of three agents operating within the same area.

In the following, the experiments are described in detail. Note that all results can be visualized in the accompanying video at `https://youtu.be/BlFbiuV-d10`

---

[1] `https://sketchfab.com/3d-models`

Map navigation with multiple agents

In this experiment we show the intended use-case of the proposed system, guiding the safe and successful navigation of a team of four UAVs in a previously unknown area given an ordered list of user-defined points of interest (i.e. waypoints), as shown in Fig. 4.4. The waypoints here were selected by dividing up the space amongst the agents and sampling the corresponding area to form a lawnmower pattern. All agents successfully reached all accessible points of interest, while any ill-placed waypoints (e.g. inside or too close to buildings) get discarded by the global planner during navigation. After reaching the last point of interest, all agents navigate successfully back to their starting positions, by planning exclusively in known free space. In Table 4.2 we report the bandwidth usage statistics due to the communication between one agent and the central server over the experiment. Thanks to the compression and the filtering of the point clouds as described in Section 2, the average bandwidth consumption is maintained low.

Map re-use between two agents with overlapping areas of interest

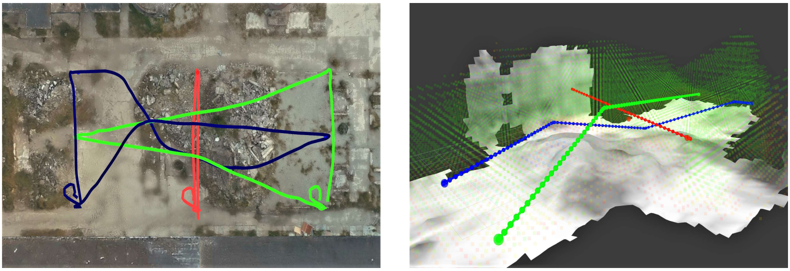
In this experiment, we showcase the advantage of the proposed centralized architecture. As the dense map contains all agents' information, parts mapped by one agent can be re-used for planning by another agent, as shown in this experiments with two UAVs operating in overlapping parts of the model. In Fig. 4.5b, the first agent, shown in green, navigates through a small alley in order to reach its first waypoint. The second agent, shown in red, exploits the information gathered by the first agent, enabling it to take the short path through the same alley instead of navigating around the block, as illustrated in Fig. 4.5c.

Hierarchical planning for three agents within the same area of interest

In the last experiment, we demonstrate the proposed hierarchical approach of the multi-robot global planner. As depicted in Fig. 4.6a, the goal positions for the three agents are placed within the same navigation area. The path followed by the agent with highest priority is shown in red, while the trajectories of the agents with intermediate and lowest priority are represented in green and blue, respectively. Since the agent with highest priority has to reach two waypoints placed in front of it, it plans straight-line trajectories, while the other two agents have to compute intertwined paths. While the UAV in green has to consider only the red UAV, the area of flight of the blue UAV is considerably reduced by the presence of the other two robots. Nonetheless, all agents are able to reach the assigned goals avoiding collisions amongst them and with the obstacles in the map. In Fig. 4.6b we show a 3D-view of the planning step when the agents are about to reach their second waypoints. After the completion of the mission, all the agents return to their respective homing positions.

4 Conclusions

In this paper, we propose a multi-robot estimation and coordination framework for autonomous navigation in partially unknown environments, given a sequence of user-defined points of interest. The system is demonstrated to guide the navigation of a team of UAVs as robotic agents, successfully reaching these waypoints, while ensuring no collisions amongst them or the scene.



(a) Top-view of the experiment with the paths followed by the agents.

(b) 3D-view of the crossing paths, together with the traversable space and the mesh from Voxblox.

Figure 4.6: Experiment with three agents flying in the same area. The sequence of waypoints is selected such that the paths of the agents cross each other as shown in (a). The UAV with highest priority, in red, has to reach two waypoints in front of it, while the other two UAVs, in green and in blue (lowest priority), need to navigate through the whole area in order to reach their goals. The UAVs successfully avoid collisions with each other and the scene as shown in (b).

The UAVs share their experiences with each other via a central server, which creates a globally consistent map of the navigation environment using the proposed multi-agent back-end to bring all estimates in a common reference frame. This map is then used for multi-robot global planning running on the server, coordinating the movement of the team by communicating the computed paths to the agents. These then navigate along the paths, while performing state estimation and local obstacle avoidance onboard. This architectural design allows us to drop some of the typical assumptions commonly made in the multi-robot path-planning literature, such as perfect knowledge of the agents' poses and the map of the environment.

Furthermore, the source code of the system will be made publicly available.

Future directions will investigate the improvement of the system in terms of scalability and the use of automatic assignment of waypoints to agents following a pre-specified strategy (e.g. maximizing coverage) in a bid to boost the autonomy of multi-robot missions.

Paper II

Towards Multi-robot Exploration: A Decentralized Strategy for UAV Forest Exploration

Luca Bartolomei, Lucas Teixeira and Margarita Chli

Abstract

Efficient exploration strategies are vital in tasks such as search-and-rescue missions and disaster surveying. Unmanned Aerial Vehicles (UAVs) have become particularly popular in such applications, promising to cover large areas at high speeds. Moreover, with the increasing maturity of onboard UAV perception, research focus has been shifting toward higher-level reasoning for single- and multi-robot missions. However, autonomous navigation and exploration of previously unknown large spaces still constitutes an open challenge, especially when the environment is cluttered and exhibits large and frequent occlusions due to high obstacle density, as is the case of forests. Moreover, the problem of long-distance wireless communication in such scenes can become a limiting factor, especially when automating the navigation of a UAV swarm. In this spirit, this work proposes an exploration strategy that enables UAVs, both individually and in small swarms, to quickly explore complex scenes in a decentralized fashion. By providing the decision-making capabilities to each UAV to switch between different execution modes, the proposed strategy strikes a great balance between cautious exploration of yet completely unknown regions and more aggressive exploration of smaller areas of unknown space. This results in full coverage of forest areas of variable density, consistently faster than the state of the art. Demonstrating successful deployment with a single UAV as well as a swarm of up to three UAVs, this work sets out the basic principles for multi-robot exploration of cluttered scenes, with up to 65% speed up in the single UAV case and 40% increase in explored area for the same mission time in multi-UAV setups.

Published in:
arXiv.org, 2023
https://arxiv.org/abs/2301.08537

1 Introduction

The growing interest in Unmanned Aerial Vehicles (UAVs) has led to their extensive deployment in tasks such as inspection and search-and-rescue missions. In these applications, the capacity of the robot to quickly explore and map unknown environments autonomously is fundamental. The literature on this topic is extensive, and many different approaches have been proposed throughout the years [28, 77, 121, 126, 165]. However, one of the biggest challenges in the exploration of unknown environments is the capacity to achieve a good trade-off between the competing goals of shorter exploration times of an area of interest (i.e. pushing for high-speed navigation) and safety, which requires caps on the velocity of each robot. In fact, navigating in the vicinity of the boundaries between known and unknown space is challenging, as the robot can get stuck in dead ends, or needs to perform complex dodging maneuvers to avoid collisions. Consequently, to maintain the safety of both the platform and its surroundings, most path planners generate conservative start-and-stop motions, not fully exploiting the capacity of a UAV to fly at high speeds. This effect is exacerbated when the environment to explore is particularly cluttered, as is the case in forests, leading to inefficient and incomplete coverage. By design, these methods generally drive the exploration process by biasing exploration towards large areas of unexplored space. While this strategy could be advantageous in open and wide spaces, it can be detrimental when exploring cluttered scenes. In fact, the main pitfall of such strategies is that, while the exploration process attempts to cover as much unknown space as possible, when this is deployed in environments with many obstacles, thinner trails of unknown space are left unexplored (e.g. due to occlusions), imposing the need for a second sweep of the environment over mostly explored areas.

Aiming to mitigate these issues, pushing for faster coverage of the areas of interest, multi-robot extensions for exploration have also been proposed [9, 32, 119, 140]. However, these focus on the problem of coordination at the system-level and, and while they can perform better from a global planning point of view, they suffer from the same limitations as the single-UAV case in obstacle-dense environments.

Motivated by these challenges, in this work we propose an exploration strategy for autonomous UAV robots aiming to explore forests of increasing tree density, as they pose some of the most difficult challenges for exploration planning. Our objective is to exploit the platform's dynamics to the fullest despite the high density of obstacles, in order to achieve the complete coverage of the environment efficiently. To this end, the proposed strategy enables switching between two different behaviors for each robot; namely, cautious exploration of unknown space and more aggressive maneuvers when navigating in already explored areas to clear smaller portions of unknown space caused by occlusions. We evaluate the proposed approach in a series of challenging experiments in simulation, both in randomly generated forests and in a 3D reconstruction of a real forest (Fig 5.1). Benchmarking against the state of the art reveals superior efficiency for the proposed approach achieving higher overall UAV speeds and lower exploration times. Finally, aiming to set out the scaffolding toward decentralized multi-robot exploration planning, we show how the proposed strategy can accommodate more than one UAV in the exploration mission, and we demonstrate that our method performs comparably to or better than a map-splitting centralized approach.

In summary, the contributions of this work are as follows:

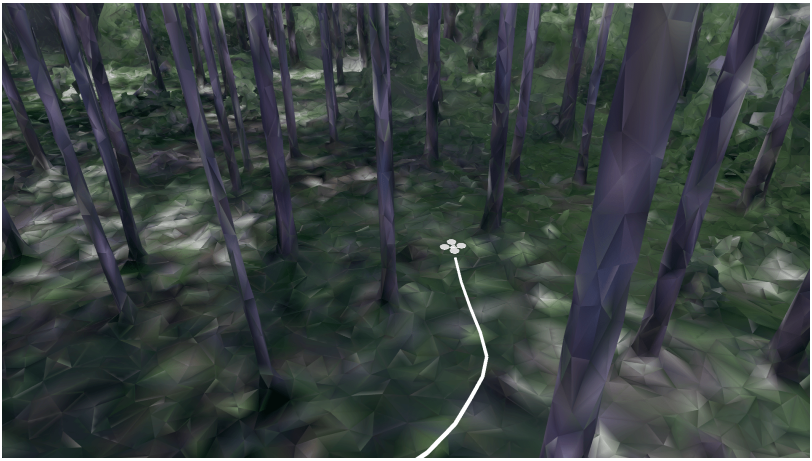


Figure 5.1: 3D-view of the proposed system guiding safe and successful exploration of a UAV in a digital model of a real-world forest [3]. The planner is able to avoid collisions between the UAV and the obstacles, clearing frontiers on-the-go by balancing cautious navigation with aggressive exploitation of known, free space, in a bid to maximize the efficiency of the exploration.

- the design of an exploration strategy, able to strike an effective balance between cautious exploration and aggressive exploitation of the explored map,
- the extension of the single-robot design to a multi-robot decentralized approach, and
- extensive evaluations in simulation, demonstrating better performance than the state of the art.

2 Related Works

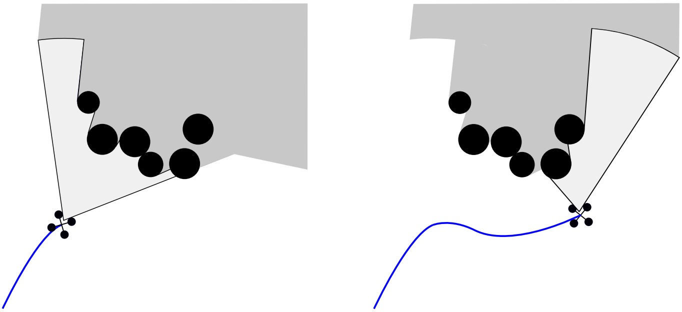
Autonomous exploration of unknown environments with UAVs has been an active field of research over the past few decades. The most popular approach to exploring an area of interest is to use frontiers, defined as the boundary between known and unknown space [155]. These can be utilized to identify potentially informative spatial regions in order to drive the exploration process efficiently until no new frontiers are found and the exploration process can be considered complete. There are different criteria used to decide which frontier to explore next, such as their proximity to the current field of view, following a greedy selection strategy, or having global planning dictate the selection [34]. However, while frontier-based approaches have been proven to yield satisfactory performances, especially in terms of coverage [77, 121], they generally lead to inefficient motions and sub-optimal action selection. This is mostly caused by the sensing modalities used to generate the map of the environment to explore, as the most common sensors, such as RGB-D and stereo cameras, have a limited detection range. Consequently, UAVs need to fly cautiously to ensure safety. Cieslewski *et al.* [28] tackle this limitation, by proposing an exploration strategy that generates velocity commands based on newly detected frontiers, in a bid to maximize the UAV's speed. This method is shown to outperform classical methods [155], but focuses only on local frontiers. Instead, FUEL [165] proposes a hierarchical planner which generates efficient global paths, while encouraging safe and agile local maneuvers for high-speed exploration. FUEL's strategy performs better than [28] and [155] in scenes with low obstacle densities. However, it is more computationally demanding, as it needs to maintain a list of active frontiers, as well as to compute accurate distances between them. This additional bookkeeping becomes prohibitive and impractical in more cluttered and complex environments such as forests. In fact, in this type of scenery, the number of frontiers quickly increases due to occlusions caused by tree trunks, branches, and shrubs.

Another line of research focuses instead on sampling-based path planning to generate viewpoints to explore the space [15, 30], by guiding the robot along possible trails of sampled configurations. The best path is generally found using a greedy approach [15], bringing to complete exploration or accurate surface reconstruction [121] depending on the information gain formulation. Nonetheless, these sampled routes may generate trajectories that deviate from the shortest paths, without taking into consideration the robot's dynamics. Consequently, this causes the UAV to navigate in zigzag patterns, leading to inefficient, slow motions and conservative maneuvers.

To tackle the limitations of frontier- and sampling-based methods, also hybrid approaches have been proposed [26, 126]. Such methods compute global paths towards the most informative frontiers while generating local trajectories using sampling-based planners. However, they do not exploit the full dynamics of the platform and generate sub-optimal routes.

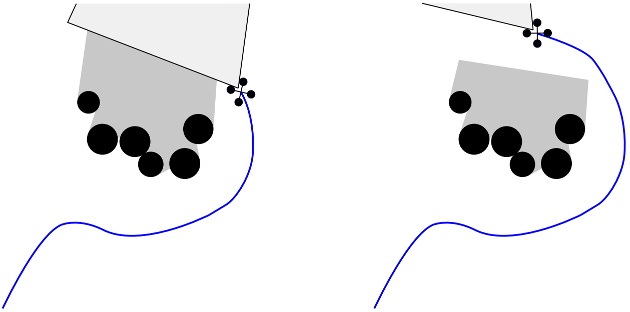
To boost the efficiency in exploration, various multi-robot cooperative frontier-based methods have also been proposed in the literature, both in centralized [84] and decentralized formats [29]. In this spirit, the work in [61] greedily assigns view configurations, while [41] distributes the workload between agents using a Voronoi-based partitioning of the area to explore. Nevertheless, these solutions suffer from the same limitations as in the single-robot case. Instead, the approach in [92] is able to generate efficient trajectories for 3D reconstruction, tackling the multi-robot coordination with a centralized architecture. However, this method requires a prior overhead flight over the area of interest, making it unsuitable for the exploration of forests. The approach proposed in [140] puts more focus on the problem of navigating forests, but the emphasis is more on state estimation rather than on path planning.

Motivated by these limitations, in this work, we propose a strategy that allows a robot to explore complex forest-like environments while flying at high speeds, thanks to the freedom and flexibility that our planner provides to each UAV to switch between different navigation modes online. While slower, cautious exploration is performed using a frontier-based approach, we efficiently clear trails of unexplored space caused by occlusions by employing a more aggressive local exploration strategy, boosting the efficiency of the mission and pushing the overall time to cover a given area of interest down. Moreover, we demonstrate that the proposed pipeline can also be extended to the multi-robot setting in a decentralized fashion.



(a) Time-instant 1

(b) Time-instant 2



(c) Time-instant 3

(d) Time-instant 4

Figure 5.2: A schematic example demonstrating the problem with greedy frontier-based exploration, at progressive time-instants, generating islands of unknown space surrounded by free regions. The field of view of the robot is depicted as a light-gray shaded area delimited by black solid lines, while the obstacles and the unexplored space are in black and dark gray, respectively (a). The robot navigates towards the most informative frontiers (b); however, due to the limited sensor range, the space occluded by the obstacles is not cleared (c). Consequently, since the exploration process is biased towards larger, more informative frontiers, the naïve planner flies the UAV robot ignoring the smaller portion of unexplored space (d).

3 Methodology

The overall problem considered in this work is to explore unknown cluttered environments, such as forests, in the minimum time possible. We assume that the robot is equipped with a front-looking depth camera with a limited sensing range and that the robot's odometry information

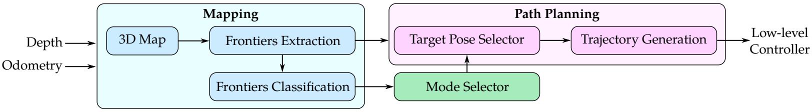


Figure 5.3: Schematic overview of the proposed exploration pipeline for a single agent. The inputs to the system are the robot's odometry and depth information. These are used to generate a 3D grid-based map of the environment, from which frontiers are extracted and clustered. The trails of frontiers are identified and used to select the adequate exploration mode for the agent. Then, the next target pose is chosen and a trajectory towards the goal pose is generated using [163].

is available at a constant rate. However, forest-like scenes are characterized by a high number of obstacles in a variety of dimensions (e.g. trunks, leaves, and branches) that make standard frontier-based exploration approaches inefficient. In fact, during the exploration process, many islands of unknown space are usually left behind, as illustrated in Fig. 5.2, necessitating subsequent passes of exploration on a nearly completely explored map. To tackle this limitation, we propose an exploration pipeline that can change the exploratory behavior of the robot depending on the frontiers in its vicinity. In particular, we propose to define two different modes of operation for the robot, namely the *Explorer* and the *Collector* modes. In the *Explorer* state, the robot is driven by frontiers and it is tasked to explore large unknown areas. Consequently, it predominantly operates on the most external boundaries between known and unknown areas. Conversely, the robot in the *Collector* mode clears small islands of unknown space generated by occlusions, that are left behind during the exploratory phase. The objective of a Collector is to clear these portions of space on the go, avoiding the need for subsequent revisits of the map, at the expense of short local detours. However, notice that these can be performed at high speed, since, when in Collector mode, the robot operates in mostly explored areas. By allowing a robot to switch between these two different modes and by finding the right trade-off between map exploration and exploitation, we can quickly reach full coverage of large cluttered environments. In the following, we first give an overview of the proposed system and our exploration strategies, and then we illustrate how it can be extended to multiple robots.

3.1 System Overview

As shown in Fig. 5.3, the pipeline is composed of three main components: a mapping system, a mode selector, and a path planner.

Given input depth and odometry information, a voxel grid map $\mathcal{M}$ of the environment is generated. At every update, frontiers are extracted from $\mathcal{M}$ and clustered. For each cluster, we adopt the sampling strategy from [165] to generate viewpoints covering the frontiers, and we use them as possible target poses during the exploration process.

Moreover, each cluster undergoes a binary classification step, where unconnected islands of frontiers, or *trails*, are identified. This is necessary to identify those regions that are likely to require an additional revisiting phase towards the end of the mission if a traditional frontier-based exploration method is utilized. Here, a cluster is considered a *trail* if its convex hull is surrounded by free space, or when it has only another neighboring cluster. This implies that most clusters at the corners of the area to be explored are classified as trails. We motivate this design choice by arguing that corners are generally problematic for exploration due to their low informative value. In fact, they are rarely covered in a first sweep of the map, implying the need for a revisiting step. The labeled clusters are then utilized by the Mode Selector to choose the best exploration strategy for the robot, deciding whether it has to persevere in its current mode, or transit to Explorer or Collector. The mode assignment is regulated according to the frontiers in the vicinity of the UAV. Given that our objective is to clear trails locally to avoid large detours on the map, we assign the role of Collector if a minimum number of trails is close to the robot. Instead, we adopt a more exploratory strategy once all smaller islands of unknown space are cleared, or when the trails are far away from the drone. Once a strategy is selected, the viewpoint of the most promising cluster is selected as the new target pose. This is fed to a path planner [163] that generates the trajectory flying the UAV toward its destination. We now describe the different exploration modes in more detail.

3.2 Exploration Strategies

Explorer

Driven by frontiers, the objective of an Explorer is to cover large areas of previously unknown space. Similarly to [28], we process the incoming clusters of frontiers $\mathcal{C}$ from the most recent map update and extract the one with the lowest cost J_E. Notice that these clusters are mostly aligned with the direction of the UAV's motion, implying that, if one of these is selected as the target, the robot avoids abrupt changes in the flight direction or aggressive maneuvers.

The cost associated to the viewpoint $\xi_c := \{\mathbf{x}_c, \gamma_c\}$ covering cluster $c \in \mathcal{C}$ is defined as

$$J_E(\xi_c) := \omega_D J_D(\xi_c) + \omega_V J_V(\xi_c) + \omega_L J_L(c), \tag{5.1}$$

where $\mathbf{x}_c \in \mathbb{R}^3$ is the position of the viewpoint and $\gamma_c \in \mathbb{R}$ its orientation. The cost J_D is the length of the path in $\mathcal{M}$ between the current robot's position and $\mathbf{x}_c$, and it is calculated using the A* algorithm. Instead, J_V is associated with the change in direction of travel, while J_L to the label of cluster c. The terms ω_D, ω_V and ω_L are constant weights.

The cost $J_V(\xi_c)$ is calculated as

$$J_V(\xi_c) := acos(\mathbf{v}_R^T \frac{\mathbf{x}_c - \mathbf{x}_R}{||\mathbf{x}_c - \mathbf{x}_R||_2}), \tag{5.2}$$

where $\mathbf{v}_R \in \mathbb{R}^3$ and $\mathbf{x}_R \in \mathbb{R}^3$ are the robot's current velocity and position, respectively. This cost is directly associated with the angle between the velocity and the direction vector towards the candidate position $\mathbf{x}_c$ covering cluster c. However, it may happen that the cluster is labeled as *trails*, e.g. in the case of occlusions caused by thin obstacles, such as tree trunks. Since an Explorer should focus on actual frontiers, we assign a penalty to these clusters:

$$J_L(c) = \begin{cases} 0 & \text{if } c \text{ is } frontier \\ p_{trail} & \text{if } c \text{ is } trail \end{cases}, \tag{5.3}$$

where p_{trail} is the constant penalty associated with trails.

We then select as target pose the next best viewpoint $\xi_{c^*} := \{\mathbf{x}_{c^*}, \gamma_{c^*}\}$ covering the cluster c^* with the lowest cost:

$$\xi_{c^*} := \arg \min_{\xi_c \ \forall c \in \mathcal{C}} J_E(\xi_c). \tag{5.4}$$

In case the UAV is trapped in a dead-end, or if no new clusters are available in front of the robot, we ignore the cost associated with the robot's velocity and we employ a greedy approach to select the new target pose. We find the best cluster in the vicinity of the robot at a maximum distance d_{max} using the same cost function as in Eq. (5.1), with J_V set to zero:

$$\begin{aligned} \xi_{c^*} := \arg \min_{\xi_c \ \forall c \in \mathcal{C}} \ & \omega_D J_D(\xi_c) + \omega_L J_L(c) \\ & s.t. \ ||\mathbf{x}_c - \mathbf{x}_R||_2 \leq d_{max}. \end{aligned} \tag{5.5}$$

Collector

The objective of a Collector is to clear as many trails as possible, in order to avoid the need for a revisiting step in poorly explored regions of the map at the end of the mission. Since this task implies a detour from the main direction of exploration, the UAV's speed needs to be maximized in order to go back to Explorer mode as soon as possible. To reach this objective, we sort the set of trails $\mathcal{C}_{trails} \subseteq \mathcal{C}$ by associating a cost J_C to each cluster $c \in \mathcal{C}_{trails}$:

$$J_C(\xi_c) := \omega_P J_P(\xi_c) + \omega_A J_A(\xi_c), \tag{5.6}$$

where J_P is associated with the time to reach $\mathbf{x}_c$ and J_A with the time to cover the angular change between the current robot's yaw and the viewpoint's orientation. Instead, ω_P and ω_A are constant weights.

Given the path π_c^R from $\mathbf{x}_R$ to $\mathbf{x}_c$ and the maximum allowed velocity v_{max}, J_P is computed as

$$J_P(\xi_c) := \frac{\text{length}(\pi_c^R)}{v_{max}}. \tag{5.7}$$

Similarly, given the robot current's heading γ_R, the viewpoint's orientation γ_c and the maximum allowed yaw rate $\dot{\gamma}_{max}$, J_A is computed as

$$J_A(\xi_c) := \frac{\angle(\gamma_R, \gamma_c)}{\dot{\gamma}_{max}}, \tag{5.8}$$

where $\angle(\gamma_R, \gamma_c)$ indicates the angular difference between γ_R and γ_c.

The robot then selects the target trail in a step-by-step greedy procedure and behaves as a Collector until all close-by trails are cleared. Since the trails are surrounded by free known space, we double the maximum velocity compared to when in Explorer mode. Consequently, the UAV is able to maximize its velocity, leading to fast motions that allow it to quickly cover all the viewpoints associated with the trails.

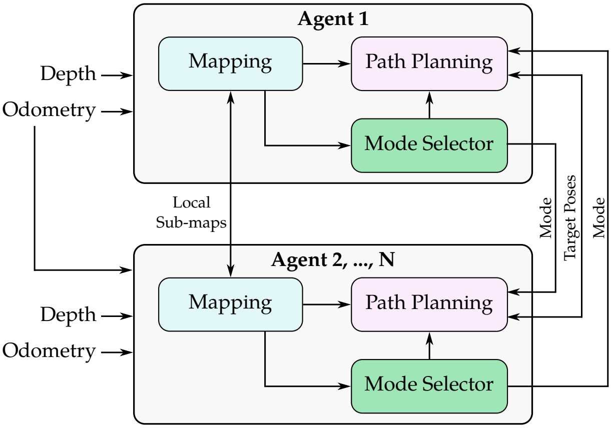


Figure 5.4: Overview of the pipeline in the multi-robot setting, where the UAVs are tasked to collaboratively build a complete map of the area of interest. To fulfill the objective, agents exchange odometry information and local sub-maps, as well as current execution mode and target poses. We assume there exists a maximum communication range between robots. If the distance between two agents exceeds this limit, communication is lost and information is not exchanged anymore, leading to poor coordination.

3.3 Extension to Multi-Robot

System Architecture

The proposed exploration strategy can be easily extended to the multi-robot case. The extended pipeline is shown in Fig. 5.4. Assuming that the agents can localize in a common reference frame, they exchange local sub-maps, as well as odometry information, current target pose, and execution mode. Notice that here we propose a decentralized architecture. Centralized approaches generally assume infinite-range communication between agents and with a ground station. However, standard Wi-Fi communications have a limited range and, when navigating in cluttered environments such as forests, communication lines can be potentially obstructed and signal can be lost. In this work, we propose to use a more flexible point-to-point strategy, assuming that there exists a maximum range of communication between each pair of agents. Our design targets to keep the agents at a valid communication distance. If the distance between agents is higher than the maximum range, communication is lost and information is not exchanged anymore, leading to poor inter-agent coordination and sub-optimal decision-making. We also assume that, when communication is lost and successively regained, agents can synchronize their maps.

Multi-robot Coordination

We encourage coordination between pairs of agents only if they perform compatible actions from an exploration point of view. This implies that, if two UAVs are operating in the same mode, they can collaborate in order to either explore more unknown spaces (Explorers) or clear trails (Collectors). On the contrary, coordination between an Explorer and a Collector should not be encouraged, since their tasks are intrinsically different. In this situation, we propose a *leader-follower* paradigm, where the Explorer (*leader*) explores unknown areas regardless of the position of the second agent, while the Collector (*follower*) follows the leader and clears the trails left unexplored. This design choice allows more flexibility during the execution of a mission, thanks to the possibility to change execution modes online. Notice that, if communication between all agents is lost, their exploration strategy falls back to the single-robot case.

Our collaboration strategy within a robotic team is encouraged as a soft constraint by modifying the cost functions in Eq. (5.1) and Eq. (5.6). If we consider robot i with position $\mathbf{x}_R^i$, given the positions $\mathcal{X}_R^i := \{\mathbf{x}_R^k\}_{k=0}^{N-1}$ of the other $N-1$ robots in the team with $k \neq i$, and their current target positions $\mathcal{G}_R = \{\mathbf{x}_{c*}^k\}_{k=0}^{N-1}$, we modify the cost functions J_E and J_C for robot i as follows:

$$\begin{aligned} J_E(\xi_c, \mathcal{X}_R^i, \mathcal{G}_R) := \omega_D J_D(\xi_c) + \omega_V J_V(\xi_c) + \\ \omega_L J_L(c) + \omega_F J_F(\xi_c, \mathcal{X}_R^i, \mathcal{G}_R) \end{aligned} \tag{5.9}$$

and

$$\begin{aligned} J_C(\xi_c, \mathcal{X}_R^i, \mathcal{G}_R) := \omega_P J_P(\xi_c) + \omega_A J_A(\xi_c) + \\ \omega_F J_F(\xi_c, \mathcal{X}_R^i, \mathcal{G}_R), \end{aligned} \tag{5.10}$$

where ω_F is a constant weight. The cost function $J_F(\xi_c, \mathcal{X}_R^i, \mathcal{G}_R)$ is defined as

$$J_F(\xi_c, \mathcal{X}_R^i, \mathcal{G}_R) := J_F^{att}(\mathcal{X}_R^i) + J_F^{rep}(\xi_c, \mathcal{X}_R^i, \mathcal{G}_R), \tag{5.11}$$

where J_F^{att} aims at keeping the agents i and k in communication range, while J_F^{rep} ensures a minimum distance between them to avoid collisions. Moreover, J_F^{rep} encourages map splitting, by assigning a high cost to candidate target positions close to other agents' current goals.

In more details, the function J_F^{att} for agent i is defined as follows:

$$J_F^{att}(\mathcal{X}_R^i) := \sum_{k=0, k\neq i}^{N-1} \mathcal{I}(i,k) \cdot \frac{1}{2} k_A ||\mathbf{x}_R^i - \mathbf{x}_R^k||_2, \tag{5.12}$$

where k_A is constant factor and $\mathcal{I}(i,k)$ is an indicator function that embeds our coordination strategy:

$$\mathcal{I}(i,k) := \begin{cases} 0 & \text{if } i \text{ Explorer and } k \text{ Collector} \\ 1 & \text{otherwise} \end{cases}. \tag{5.13}$$

This indicates that agent i is attracted toward agent k only if they are in a compatible execution mode. On the contrary, in *leader-follower* mode, i.e. when robot i is an Explorer and robot k is a Collector, the leader ignores the follower, and J_F^{att} goes to zero. Notice that instead, if i is Collector and k an Explorer, $\mathcal{I}(i,k) = 1$ and $J_F^{att} \neq 0$.

Instead, J_F^{rep} is computed as follows:

$$J_F^{rep}(\xi_c, \mathcal{X}_R^i, \mathcal{G}_R) := \sum_{k=0, k\neq i}^{N-1} J_{ik}^{rep}(\mathbf{x}_R^i, \mathbf{x}_R^k) + J_{ik}^{rep}(\mathbf{x}_c^i, \mathbf{x}_{c^*}^k), \tag{5.14}$$

where, given the Euclidean distance $d_{AB} := ||\mathbf{x}_A - \mathbf{x}_B||_2$,

$$J_{AB}^{rep}(\mathbf{x}_A, \mathbf{x}_B) := \begin{cases} k_R(d_c - d_0)^2 \frac{d_c d_0}{d_0 - d_c} & \text{if } d_{AB} \leq d_0 \\ k_R(d_{AB} - d_0)^2 & \text{if } d_c \leq d_{AB} \leq d_0 \\ 0 & \text{otherwise} \end{cases} . \tag{5.15}$$

The parameter k_R is a constant weight, while d_0 represents the minimum distance between positions A and B to have a collision. The parameter d_c represents the distance after which the positions should not approach any closer. This can be selected on the basis of the safety distance required between the UAVs. Notice that J_{ik}^{rep} is not influenced by the roles of agents i and k, as safety and minimum distance requirements need to be always met.

4 Experiments

We evaluate the proposed exploration pipeline in both single- and multi-UAV setups in simulation. In particular, we benchmark our method on a series of realistic, randomly generated forests of increasing tree densities [101], as well as on a 3D reconstruction of a real forest [3].

In the single-agent setup, we compare the proposed method against FUEL [165], while in the experiments with multiple robots we test against a centralized strategy based on map-splitting. In all tests, we use grid map resolutions of $0.10\,\mathrm{m}$ or $0.15\,\mathrm{m}$ depending on the map size, while we set the dynamic limits to $v_{max} = 1.5\,\mathrm{m/s}$ and $\dot{\gamma}_{max} = 0.9\,\mathrm{rad/s}$ for all planners. We simulate a depth camera with a fixed range of $4.5\,\mathrm{m}$ using the Vulkan-based renderer of [10] and the same physical simulator as in [165]. We report the planners' performance in terms of the time needed to complete the exploration of the scene, the total travelled distance, and the average velocity of the UAVs during each experiment.

4.1 Single-robot Experiments

In the single-robot experiments, the models of the synthetic forests are of size $50\,\mathrm{m} \times 50\,\mathrm{m} \times 2\,\mathrm{m}$, while the 3D reconstruction of the REAL FOREST has dimensions $40\,\mathrm{m} \times 40\,\mathrm{m} \times 2\,\mathrm{m}$. The map resolution is set to $0.10\,\mathrm{m}$. As reported in Table 5.1, the proposed planner outperforms FUEL [165] across all scenes in terms of the time taken to reach full coverage (Fig. 5.5), thanks to our adaptive exploration policy that leads to consistently higher UAV velocity throughout each mission, as illustrated in Fig. 5.6. These results demonstrate the benefit of using the proposed adaptive exploration strategy over a fixed-mode method. However, notice that the

	Ours	FUEL [165]
Real Forest		
Completion Time [s]	**500.7 ± 14.8**	757.7 ± 47.9
Travelled Distance $[\mathrm{m}]$	645.0 ± 20.0	**533.2 ± 11.0**
Velocity $[\mathrm{m/s}]$	**1.3 ± 0.5**	0.7 ± 0.4
Sparse Forest (0.05 trees / m^2)		
Completion Time [s]	**665.4 ± 32.7**	1114.1 ± 97.4
Travelled Distance $[\mathrm{m}]$	860.9 ± 34.0	**758.1 ± 57.6**
Velocity $[\mathrm{m/s}]$	**1.3 ± 0.6**	0.7 ± 0.5
Average-Density Forest (0.10 trees / m^2)		
Completion Time [s]	**779.6 ± 110.9**	954.1 ± 28.8
Travelled Distance $[\mathrm{m}]$	910.3 ± 63.7	**713.5 ± 33.1**
Velocity $[\mathrm{m/s}]$	**1.2 ± 0.6**	0.7 ± 0.5
Dense Forest (0.15 trees / m^2)		
Completion Time [s]	**613.2 ± 16.2**	1130.2 ± 28.8
Travelled Distance $[\mathrm{m}]$	**789.7 ± 16.8**	791.0 ± 37.7
Velocity $[\mathrm{m/s}]$	**1.2 ± 0.6**	0.7 ± 0.5
Very Dense Forest (0.20 trees / m^2)		
Completion Time [s]	**658.2 ± 57.2**	904.1 ± 109.5
Travelled Distance $[\mathrm{m}]$	802.0 ± 52.7	**680.6 ± 49.9**
Velocity $[\mathrm{m/s}]$	**1.2 ± 0.6**	0.7 ± 0.5

Table 5.1: Results of the experiments for a single agent. We report the average completion time over 3 runs, as well as the average travelled distance and velocity. The best performance is in bold. The relatively high standard deviation in the timings to complete the missions, in particular in the cases of tree densities of 0.10 and 0.20 trees/m^2, is caused by the complex nature of the map and the large number of occlusions.

proposed design leads to longer travelled distances, albeit guaranteeing that there are no small unexplored areas left. In fact, decision-making both in Explorer and Collector modes is done on a local-map level, and this may cause the UAV to fly longer routes, deviating from the shortest path. Nonetheless, in the proposed strategy we compensate for this shortcoming by encouraging decisions leading to higher UAV velocities, and thus shorter mission times.

4.2 Multi-robot Experiments

In the multi-robot experiments, the proposed planning strategy is tested in a variety of models with fixed, homogeneous obstacle density, as well as in a randomly generated forest with tree density varying across different regions of the map.

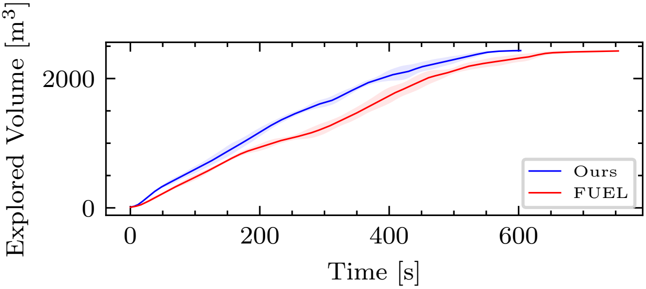


Figure 5.5: The average exploration rate during the experiments in the model REAL FOREST. The shaded region shows the standard deviation. The proposed method reaches complete coverage in less time than FUEL [165].

Maps with a fixed tree density

The results of the multi-robot collaborative exploration strategy of maps with fixed tree densities with two agents are shown in Table 5.2, where a maximum connection distance of 50 m for data exchange between agents is assumed (Fig. 5.7). Here, experiments are performed in forest models of size 100 m × 50 m × 2 m with a map resolution of 0.10 m. The proposed approach is compared against a centralized strategy we devised, employing the FUEL planner [165] and assigning the UAVs to explore maps of equal sizes (i.e. using map-splitting). Note that this strategy assumes homogeneous forest maps, and knowledge of the original map size, which renders this unsuitable for realistic deployment unlike the proposed approach; however, comparisons are presented for the sake of benchmarking.

The proposed strategy reaches comparable results with respect to the centralized approach using FUEL. Similarly to single-robot exploration, the proposed strategy flies the UAVs at higher speeds, incurring longer travel distances. Moreover, our strategy enables automatic load balancing of the exploration mission, yielding similar exploration rates per agent as illustrated in Fig. 5.8. However, in denser forest models, the performance of the proposed approach is seen to degrade, as the UAVs are tasked to fly within a connection range to each other, resulting in limited freedom of movement. This is exacerbated by the increased number of obstacles and occlusions in smaller maps, leading to lower exploration rates. Nevertheless, as aforementioned, the proposed approach is realistically deployable in contrast to the baseline strategy using FUEL.

Map with non-homogeneous tree density

The results of the experiments in a map with non-homogeneous obstacle densities with a team of two agents are shown in Table 5.3. We utilize a model with size 100 m × 200 m × 2 m, with different tree densities (0.2, 0.3 and 0.5 trees/m^2) across distinct map regions. The occupancy grip map resolution is set to 0.15 m, with a maximum inter-agent communication range of

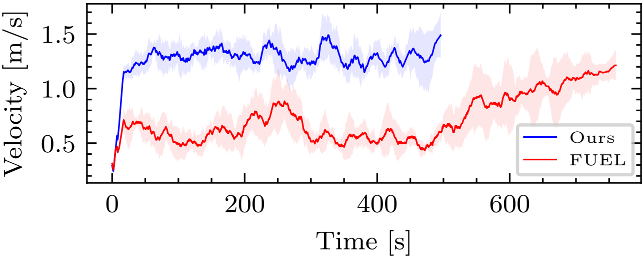


Figure 5.6: The average UAV velocity during the experiments in the REAL FOREST. The shaded region indicates the standard deviation. The proposed strategy is able to fly the UAV at higher velocities than FUEL leading to improved mission efficiency (i.e. time to mission completion). Towards the end of the mission, the UAV following the FUEL strategy also speeds up, as by then the map is mostly explored, and smaller trails are cleared, resembling the Collector mode in the proposed strategy.

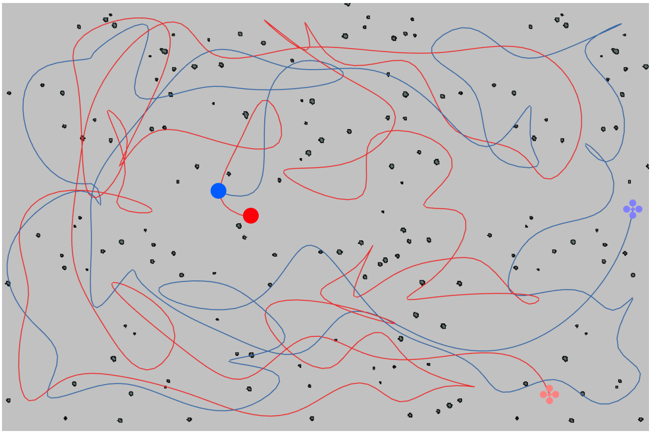


Figure 5.7: Top view of an exploration mission, where a team of two UAVs is tasked to map a randomly generated forest with tree density 0.05 trees/m^2, illustrated with the black dots here. The initial positions of the UAVs are denoted as colored blobs and the final positions as enlarged drone models for clearer visualization. The map is represented as a 2D occupancy grid obtained by slicing the 3D model at $1.5\,\mathrm{m}$ from the ground.

	Ours	Split Map (FUEL [165])
Sparse Forest (0.05 trees / m^2)		
Completion Time [s]	**780.3 ± 32.2**	834.3 ± 64.3
Travelled Distance [m]	958.8 ± 9.2 958.7 ± 60.4	**595.7 ± 51.8**
Velocity [m/s]	**1.2 ± 0.6** **1.3 ± 0.7**	0.7 ± 0.4
Average-Density Forest (0.10 trees / m^2)		
Completion Time [s]	**838.5 ± 64.4**	848.1 ± 98.5
Travelled Distance [m]	915.3 ± 68.4 906.2 ± 58.3	**616.1 ± 40.7**
Velocity [m/s]	**1.2 ± 0.5** **1.1 ± 0.6**	0.8 ± 0.4
Dense Forest (0.15 trees / m^2)		
Completion Time [s]	786.3 ± 40.7	**754.0 ± 23.9**
Travelled Distance [m]	839.0 ± 26.1 882.2 ± 32.0	**586.7 ± 14.2**
Velocity [m/s]	**1.2 ± 0.7** **1.3 ± 0.7**	0.8 ± 0.4
Very Dense Forest (0.20 trees / m^2)		
Completion Time [s]	803.7 ± 52.8	**705.8 ± 73.2**
Travelled Distance [m]	873.8 ± 47.0 912.6 ± 42.7	**580.3 ± 73.3**
Velocity [m/s]	**1.2 ± 0.7** **1.2 ± 0.8**	0.7 ± 0.5

Table 5.2: Results in randomly generated forests with fixed tree densities explored with two UAVs, averaged over 3 runs. For the proposed strategy we report the average travelled distance and velocity per agent. The best performance is highlighted in bold.

$200\,\mathrm{m}$. We perform the same analysis as in Sec. 4.2, assuming a realistic maximum flight time of $1500\,\mathrm{s}$ for the UAVs. As none of the tested planners is able to fully explore the environment within the allowed time, we report the total team coverage at fixed timestamps. The proposed approach consistently outperforms the solution based on map-splitting using FUEL [165] as the planning back-end. The gain in performance is related to the capacity of our strategy to fly the UAVs faster in regions with lower obstacle density and to explore cautiously more cluttered areas while clearing smaller frontier trails on the go. This leads to a more efficient exploration process, able to better exploit the capacity of UAVs to perform highly dynamic flights.

Finally, we report the performance of our method when a team of three robots is deployed. As shown in Table 5.4, a larger team size yields higher total coverage. This demonstrates that this work presents an effective strategy for peer-to-peer sharing of the responsibility of exploration of a large forest area and that the extension to larger teams of multiple UAVs can be realized following this paradigm, pushing for scalable, decentralized multi-robot planning for exploration.

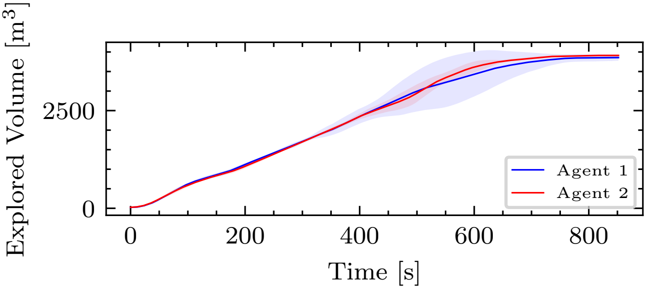


Figure 5.8: The average exploration rate per agent with the proposed approach using two UAVs during the experiments in a random forest with density 0.10 trees/m^2. The shaded region shows the standard deviation. The explored volume is shown to be consistently balanced across the two agents.

5 Conclusion and Future Work

In this work, we propose an exploration pipeline for autonomous UAVs operating in complex, cluttered environments, with a particular focus on forests. We choose this type of environment as one of the inherently most challenging for effective planning due to the increased number of obstacles and occlusions that they exhibit. The proposed strategy allows each UAV to switch between different exploratory behaviors, autonomously balancing cautious exploration of unknown space and more aggressive maneuvers, exploiting already mapped space within a mission. This leads to faster completion times due to higher-speed flights and, consequently, to more efficient and faster map coverage than the state of the art. Moreover, we show how the proposed method can be extended to three, and potentially more robots in a decentralized fashion, demonstrating automatic and effective load balancing across the participating agents. Following the push for automating higher-level decision-making in robotic missions, this work constitutes a key milestone towards effective exploration planning for robotic teams.

The natural next step for this work is to address the integration and deployment of the proposed pipeline onboard real platforms, while further investigations will push advancing coordination strategies in larger multi-robot teams.

	Ours	Split Map (FUEL [165])
Travelled Distance [m]	1481.9 ± 467.0 1487.0 ± 440.8	**735.2 ± 500.1** **697.1 ± 365.6**
Velocity [$\mathrm{m/s}$]	**1.3 ± 0.6** **1.2 ± 0.7**	0.7 ± 0.4 0.6 ± 0.5
Explored Volume [m^3] - 300 s	**542.3 ± 23.3** 565.9 ± 31.4	514.6 ± 313.0 **632.0 ± 184.8**
Explored Volume [m^3] - 600 s	**1062.8 ± 81.3** **1011.6 ± 65.3**	840.0 ± 580.7 728.3 ± 163.5
Explored Volume [m^3] - 900 s	**1400.0 ± 265.3** **1302.6 ± 110.5**	1145.3 ± 814.7 798.8 ± 146.2
Explored Volume [m^3] - 1200 s	**1626.7 ± 431.2** **1466.5 ± 185.4**	1330.1 ± 905.3 910.4 ± 83.9
Explored Volume [m^3] - 1500 s	**1816.0 ± 550.8** **1637.0 ± 299.1**	1437.8 ± 971.8 1040.5 ± 124.3

Table 5.3: Results in a randomly generated forest with non-homogeneous tree densities when explored with two UAVs, averaged over 3 runs. We report the average travelled distance and velocity per agent at 1500 s, as well as the total explored volume by the team at different timestamps. The best performance is highlighted in bold.

Timestamp	Two Agents	Three Agents
300 s	1108.2 ± 54.7 m^3	**1208.0 ± 91.6 m^3**
600 s	2074.3 ± 144.8 m^3	**2098.4 ± 143.8 m^3**
900 s	2702.6 ± 375.9 m^3	**2748.3 ± 111.9 m^3**
1200 s	3093.2 ± 616.6 m^3	**3223.5 ± 127.9 m^3**
1500 s	3453.0 ± 849.9 m^3	**3621.8 ± 321.0 m^3**

Table 5.4: Results in a randomly generated forest with varying tree densities across different regions of the model, averaged over 3 runs. Here, the map is explored with teams composed of two and three UAVs. We report the total volume covered by the team at different timestamps. The best performance is highlighted in bold.

Paper III

Perception-aware Path Planning for UAVs using Semantic Segmentation

Luca Bartolomei, Lucas Teixeira and Margarita Chli

Abstract

In this work, we present a perception-aware path-planning pipeline for Unmanned Aerial Vehicles (UAVs) for navigation in challenging environments. The objective is to reach a given destination safely and accurately by relying on monocular camera-based state estimators, such as Keyframe-based Visual-Inertial Odometry (VIO) systems. Motivated by the recent advances in semantic segmentation using deep learning, our path-planning architecture takes into consideration the semantic classes of parts of the scene that are perceptually more informative than others. This work proposes a planning strategy capable of avoiding both texture-less regions and problematic areas, such as lakes and oceans, that may cause large drift or failures in the robot's pose estimation, by using the semantic information to compute the next best action with respect to perception quality. We design a hierarchical planner, composed of an A* path-search step followed by B-Spline trajectory optimization. While the A* steers the UAV towards informative areas, the optimizer keeps the most promising landmarks in the camera's field of view. We extensively evaluate our approach in a set of photo-realistic simulations, showing a remarkable improvement with respect to the state-of-the-art in active perception.

Published in:
IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2020
DOI: 10.1109/IROS45743.2020.9341347

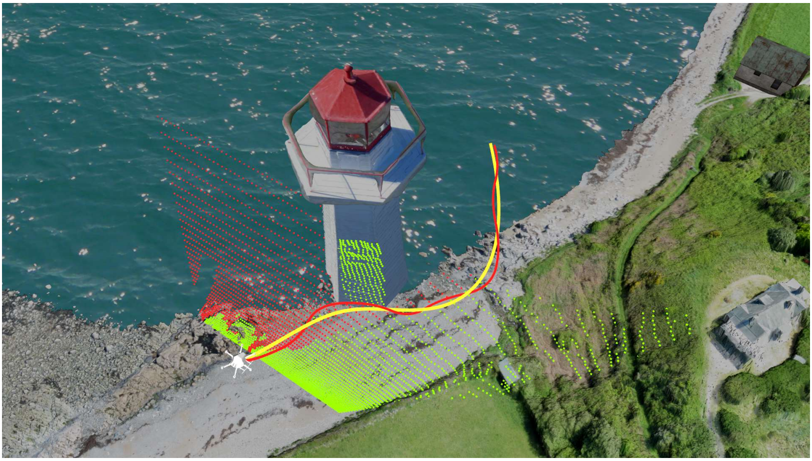


Figure 6.1: 3D-view of a planning iteration of the proposed planner in the *Bay* experiment. The informative areas are highlighted with green dots, while the perceptually degraded parts are in red. The planner is able to avoid the collision between the UAV and the obstacle, while steering the robot towards the informative areas. The red line is the trajectory computed by the path-searching algorithm, while the yellow line is the trajectory after optimization.

1 Introduction

Safe navigation in unknown and dynamic environments is a fundamental skill for truly autonomous mobile robots. For successful autonomous navigation, a robot must map the environment with sufficient accuracy to avoid collisions with obstacles and at the same time localize itself in this estimated map. Today, both real-time localization and mapping are still open problems. While GPS is widely used, there are numerous situations that it can fail, such as, around high buildings and mountains, due to bad weather, and jamming. In these cases, ground robots can just stop, but aerial robots need a safe method to reach an emergency landing spot or even better continue to fly towards their destination. Visual-based navigation is an alternative to GPS, usually using Visual-Inertial Simultaneous Localization and Mapping (VI-SLAM) algorithms. In addition, VI-SLAM is also capable of computing the local map to avoid collision with unknown or dynamic objects [138]. However, it also has its limitation; for example, these algorithms do not perform well in texture-less areas. Recently, the emergence of perception-aware path-planning algorithms showed it is possible to avoid paths through texture-less regions, such as [33, 162].

This work proposes an active perception path-planning method that avoids not only texture-less regions, but also is capable of avoiding areas with sufficient texture, albeit unsuitable for robust localization (e.g. due to dynamic objects and specularities). Our approach aims to leverage recent advances in real-time semantic segmentation using deep learning to identify the problematic areas. Today, there is a plethora of algorithms that are capable of detecting well-known problematic areas to fly. Lakes and oceans can be easily detected by algorithms such as [134], as well as roads and people can be recognized by [117]. In brief, the contributions of this work are the following:

- the design of a perception-aware navigation architecture for UAVs in challenging environments,
- the design of a hierarchical path-planning pipeline able to incorporate information from semantic segmentation with real-time re-planning capabilities, and
- an extensive quantitative evaluation of the performances of the proposed system with respect to the state-of-the-art in active perception in photo-realistic simulations.

2 Related Works

The link between active perception and the problem of SLAM, providing estimates about the robot's pose and its workspace, is strong [18, 95]. Both the motion and the path followed by a robot have a great impact on the performance of state estimation algorithms, so the aim in active perception is to fill the gap between path-planning and state estimation by considering the robot's state uncertainty during planning. This constitutes a fundamental step towards the creation of a fully autonomous system, able to cope with the uncertainties present in a real mission.

Active perception has roots in the seminal work of [8], advocating that sensory performance can be improved by proper selection of control actions. Other early works describe the importance of using sensors actively (or *active vision*) for problems, such as structure from motion [4]. Since then, efforts have been made to integrate perception, path-planning and control in a unified framework. One of the first successful approaches is the work in [36], attempting to control two movable cameras on a ground robot to reduce the uncertainty during localization estimation. Based on their earlier results for incremental building and maintaining of maps for a navigating robot, [36] relies on the cross-coupling between the pose of the robot and the mapped features. In the same spirit, [7] and [2] provide more recent, successful examples that integrate path-planning with control and state estimation. In contrast to [36], these works do not use sensors actively and instead focus on motion generation for enhancing the state estimation process.

In general, the problem of active perception can be formulated as Partially Observable Markov Decision Processes (POMDPs) [66], providing a framework for planning under uncertainty. However, the complexity of solving high-dimensional POMDP models motivates more efficient solutions. Belief-space planning emerged to allow the integration of the expected robot belief into its motion planning efficiently [65] and led to powerful tools, such as belief roadmaps [114] and belief trees [18], which were used to approach the problem of exploration and mapping [129]. In recent years, various approaches based on receding-horizon planning strategies emerged, combining efficient exploration of unknown environments with belief-space based planning in order to enhance the on-the-go mapping behaviour of the robot [104, 162]. One

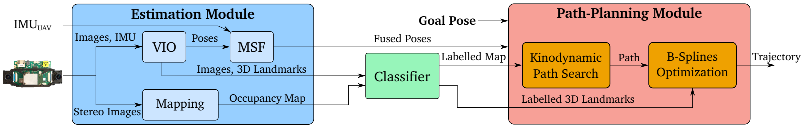


Figure 6.2: Schematic representation of the pipeline. In the Estimation Module, we process the sensor inputs (images and IMU readings) to estimate the pose of the UAV and the positions of the 3D landmarks. These landmarks, together with the images and the occupancy map obtained from the stereo camera, go through a classification step, where each point is evaluated on its utility for camera-based state estimation. The labelled data and the poses are used the Path-Planning Module to generate the next best trajectory while considering both the dynamics of the platform and the perception quality.

of the main objectives is to create a computationally tractable planning framework that can generate optimized navigation paths online, running alongside all other essential computation onboard a small robot [1].

Currently, the most relevant competitor to this work is Zhang and Scaramuzza [162], proposing a perception-aware receding-horizon approach that generates a collection of possible trajectories and evaluates them in terms of landmark concentration, collision probability, and distance to the goal. In this work, we show that landmark concentration alone is not enough for selecting the best areas to fly trough, because unstable landmarks can introduce significant errors in visual-based localization systems. Instead, we employ semantics to evaluate the quality of both the candidate areas for navigation and the landmarks. We propose a path-planning pipeline building on top of [163] able to incorporate this additional information in order to encourage the navigation through informative regions of the space and to favor the triangulation of high-quality landmarks. Our approach shows noticeable improvements in state estimation performance when compared to a purely reactive planning strategy and the perception-aware planner proposed by [162]. Our pipeline is described in the next sections.

3 Problem Description

The overall problem considered in this work is to reach a predefined goal pose while minimizing the drift in a visual-based pose estimation algorithm. The core assumption of this work is that there exist areas which are more suitable for localization than others, and our objective is to fly through them as much as possible. We formulate this as a path-planning problem, where the perception quality extracted from semantic labels plays a primary role. We aim to generate smooth and collision-free trajectories in a receding horizon fashion while considering the constraints of Keyframe-based Visual-Inertial Odometry (VIO) systems. In this work, semantic classes (e.g. water and ground) are manually mapped to a certain level of perceptual informativeness. To incorporate the perception quality, we employ a path-searching algorithm to encourage navigation in well-textured areas, followed by a trajectory optimization step where the objective is to keep the best 3D landmarks estimated by the VIO pipeline in the field of view of the camera.

4 System Overview

As shown in Fig. 6.2, the proposed pipeline consists of two main components; a monocular camera-based state estimation module and a path-planning architecture.
The platform is equipped with a front-looking stereo camera with hardware-synchronized IMU [100]. A dense 3D reconstruction of the surroundings of the robot is obtained from the pair of stereo images, while only one camera is used to estimate the robot's state. The dense point cloud is stored as an occupancy map by means of a 3D circular buffer [143]. We employ the keyframe-based VIO system VINS-Mono [116] to estimate the 6-Degrees-of-Freedom (DoF) pose of the camera using the stream of grayscale images and IMU measurements. This estimate is further fused with the readings of an additional IMU mounted on the center of the UAV using the Multi-Sensor Fusion (MSF) framework [82] in order to get better velocity estimates for control purposes. In addition to the poses, the VIO module gives the estimated positions of the 3D landmarks used for localization.

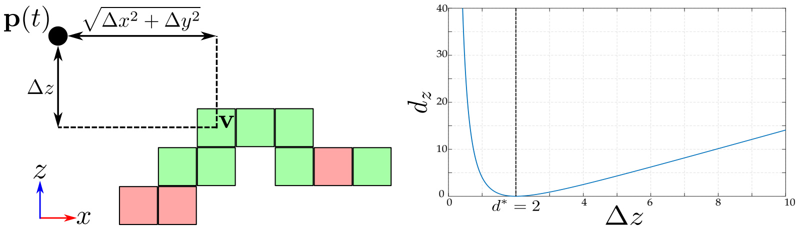
Before being sent to the trajectory generation module, both the dense 3D occupancy map and the sparse landmarks go through a classification step, which provides binary labels for all the points. We consider a point to be of high quality if it belongs to parts of the scene useful for camera-based state estimation. This is motivated by the fact that some parts of the environment, such as water or moving objects, can produce erroneous state estimates because VIO systems assume non-specular surfaces and static scenes. Relying on such features may have disastrous consequences, e.g. increasing drift in the estimation or failures. Given the detection problem of moving objects, such as cars and people, as well as ground classification, is not the core of our contribution here, we used ground-truth semantics in our experiments, but there are several off-the-shelf algorithms available [117, 134] that we could use to this end. The labelled occupancy map and landmarks are used by the path-planning architecture to reason about the next best action. The path-planning module is based on a hierarchical structure, composed of a path-search step followed by trajectory optimization. The path-search planner uses the labelled occupancy map to steer the robot towards potentially informative areas using the A^* algorithm. The output path might be sub-optimal, so it is later refined by a B-Spline-based planner. We formulate an optimization problem in an attempt to follow the trail of high-quality landmarks by keeping them in the field of view of the camera during navigation. In the next section, the proposed path-planning architecture is explained in details.

5 Path Planning

The first step of the planning pipeline consists of a kinodynamic A^* path search [39]. Its output is used as an initial guess by a second planner, which parametrizes the trajectory as continuous B-Splines and performs trajectory optimization. In the following, we indicate with $\mathcal{M}$ the occupancy map of the environment.

5.1 Kinodynamic Path Search

The proposed path-searching module builds on top of the kinodynamic A^* algorithm presented in [163]. Similarly to the original implementation, we use motion primitives instead of straight



(a) Side view of the occupancy map. The green voxels belong to $\mathcal{M}_G$, while the red ones correspond to low-quality areas. The distances used in the computation of d_M for a voxel $\mathbf{v}$ are highlighted.

(b) Plot of the cost d_z as a function of the height difference Δz and the threshold d^*.

Figure 6.3: Representation of the terms necessary for the computation of d_M in the path-search problem.

lines as graph edges in order to respect the multicopter's dynamics. As the first step is the iterative expansion of primitives in $\mathcal{M}$, the trajectories ending in the same voxels are pruned and only the one with the smallest cost is maintained. After expansion and pruning, the remaining primitives are checked for safety and dynamic feasibility. Finally, in order to improve the chances of reaching the goal, we adopt the analytic expansion strategy detailed in [163].
The path search is limited to the position of the robot in $\mathbb{R}^3$, and thanks to the differential flatness of multirotor systems [88], the trajectory is represented as three, independent, time-parametrized polynomial functions $\mathbf{p}(t)$:

$$\mathbf{p}(t) := [p_x(t), p_y(t), p_z(t)]^T, \; p_d(t) = \sum_{k=0}^{K} a_k t^k \tag{6.1}$$

with $d \in \{x, y, z\}$. Notice that the orientation is not considered, since at this point the objective is to steer the robot towards informative areas rather than enforcing rotations towards them. Thus, the orientation $\theta(t)$ along the path is computed as:

$$\theta(t) := \arctan \frac{\dot{p}_y(t)}{\dot{p}_x(t)}. \tag{6.2}$$

We consider the multirotor system as linear and time-invariant, with state $\mathbf{s}(t) := [\mathbf{p}(t)^T, \dot{\mathbf{p}}(t)^T, \ldots, \mathbf{p}^{(n-1)}(t)^T]^T \in \chi \subset \mathbb{R}^{3n}$ and control input $\mathbf{u}(t) := \mathbf{p}^{(n)}(t) \in \mathcal{U} := [-u_{max}, u_{max}]^3 \subset \mathbb{R}^3$. We set $n = 2$, corresponding to a double integrator.
As the aim is to encourage the navigation in well-textured areas, the cost of a trajectory is defined as

$$\mathcal{J}(T) = \int_0^T \Big(w_u \|\mathbf{u}(t)\|^2 + w_M d_M(\mathbf{p}(t), \mathcal{M}) \Big) dt + w_T T, \tag{6.3}$$

where $\|\mathbf{u}(t)\|^2$ is the control cost; $d_M(\mathbf{p}(t), \mathcal{M})$ represents a penalty for navigating far away from informative areas; and T is the total time of the trajectory. The terms w_u, w_M and w_T are the constant weights associated to each cost. While the meaning of the first and last terms is clear, we now describe the term $d_M(\mathbf{p}(t), \mathcal{M})$ in more detail.

Given the latest labelled occupancy map $\mathcal{M}$, we treat the informative voxels as attractors. Notice that the remaining voxels do not act as repellers, because $d_M(\mathbf{p}(t), \mathcal{M})$ is a soft constraint on the position of the robot. This is motivated by extreme situations, where most or all of the area is texture-less. In this case, the robot should not interrupt its navigation towards the goal, even if it has to navigate these poorly textured regions. In such situations, the path-search algorithm tries to minimize the control cost and total time required to reach the goal.

If we define $\mathcal{M}_G \subseteq \mathcal{M}$ as the set of voxels corresponding to high-quality areas, $d_M(\mathbf{p}(t), \mathcal{M})$ is computed as

$$\begin{aligned} d_M(\mathbf{p}(t), \mathcal{M}) := \sum_{\mathbf{v} \in \mathcal{M}_G \subseteq \mathcal{M}} d_v(\mathbf{p}(t), \mathbf{v}) = \\ \sum_{\mathbf{v} \in \mathcal{M}_G \subseteq \mathcal{M}} d_{xy}(\mathbf{p}(t), \mathbf{v}) + d_z(\mathbf{p}(t), \mathbf{v}), \end{aligned} \tag{6.4}$$

where each voxel $\mathbf{v} \in \mathcal{M}_G$ has coordinates $\mathbf{v} = [v_x, v_y, v_z]^T$. The cost $d_M(\mathbf{p}(t), \mathcal{M})$ is composed of two potential functions (Fig. 6.3). While $d_{xy}(\mathbf{p}(t), \mathbf{v})$ encourages navigation on top of texture-full areas, $d_z(\mathbf{p}(t), \mathbf{v})$ is a soft constraint on the minimum distance from the ground, as shown in Fig. 6.3a. The two costs are calculated as

$$d_{xy}(\mathbf{p}(t), \mathbf{v}) := (p_x(t) - v_x)^2 + (p_y(t) - v_y)^2, \tag{6.5}$$

and by defining $\Delta z := |p_z(t) - v_z|$,

$$d_z(\mathbf{p}(t), \mathbf{v}) := d^* \Delta z + \frac{1}{2} \frac{{d^*}^4}{\Delta z^2} - \frac{3}{2} {d^*}^2, \tag{6.6}$$

where d^* controls the minimum height of the robot with respect to the voxels in $\mathcal{M}_G$. The cost d_z is plotted as a function of Δz in Fig. 6.3b.

From these definitions, the cost of a motion primitive with discrete inputs $\mathbf{u}(t) = \mathbf{u}_k$ and time duration τ is $e_c = (w_u \|\mathbf{u}_k\|^2 + w_M d_{M,k} + w_T)\tau$. The final cost of the path to reach a state $\mathbf{s}_c$ from the start state $\mathbf{s}_s$ consisting of N primitives is

$$g_c = \sum_{i=0}^{N} (w_u \|\mathbf{u}_{k_i}\|^2 + w_M d_{M,k_i} + w_T)\tau. \tag{6.7}$$

In order to speed up the search in the A^* algorithm, the choice of an admissible and consistent heuristic is fundamental. We propose to use the heuristic function $h_c := h_{u,T} + h_M$, where $h_{u,T}$ is obtained by minimizing the relaxed cost function from $\mathbf{s}_c$ to the goal state $\mathbf{s}_G$

$$\mathcal{J}(T) = \int_0^T w_u \|\mathbf{u}(t)\|^2 dt + w_T T \tag{6.8}$$

and $h_M := d_M(\mathbf{p}_G, \mathcal{M})$, where $\mathbf{p}_G$ corresponds to the position in the goal state $\mathbf{s}_G$. Finally, the total cost to reach the goal state is defined as $f_c = g_c + h_c$. Regarding the computation of $h_{u,T}$, the interested reader can refer to [163].

5.2 Trajectory Optimization

While the trajectory computed in the path-searching step encourages navigation towards the informative areas, the information produced by the VIO module, i.e. the 3D landmarks, is not utilized. Therefore, a trajectory optimization step leveraging this additional level of information is necessary. We propose to parametrize the trajectory $\boldsymbol{\pi}(t)$ as an uniform B-Spline and formulate an optimization problem where the aim is to generate smooth and feasible trajectories encouraging the tracking and triangulation of high-quality 3D landmarks.

Uniform B-Splines

A B-Spline $\boldsymbol{\pi}(t)$ of degree K can be evaluated as follows:

$$\boldsymbol{\pi}(t) = \sum_{i=0}^{N} \mathbf{q}_i B_{i,K-1}(t), \tag{6.9}$$

where $\mathbf{q}_i$ are the control points at time t_i with $i \in \{0, \dots, N\}$, and $B_{i,K-1}(t)$ are the basis functions.

In our case, each control point in $\{\mathbf{q}_0, \mathbf{q}_1, \dots, \mathbf{q}_N\}$ encodes both the position and orientation of the robot, i.e. $\mathbf{q}_i := [x_i, y_i, z_i, \theta_i]^T \in \mathbb{R}^4$ with $\theta_i \in [-\pi, \pi)$. We adopt a uniform representation, meaning the elements of the knot vector $[t_0, t_1, \dots, t_{N+1+K}]$ and $t_m \in \mathbb{R}$ have a fixed knot span $\Delta t = t_{m+1} - t_m$. For sake of readability, we drop the explicit dependency on time t in the following.

Problem Formulation

The planning problem is formulated as an optimization with the following cost function:

$$\mathcal{F}_{TOT} = \lambda_s \mathcal{F}_s + \lambda_f \mathcal{F}_f + \lambda_c \mathcal{F}_c + \lambda_l \mathcal{F}_l + \lambda_v \mathcal{F}_v, \tag{6.10}$$

where $\mathcal{F}_s$ is the smoothness cost; $\mathcal{F}_c$ is the collision cost; $\mathcal{F}_f$ is a soft limit on the derivatives (velocity and acceleration) over the trajectory; $\mathcal{F}_l$ is the penalty associated to loosing track of the high-quality landmarks currently in the field of view; and $\mathcal{F}_v$ is a soft constraint on the co-visibility between control points of the spline. The coefficients $\lambda_s, \lambda_c, \lambda_f, \lambda_l$ and λ_v are the constant weights associated to each cost.

For a B-Spline of degree K defined by $N + 1$ control points $\{\mathbf{q}_0, \mathbf{q}_1, \dots, \mathbf{q}_N\}$, our optimization acts on $\{\mathbf{q}_K, \mathbf{q}_{K+1}, \dots, \mathbf{q}_{N-K}\}$ while keeping the first and last K control points fixed due to boundary constraints. Thus, we do not have to add a cost associated to the end point of the trajectory. Furthermore, in contrast to other recent approaches, such as [143], in the formulation of the smoothness and feasibility costs, we explicitly remove the dependency on time and we perform time re-allocation after optimization; we notice that this approach does not degrade the quality of the trajectory.

We formulate the cost functions as elastic bands [167] by writing fictitious forces $\mathbf{F}_{i,j} := \mathbf{q}_i - \mathbf{q}_j$ acting from the control points $\mathbf{q}_j$ to $\mathbf{q}_i$. From this, we formulate the smoothness cost as

$$\begin{aligned}\mathcal{F}_s = \sum_{i=K-1}^{N-K+1} \|\mathbf{F}_{i+1,i} + \mathbf{F}_{i-1,i}\|^2 = \\ \sum_{i=K-1}^{N-K+1} \|(\mathbf{q}_{i+1} - \mathbf{q}_i) + (\mathbf{q}_{i-1} - \mathbf{q}_i)\|^2.\end{aligned} \tag{6.11}$$

The purpose of $\mathcal{F}_s$ is to avoid discontinuities in the position and in the velocity and acceleration profiles. Similarly, given the maximum velocity and acceleration v_{max} and a_{max}, the feasibility cost $\mathcal{F}_f$ penalizes the segments of the trajectory exceeding these limits. The penalty for a single dimension velocity v_d with $d \in \{x, y, z\}$ is:

$$F_v(v_d) = \begin{cases} (v_d^2 - v_{max}^2)^2 & \text{if } v_d^2 \geq v_{max}^2 \\ 0 & \text{otherwise} \end{cases}. \tag{6.12}$$

The cost for the acceleration $F_a(a_d)$ is identical. The final feasibility cost is then given by the contribution of both the velocity and acceleration limits:

$$\mathcal{F}_f = \sum_{d\in\{x,y,z\}} \sum_{i=K-1}^{N-K+1} F_v(v_{i,d}) + F_a(a_{i,d}). \tag{6.13}$$

Differently from $\mathcal{F}_s$ and $\mathcal{F}_f$, the collision cost acts as a repulsive force, penalizing the trajectory points that are within a threshold distance d_o to the closest obstacles:

$$\mathcal{F}_c = \sum_{i=K}^{N-K} c(\mathbf{q}_i) \tag{6.14}$$

with

$$c(\mathbf{q}_i) = \begin{cases} (d(\mathbf{q}_i) - d_o)^2 & \text{if } d(\mathbf{q}_i) \leq d_o \\ 0 & \text{otherwise} \end{cases}, \tag{6.15}$$

where $d(\mathbf{q}_i)$ is the distance between $\mathbf{q}_i$ and the closest obstacle.

Instead, the cost $\mathcal{F}_l$ encourages the triangulation of high-quality landmarks and assigns a high penalty to the control points with few or none such landmarks in the field of view. The choice of using the number of tracked landmarks instead of other information gain formulations, is dictated by the fact that VIO systems benefit more from good spatial distributions of landmarks than sparse informative measurements. The camera frustum is characterized by its horizontal and vertical angular apertures, indicated as θ_{hor} and θ_{ver}, respectively. We consider a landmark to be visible if it lies within the field of view, which is identified by five half-spaces.

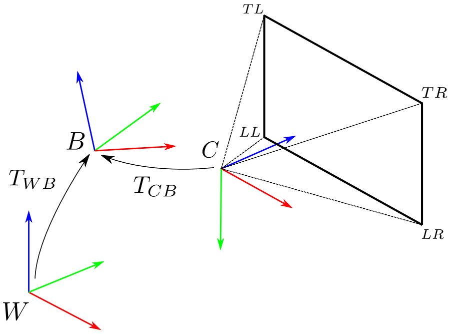


Figure 6.4: Representation of the field of view of the camera and of the reference frames. W corresponds to the odometry reference frame, while B is the body (IMU) and C the reference frame of the camera.

Notice that, in order to be able to use standard tools for numerical optimization, we need to formulate this visibility condition as a continuous, differentiable indicator function [98]. The half spaces are represented by the set of vectors connecting the optical center of the camera to the vertices of the sensor, as shown in Fig. 6.4. In the camera reference frame C, these vectors are

$$\begin{aligned} \mathbf{v}_{TR} &= f \cdot \begin{bmatrix} \sin\theta_{hor} & -\sin\theta_{ver} & 1 \end{bmatrix}^T \\ \mathbf{v}_{LR} &= f \cdot \begin{bmatrix} \sin\theta_{hor} & \sin\theta_{ver} & 1 \end{bmatrix}^T \\ \mathbf{v}_{LL} &= f \cdot \begin{bmatrix} -\sin\theta_{hor} & \sin\theta_{ver} & 1 \end{bmatrix}^T \quad , \\ \mathbf{v}_{TL} &= f \cdot \begin{bmatrix} -\sin\theta_{hor} & -\sin\theta_{ver} & 1 \end{bmatrix}^T \\ \mathbf{v}_{Z} &= f \cdot \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T \end{aligned} \tag{6.16}$$

where f is the focal length in meters.
In order to check whether a landmark lies within the camera frustum, it has to be expressed in the camera frame C; however, the set of available 3D landmarks $\mathcal{L}^W$ from the VIO module are expressed in the odometry frame W. If we indicate the subset of high-quality landmarks as $\mathcal{L}_G^W \subseteq \mathcal{L}^W$, we can express its elements $\mathbf{l}_W$ in C as follows:

$$\mathbf{l}_C = \mathbf{R}_{CW}\mathbf{l}_W + \mathbf{t}_{CW}, \tag{6.17}$$

where $\mathbf{R}_{CW} \in SO(3)$ and $\mathbf{t}_{CW} \in \mathbb{R}^3$ are the rotation matrix and the translation vector from W to C, respectively. For ease of representation, we adopt homogeneous matrices $\mathbf{T}_{AB} \in SE(3)$ to indicate transformations from B to A:

$$\mathbf{T}_{AB} = \begin{bmatrix} \mathbf{R}_{AB} & \mathbf{t}_{AB} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix}. \tag{6.18}$$

To obtain $\mathbf{T}_{CW}$ for a given control point $\mathbf{q} = [x, y, z, \theta]^T$, we proceed as follows. As the VIO estimation corresponds to the pose of the body B (generally the IMU in VIO systems) in W (i.e. $\mathbf{T}_{WB}$), we need to concatenate the fixed transformation $\mathbf{T}_{CB}$, which is known beforehand from calibration:

$$\mathbf{T}_{CW}(\mathbf{q}) = \mathbf{T}_{CB}\mathbf{T}_{WB}^{-1}(\mathbf{q}). \tag{6.19}$$

Since the control points of the B-Spline represent the body in W, we can obtain the homogeneous transformation $\mathbf{T}_{WB}$ for a control point $\mathbf{q}$ as

$$\mathbf{T}_{WB}(\mathbf{q}) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & x \\ \sin\theta & \cos\theta & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{6.20}$$

We can now define the following indicator function over the set of all high-quality landmarks expressed in the camera frame (i.e. $\mathcal{L}_G^C$) and current control point $\mathbf{q}$:

$$F_l(\mathbf{q}) = \sum_{\mathbf{l}_C \in \mathcal{L}_G^C} \prod_{j=0}^{5} o_j(\mathbf{q}, \mathbf{l}_C). \tag{6.21}$$

where o_j are the elements of

$$\mathbf{o}(\mathbf{q}, \mathbf{l}_C) = \begin{bmatrix} \frac{1}{2}\Big(1 + \tanh((\mathbf{v}_{TR} \times \mathbf{v}_{LR}) \cdot \mathbf{l}_C)\Big) \\ \frac{1}{2}\Big(1 + \tanh((\mathbf{v}_{TL} \times \mathbf{v}_{TR}) \cdot \mathbf{l}_C)\Big) \\ \frac{1}{2}\Big(1 + \tanh((\mathbf{v}_{LL} \times \mathbf{v}_{TL}) \cdot \mathbf{l}_C)\Big) \\ \frac{1}{2}\Big(1 + \tanh((\mathbf{v}_{LR} \times \mathbf{v}_{LL}) \cdot \mathbf{l}_C)\Big) \\ \frac{1}{2}\Big(1 + \tanh(\mathbf{v}_Z \cdot \mathbf{l}_C)\Big) \end{bmatrix}. \tag{6.22}$$

The final cost for the spline optimization corresponds to

$$\mathcal{F}_l = -\sum_{i=K}^{N-K} F_l(\mathbf{q}_i), \tag{6.23}$$

maximizing the number of the tracked high-quality landmarks.

The last cost function $\mathcal{F}_v$ is associated to the co-visibility between consecutive control points in the B-Spline. The purpose is to add a soft constraint on the orientations of the robot, in order to keep them aligned with the direction of travel. For a two consecutive control points $\mathbf{q}_i$ and $\mathbf{q}_{i+1}$, we compute the orientation vector in the xy plane as

$$\mathbf{n}_{i+1,i} = \begin{bmatrix} x_{i+1} - x_i & y_{i+1} - y_i \end{bmatrix}^T \in \mathbb{R}^2. \tag{6.24}$$

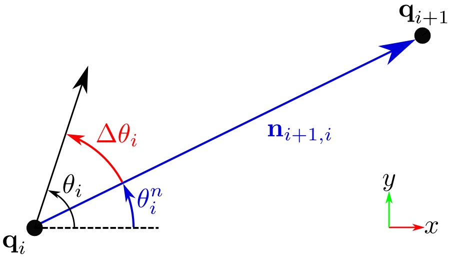


Figure 6.5: Terms for the computation of the co-visibility cost. For two consecutive control points $\mathbf{q}_i$ and $\mathbf{q}_{i+1}$, we compute the angular difference $\Delta\theta_i$ between the orientation θ_i in $\mathbf{q}_i$ and the direction vector $\mathbf{n}_{i+1,i}$.

We do not use the z coordinates, as the orientation is expressed as a rotation around the Z-axis; therefore, the z component does not have any effect on the cost. As shown in Fig. 6.5, the orientation of $\mathbf{n}_{i+1,i}$ is $\theta_i^n = \theta_i^n(\mathbf{q}_i, \mathbf{q}_{i+1}) = atan2(n_{i+1,i}^y, n_{i+1,i}^x)$, where the superscript indicates the corresponding component in the vector. For the control point $\mathbf{q}_i$, we can compute the absolute difference in orientation with respect to the direction vector $\mathbf{n}_{i+1,i}$ as $\Delta\theta_i := |\theta_i - \theta_i^n|$. The cost for a couple of two consecutive control points is

$$F_v(\mathbf{q}_i, \mathbf{q}_{i+1}) = \begin{cases} (\Delta\theta_i - \Delta\theta^*)^2 & \text{if } \Delta\theta_i \geq \Delta\theta^* \\ 0 & \text{otherwise} \end{cases}, \tag{6.25}$$

where $\Delta\theta^*$ is the maximum allowed angular deviation. Finally, the cost associated to the co-visibility is

$$\mathcal{F}_v = \sum_{i=K}^{N-K-1} F_v(\mathbf{q}_i, \mathbf{q}_{i+1}). \tag{6.26}$$

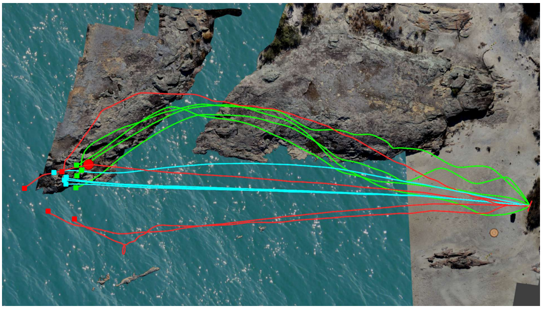
This cost tries to translate both the control points, but for a given $\mathbf{q}_i$, the successive $\mathbf{q}_{i+1}$ is not subject to changes in orientation, as the gradient of $\mathcal{F}_v$ with respect to θ_{i+1} is zero. Moreover, notice that the costs $\mathcal{F}_l$ and $\mathcal{F}_v$ are in competition with each other since the former encourages to look in the direction of the well-localized landmarks, while the latter aligns the robot orientation with the direction of travel.

6 Experiments

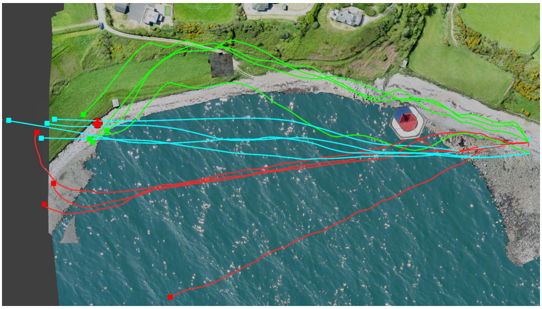
A series of challenging experiments in photo-realistic simulations using the Gazebo and RotorS frameworks [48] were conducted to show the effectiveness of the proposed method. The simulated UAV is an AscTec Firefly with a front-looking camera mounted with a pitch angle of 30°. Differently from the experiments reported in [162], the simulation environments do not present texture-less areas; instead, some parts of the models were specifically built to create hard chal-

Experiment	Dimensions	Ours	Reactive	[162]
Rocks	$30m \times 50m$	**1.46 m**	2.98 m	4.96 m
Bay	$40m \times 55m$	**1.62 m**	6.58 m	7.28 m
Long Beach	$100m \times 190m$	**6.05 m**	16.67 m	9.24 m

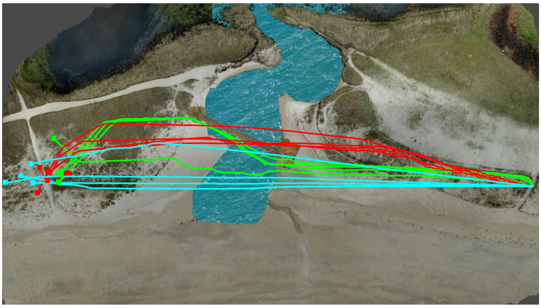
Table 6.1: Results of the experiments in the three simulations. We report the average missed distance, i.e. the distance between the final position of the UAV and the real goal position.



(a) Top-view of the experiment *Rocks*. Only our planner (green) is able to consistently fly the UAV on top of the coast and reach the goal.



(b) Top-view of the experiment *Bay*. Despite the presence of the obstacle, our planner (green) is able to steer the robot towards more informative areas. In this experiment, [162] (red) performs worse than the reactive planner (cyan), causing a complete failure case, since it follows the low-quality landmarks extracted on the water surface. Moreover, both the reactive planner and [162] fly the robot over the boundaries of the model.



(c) Top-view of the experiment *Long Beach*. Our planner (green) is able to steer away from dangerous areas (e.g. lake), by flying close to texture-full structures, such as the bridge in the middle of the model.

Figure 6.6: Top-views of the experiments in photo-realistic simulations. The outputs of the proposed planner are depicted in green, while the paths in cyan and in red correspond to a purely reactive planning and the perception-aware planner proposed in [162], respectively. The goal position is depicted as a red blob and the end positions of the trajectories are indicated with a square.

lenges for camera-based state estimators, such as water with moving waves. We show that our planner is able to steer the robot away from these problematic areas, enhancing the performance of the VIO system.

We tested three different models[1], shown in Fig. 6.6. In each scenario, we commanded the UAV to fly to a predefined destination 4 times. The input goal is initially expressed in the reference frame of the simulator and then is transformed into the local reference frame of the robot. Notice that the initial positions of the robot can slightly vary between different runs of the same experiment, due to the initialization procedure of the VIO and MSF modules.

In order to evaluate the approach, we use the state estimation error as a metric, especially because the UAV did not crash in any obstacles in any experiment, despite the high error in localization. This is an advantage of using local information (e.g. local occupancy maps) for obstacle avoidance. For each experiment, we report the missed distance to the goal, i.e. the distance between the end position of the robot and the real position of the goal. The results are reported in Table 6.1. In Fig. 6.7 we plot the absolute error in position with respect to travelled distance, where the error is computed as the absolute difference from the estimated position of the UAV with respect to the ground truth.

We compare the performance of our method with a purely reactive navigation approach and the perception-aware planner proposed by Zhang and Scaramuzza [162]. The reactive planner is based on our proposed planner without considering the perception costs, i.e. setting w_M in Eq. (6.3) and the weights λ_l and λ_v in Eq. (6.10) to zero.

---

[1] `https://sketchfab.com/3d-models`

The experiments are visualized in the accompanying video.

The first scene *Rocks* (Fig. 6.6a) consists of a rocky coast cut in between by a medium-size river. The left half of the model contains just water, which is moving and reflecting light. As there are no obstacles blocking the path, we observe that the purely reactive planner computes mostly straight line trajectories to the goal to the other end of the scene. However, these fly the robot in areas of low-quality perception, resulting in large state estimation errors. The perception-aware planner of [162] is also not able to steer the robot towards more informative areas (e.g. the coast), since it uses all the available landmarks extracted from VINS-Mono and does not filter the high-quality from the low-quality ones. It even performs worse than the reactive planner, since it tries to keep track of the landmarks extracted on the water surface, causing failures in the estimation. On the contrary, our planner is able to use information on more informative areas, encouraging the navigation along the coast. This leads to a noticeable improvement in accuracy with respect to the other two planners, at the expense of slightly higher trajectory lengths, as shown in Fig. 6.7a.

Similarly, the second scene *Bay* contains moving water in the middle, which lies between the starting point and the destination. A top view is shown in Fig. 6.6b. Moreover, close to the initial position of the robot, we place a large obstacle (lighthouse), which we label as useful for localization purposes (Fig. 6.1). The aim of this experiment is to show both the obstacle avoidance capability of the planners and the impact obstacles have on the perception cost. The lighthouse acts as an attractor for the perception cost, as it is rich in high-quality features. On the other hand, it also works as a repeller from the obstacle avoidance perspective. It is noticeable how our planner is able to satisfy the perception constraint while avoiding the lighthouse, by flying to its right along the coast. The other methods are also able to avoid collisions, but the obstacle pushes the trajectories towards low-quality parts of the scene, causing large drift and, consequently, higher missed distances.

The last experiment *Long Beach* (Fig. 6.6c) presents a set of dangerous areas, such as water and almost texture-less desert-like parts. In the middle of the model, we place a bridge of top of water, which can be utilized for localization during navigation. Despite that our planner does not navigate the robot on top of it, the drift in the estimation remains limited because our planning strategy encourages the robot to face towards the landmarks extracted on the structure of the bridge. While the path-search step initializes the trajectory next to this structure, the trajectory optimizer enhances the triangulation and tracking of the high-quality landmarks. In this experiment, [162] performs better than the reactive planner, because it steers the robot towards textured areas, while the purely reactive strategy flies the robot on top of the desert-like part of the scene. Thus, [162] improves the localization accuracy by avoiding texture-less areas, but it shows worse performance in comparison to our approach.

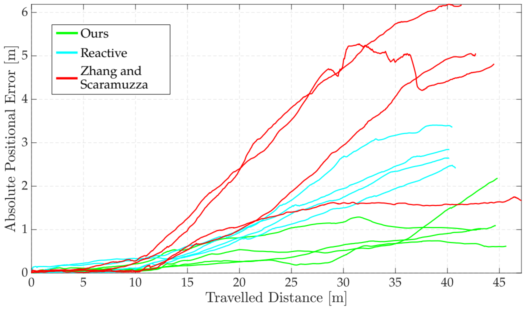
Overall, our method performs better than purely reactive planning and the perception-aware planner [162], as in all the experiments, the missed distance to the goal is significantly reduced, at the expense of slightly bigger trajectory length. Moreover, as shown in Fig. 6.7, the estimation error in position remains bounded, showing a noticeable increase in performance with respect to our competitors.

7 Conclusion and Future Work

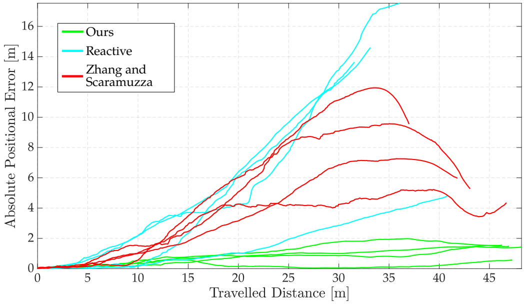
In this paper, we propose a perception-aware path-planning architecture for goal reaching tasks. Our approach pushes the boundaries of the state of the art by incorporating the semantic labels of the 3D landmarks tracked by a VIO system into the path-planning instead of using only the existence of landmarks in certain areas as information.

This design allows us to navigate through areas with harder perceptual conditions. We demonstrate that we can reach the assigned goals with limited error and drift, showing noticeable improvements in performance with respect to purely reactive planning and the latest available solution in state-of-the-art of active perception.

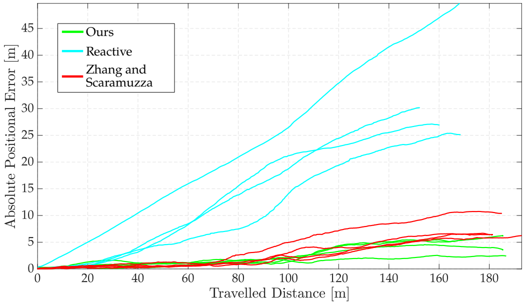
Future directions include investigating the incorporation of the uncertainty of labels in semantic segmentation and feature classifiers when deployed in aerial vehicles. Another direction is to extend this pipeline to work with emergency landing spots detection, where the robot should also favourite to stay at a close distance from them.



(a) Plot of the absolute positional error in the *Rocks* experiment.



(b) Plot of the absolute positional error in the *Bay* experiment.



(c) Plot of the absolute positional error in the *Long Beach* experiment.

Figure 6.7: Absolute error in position of the state estimate with respect to increasing travelling distance.

Paper IV

Semantic-aware Active Perception for UAVs using Deep Reinforcement Learning

Luca Bartolomei, Lucas Teixeira and Margarita Chli

Abstract

This work presents a semantic-aware path-planning pipeline for Unmanned Aerial Vehicles (UAVs) using deep reinforcement learning for vision-based navigation in challenging environments. Driven by the maturity of works in semantic segmentation, the proposed path-planning architecture uses reinforcement learning to distinguish the parts of the scene that are perceptually more informative using semantic cues, in effect guiding more robust, repeatable, and accurate navigation of the UAV to the predefined goal destination. Assuming that the UAV performs vision-based state estimation, such as keyframe-based visual odometry, and semantic segmentation onboard, the proposed deep policy network continuously evaluates the optimal relative perceptual informativeness of each semantic class in view. A perception-aware path planner uses these informativeness values to perform trajectory optimization in order to generate the next best action with respect to the current state and the perception quality of the surroundings, essentially guiding the UAV to avoid flying over perceptually degraded regions. Thanks to the use of semantic cues, the policy can be trained in a large number of non-photorealistic randomly-generated scenes, and results to an architecture that is generalizable to environments with the same semantic classes, independently of their visual appearance. Extensive evaluations on challenging, photorealistic simulations reveal a remarkable improvement in robustness and success rate with the proposed approach over the state of the art in active perception.

Published in:
IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2021
DOI: 10.1109/IROS51168.2021.9635893

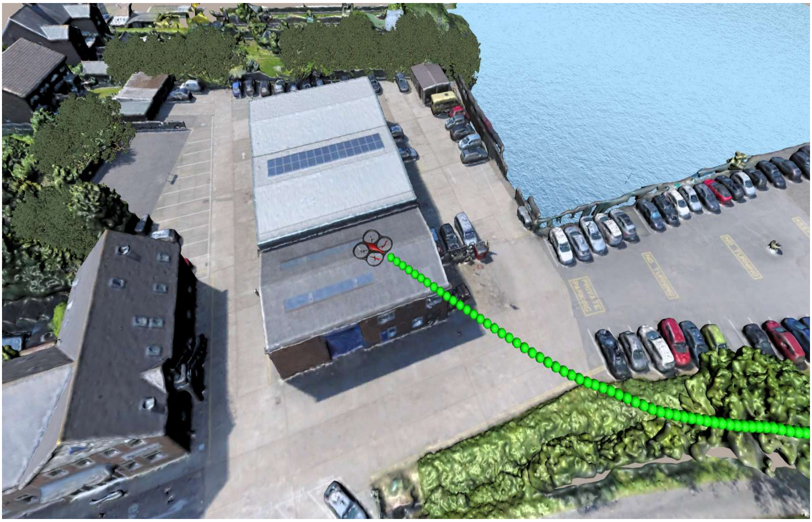


Figure 7.1: 3D-view of the path flown by the UAV towards the destination in the *Baxall* experiment. The travelled path (in green) indicates that the robot is able to successfully avoid areas that are problematic for camera-based state estimators, such as water, while navigating on top of reliable textures, such as terrain and buildings.

1 Introduction

The capability of a robot to achieve precise localization and perform safe navigation is a fundamental skill for genuinely autonomous systems. In the case of Unmanned Aerial Vehicles (UAVs), these abilities play an even more significant role due to the agility of these platforms. For example, in industrial inspection and package delivery, UAVs have to map the environment with sufficient accuracy to avoid collisions with obstacles and reach the goal position accurately. However, both real-time localization and mapping are still open problems since the most used sensors, such as GPS, can fail numerous situations due to bad weather, jamming, or in the presence of mountains or tall buildings. Avoiding failures in localization systems for aerial robots is fundamental because, as opposed to ground robots, they cannot instantaneously halt all motion, but instead, they either need to reach an emergency landing spot or to continue flying towards their destination. Vision-based Simultaneous Localization And Mapping (SLAM) is now well accepted for UAV navigation [138]; however, its performance (i.e. robustness, accuracy) is heavily influenced by the condition of the navigation environment. For example, vision-based SLAM's accuracy degrades significantly (and even fails) in the presence of dynamic objects (e.g. people, cars, trees moving in the wind) and areas with no texture or textures exhibiting specularities (e.g. oceans, lakes).

This work proposes an active perception path-planning algorithm for vision-based UAV navigation based on Reinforcement Learning, guiding the UAV to reach a predefined goal position while avoiding texture-less and poorly textured areas, as well as regions that are less reliable for localization, such as lakes, streets with moving cars, and trees (Fig. 7.1). Given as input a semantic mask of the scene, the proposed framework outputs online the optimal policy that allows steering the robot away from potentially dangerous areas by assigning a perceptual informativeness score to each semantic class. We train our policy in a set of non-photorealistic randomly-generated 3D models, and we test it in a set of previously unseen environments, including photorealistic 3D models of real-life places. The algorithm is agnostic to the semantic segmentation algorithm and works with any set of semantic classes.

In brief, the contributions of this work are the following:

- the design of a reinforcement learning-based semantic-aware path-planning algorithm for vision-based aerial navigation in challenging environments,
- the design of a training framework and a policy architecture able to generalize to unseen environments, and
- an extensive quantitative evaluation of the performances of the proposed system with respect to the state-of-the-art in active perception in photorealistic simulations.

2 Related Works

The performance of Visual SLAM algorithms, providing estimates of the robot's pose and surroundings, is greatly influenced by the motion and the path followed by the robot, as sharp and highly dynamic movements can lead to wrong estimates or even failure. Active perception constitutes a fundamental step towards the creation of fully autonomous systems, able to cope with the uncertainties present in a real mission. The core concept in active perception is that sensory performance can be improved by proper selection of motion-control actions [8]. Modern works propose solutions integrating perception, path planning and control in a unified framework [2, 7]. From a theoretical perspective, the problem of active perception is formulated as Partially Observable Markov Decision Processes (POMDPs) [66], which are in general complex to solve. The drive to find efficient solutions led to the conception of belief roadmaps [114] and belief trees [18], while more recently, receding horizon approaches and perception-aware path-planning algorithms emerged [104, 162], showing how texture-less regions of the environment could be avoided [33] using the internal state of a SLAM system.

One of the most relevant rival to this work is the perception-aware planner in [162], proposing to generate motion primitives and evaluate them by considering the concentration of landmarks in each area, the probability of collision and the distance to the goal. In [12], however, it is demonstrated that landmark concentration alone is not enough to identify the best areas to fly through, and instead, a perception-aware planner employing semantics to evaluate the quality of the candidate areas for navigation is proposed. Using the semantic segmentation of the SLAM input image as an additional cue in a perception-aware planning algorithm, [12] was shown to achieve the best results in terms of accuracy and robustness of localization to date. Nonetheless, it is not able to adapt dynamically to changes in the navigation area at flight time,

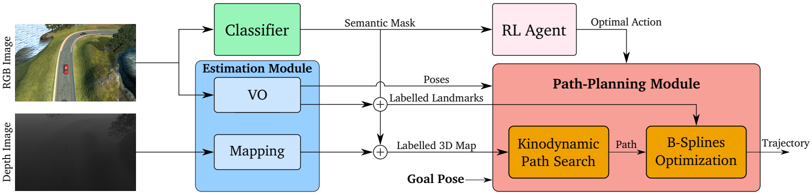


Figure 7.2: Schematic overview of the pipeline. In the Estimation Module, we process the sensor inputs to estimate the pose of the UAV and the 3D landmarks. The landmarks, together with the occupancy map obtained from the depth images, go through a classification step, where each point is assigned to a semantic class. The RL agent utilizes the semantic mask to generate the optimal action, which consists of the set of weights to assign to each semantic class. The optimal choice of weights is then communicated to the perception-aware path planner from [12]. Finally, the Path-Planning Module outputs the optimal trajectory, while considering both the dynamics of the platform and perceptional quality.

as it assigns fixed binary informativeness scores to every semantic class in the scene. Furthermore, this approach requires manual score definition, which can become challenging with an increasing number of classes. Addressing this, this work proposes a pipeline that tackles these limitations using reinforcement learning-based active perception, dynamically identifying the most reliable regions for localization from semantic information. Reinforcement Learning (RL) can allow autonomous systems to learn policies for complex tasks, such as control [63] and obstacle avoidance [68], reducing the engineering effort to the design of a suitable reward function [110]. In combination with deep neural networks, deep RL provides a powerful tool capable of mapping high-dimensional sensory inputs to optimal actions, showing promising results in fields, such as autonomous driving [148] and robot navigation [67, 161].

In this spirit, this work tackles the limitations of [12] using a RL-trained deep-policy-based perception-aware path-planning pipeline that is able to reduce the drift in vision-based SLAM, without the need for manual setting of informativeness scores, but with real-time and onboard auto-tuning of scores instead. In a nutshell, given an input semantic mask of the scene, the proposed planner learns to encourage navigation through regions suitable for visual localization, by adapting to the scene online and assigning different importance to the different semantic classes. Our approach exhibits great improvement in the success rate of missions when compared to purely reactive planning and to state-of-the-art perception-aware planners, such as the ones proposed in [162] and [12].

3 Method

The problem targeted in this work is for a UAV to reach a predefined goal pose, while minimizing the drift in the onboard vision-based SLAM algorithm. Our main objective is to identify and fly through suitable areas for visual localization, avoiding spatial regions that lead to high error and pose estimation failure. We formulate this as a path-planning problem, where a deep

RL agent is trained to identify reliable areas from semantically labelled images. Semantics is a valuable source of information for the agent, as drift in pose estimation is generally consistent for areas belonging to the same semantic class in case of good illumination conditions. Moreover, using mid-level representations, such as semantic masks as input is proven to yield better generalization of the policy [27]. At the same time, our architecture allows to decouple the problems of semantic labeling and path planning. This decoupling is essential for deployability, as learning directly the mapping from raw camera data to perceptual informativeness for each semantic class requires an implicit semantic segmentation step, which would take extremely long training time in an RL fashion.

Given that real-time semantic segmentation has already reached a high degree of maturity [128], this work assumes a high-quality semantic segmentation image as input. Using an adaptation of the approach of [12], we generate smooth and collision-free trajectories in a receding horizon fashion, considering the limitations of keyframe-based Visual Odometry (VO) systems used for pose estimation. Semantic classes are mapped to a certain score of perceptual informativeness by a policy trained in randomly generated scenes using Unity[1] and the Flightmare frameworks [132]. These scores, recorded per class, are successively fed to the path planner, which encourages navigation in areas that cause less drift and keep the best 3D landmarks estimated by the VO pipeline in the field of view of the camera.

3.1 System Overview

As shown in Fig. 7.2, the proposed pipeline consists of three main components; a monocular camera-based pose estimation module, a deep RL agent and a path-planning module. We assume a stream of RGB and depth images is available. The depth images are used to generate 3D reconstructions of the local surroundings of the robot, while the VO algorithm utilizes the RGB images to estimate the robot's pose. The dense point cloud is stored in an occupancy map employing a 3D circular buffer [143].

We employ the keyframe-based VO system ORB-SLAM [97] without loop closures to estimate the pose of the camera, but our pipeline is agnostic to the adopted VO algorithm. Since ORB-SLAM is a monocular vision-only system, the scale cannot be retrieved. However, as our agent is trained in simulation, we have access to ground-truth information that is used to re-scale the estimated position and the 3D landmarks.

Both the 3D occupancy map and the sparse landmarks go through a classification step providing the semantic labels for all the points, which are then fed to the path-planning module. Given that the detection of moving objects (e.g. cars, people, trees) and ground classification are not the core contribution of this work, we use ground-truth semantic masks in our experiments, but there are a plethora of off-the-shelf algorithms available [117, 134]. The semantic mask also serves as input to the deep RL policy, which outputs values associated with the perceptual informativeness of each semantic class. These values are communicated to the planner, which uses them to reason about the next best action. The policy output is utilized as a set of weights in the objective function to be optimized in the path-generation step, favoring the tracking and triangulation of points belonging to parts of the scene useful for camera-based state estimation. In the next section, the deep RL agent structure and its interface with the path-planning module

---

[1] `https://unity.com/`

are explained in detail.

3.2 Perception-aware Path Planning

Our objective is to let the agent fly through areas well-suited for VO by learning which semantic classes are less likely to generate localization drift. The robot learns this by interacting with the environment, selecting an action, and receiving a reward value as feedback. Here, an action corresponds to a set of weights for each semantic class in the perception objective function, to be optimized in the path-planning module. The planning pipeline is an adaptation of [12], where originally semantic weights are manually assigned, and used in a kinodynamic A^* path search, followed by B-Spline trajectory optimization.

Kinodynamic Path Search

In the first planning step, the aim is to encourage navigation in well-textured areas. The path search is limited to the robot's position in $\mathbb{R}^3$ and, using the differential flatness of the multirotor systems, the trajectory is represented as three independent time-parametrized polynomial functions $\mathbf{p}(t)$:

$$\mathbf{p}(t) := [p_x(t), p_y(t), p_z(t)]^T,\; p_d(t) = \sum_{k=0}^{K} a_{d,k} t^k \tag{7.1}$$

with $d \in \{x, y, z\}$. We assume the multirotor system to be linear and time-invariant, and we define its state as $\mathbf{s}(t) := [\mathbf{p}(t)^T, \dot{\mathbf{p}}(t)^T, \ldots, \mathbf{p}^{(n-1)}(t)^T]^T \in \chi \subset \mathbb{R}^{3n}$, with control input $\mathbf{u}(t) := \mathbf{p}^{(n)}(t) \in \mathcal{U} := [-u_{max}, u_{max}]^3 \subset \mathbb{R}^3$ and $n = 2$, corresponding to a double integrator. Given the current robot's state $\mathbf{s}(t)$, the control input $\mathbf{u}(t)$ and a labelled occupancy map $\mathcal{M}$ of the environment, we define the cost of a trajectory as

$$\mathcal{J}(T) = \int_0^T \Big(w_u \|\mathbf{u}(t)\|^2 + \sum_{j=0}^{N} w^j d_M^j(\mathbf{p}(t), \mathcal{M}) \Big) dt + w_T T, \tag{7.2}$$

where $\|\mathbf{u}(t)\|^2$ is the control cost; $d_M^j(\mathbf{p}(t), \mathcal{M})$ represents a penalty for navigating far away from areas associated to the semantic class $j \in \{0, \cdots, N\}$ with N the total amount of classes; and T is the total time of the trajectory.

The terms w_u and w_T are constant weights associated with the respective costs, while the w^j is the weight associated with the semantic class j assigned by the current optimal policy and is subjected to changes as the agent gathers additional experience. The cost $d_M^j(\mathbf{p}(t), \mathcal{M})$ is defined as

$$\begin{aligned} d_M^j(\mathbf{p}(t), \mathcal{M}) &:= \sum_{\mathbf{v}_j \in \mathcal{M}_j \subseteq \mathcal{M}} d_v(\mathbf{p}(t), \mathbf{v}_j) = \\ &\sum_{\mathbf{v}_j \in \mathcal{M}_j \subseteq \mathcal{M}} d_{xy}(\mathbf{p}(t), \mathbf{v}_j) + d_z(\mathbf{p}(t), \mathbf{v}_j), \end{aligned} \tag{7.3}$$

where $\mathbf{v}_j = [v_x, v_y, v_z]^T$ are the voxels of the occupancy map $\mathcal{M}$ with semantic label j, indicated with $\mathcal{M}_j \subseteq \mathcal{M}$. The cost $d^j_M(\mathbf{p}(t), \mathcal{M})$ is composed of the two potential functions that are calculated as

$$d_{xy}(\mathbf{p}(t), \mathbf{v}_j) := (p_x(t) - v_x)^2 + (p_y(t) - v_y)^2, \tag{7.4}$$

and, by defining $\Delta z := |p_z(t) - v_z|$,

$$d_z(\mathbf{p}(t), \mathbf{v}_j) := d^* \Delta z + \frac{1}{2} \frac{{d^*}^4}{\Delta z^2} - \frac{3}{2} {d^*}^2, \tag{7.5}$$

where d^* controls the minimum height of the robot with respect to the voxels in $\mathcal{M}_j$. In order to speed up the search in the A^* algorithm, we use the same heuristic as in [12], adapted to match the new cost definitions.

Trajectory Optimization

While the trajectory computed in the path-searching step encourages navigation towards informative areas, the trajectory optimization step leverages the additional information given by the 3D landmarks in the VO module. The trajectory $\boldsymbol{\pi}(t)$ is parametrized as a uniform B-Spline of degree K and it is defined as

$$\boldsymbol{\pi}(t) = \sum_{i=0}^{N} \mathbf{q}_i B_{i,K-1}(t), \tag{7.6}$$

where $\mathbf{q}_i$ are the control points at time t_i with $i \in \{0, \ldots, N\}$, and $B_{i,K-1}(t)$ are the basis functions. Each control point in $\{\mathbf{q}_0, \mathbf{q}_1, \ldots, \mathbf{q}_N\}$ encodes both the position and orientation of the robot, i.e. $\mathbf{q}_i := [x_i, y_i, z_i, \theta_i]^T \in \mathbb{R}^4$ with $\theta_i \in [-\pi, \pi)$. The B-Spline is optimized in order to generate smooth, collision-free trajectories, encouraging the triangulation and tracking of high-quality landmarks. For a B-Spline of degree K defined by $N+1$ control points $\{\mathbf{q}_0, \mathbf{q}_1, \ldots, \mathbf{q}_N\}$, our optimization acts on $\{\mathbf{q}_K, \mathbf{q}_{K+1}, \ldots, \mathbf{q}_{N-K}\}$ while keeping the first and last K control points fixed due to boundary constraints. The optimization problem is formulated as a minimization of the cost function

$$\mathcal{F}_{TOT} = \lambda_s \mathcal{F}_s + \lambda_f \mathcal{F}_f + \lambda_c \mathcal{F}_c + \lambda_l \mathcal{F}_l + \lambda_v \mathcal{F}_v, \tag{7.7}$$

where $\mathcal{F}_s$ is the smoothness cost; $\mathcal{F}_c$ is the collision cost; $\mathcal{F}_f$ is a soft limit on the derivatives (velocity and acceleration) over the trajectory; $\mathcal{F}_l$ is the penalty associated with losing track of the high-quality landmarks currently in the field of view; and $\mathcal{F}_v$ is a soft constraint on the co-visibility between control points of the spline. The coefficients $\lambda_s, \lambda_c, \lambda_f, \lambda_l$ and λ_v are the fixed weights associated to each cost. While we maintain the original cost formulations, similarly to Eq. (7.2), we propose a novel perception cost that accommodates multiple semantic

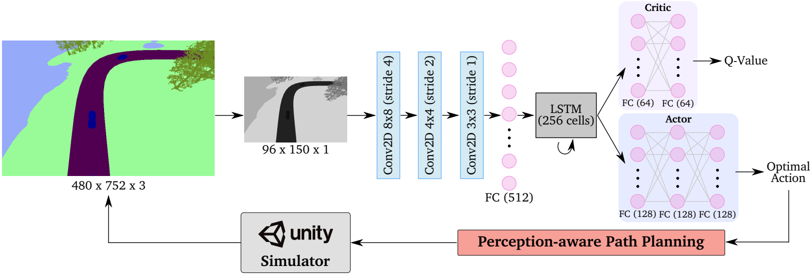


Figure 7.3: Schematic representation of the policy architecture and its interaction with the path planner. The semantic images are rendered in the Unity engine and are first converted into grayscale images and downsampled. The obtained images go through the Actor-Critic network, composed of three CNN layers, a linear FC layer and a LSTM module. The Critic then outputs the Q-Value from two FC layers, while the Actor feeds the optimal action to the planner, consisting of a set of weights associated to each semantic class. The policy optimization is performed by means of the PPO algorithm [124].

classes:

$$\mathcal{F}_l = -\sum_{j=0}^{N} \sum_{\mathbf{l}_C \in \mathcal{L}_j^C} w^j \prod_{k=0}^{5} o_k(\mathbf{q}, \mathbf{l}_C), \tag{7.8}$$

where $\mathcal{L}_j^C$ is the set of 3D landmarks associated to class j expressed in camera frame C, and o_k a smooth indicator function determining the visibility of landmark $\mathbf{l}_C$ from the control pose $\mathbf{q}$ [12]. The optimal set of weights for each semantic class is computed in real-time by a policy modeled as a neural network, trained in an episode-based deep RL-fashion. In the next section, we introduce our policy architecture and explain the training process in detail.

3.3 Deep RL Policy

The RL policy maps from semantic masks to actions, employing an Actor-Critic model with the architecture shown in Fig. 7.3. Here the action consists of the set of weights $w^j \in [0, 1]$ with $j \in \{0, \cdots, N\}$ used by the planner in Eq. (7.2) and Eq. (7.3), where N is the number of semantic classes in the scene fixed by the user. The Actor and the Critic networks share the first part, composed of a 3-layer Convolutional Neural Network (CNN), followed by a Long-Short Term Memory (LSTM) module. The LSTM is responsible for the memory of the policy and captures spatial dependencies that would otherwise be hard to identify, as some semantic classes can be linked together (e.g. cars and roads). The final part of the Critic consists of two Fully Connected (FC) layers composed of 64 units, while the optimal action is output by the Actor from three FC layers with 128 units each. Policy optimization is then performed at fixed-step intervals employing the on-policy algorithm PPO [124]. Moreover, in order to reduce the hyperspace dimension, we convert the color mask into grayscale and downsample the resulting image. We utilize the semantic image as input because this mid-level visual representation is more generic than the raw color image, and it is demonstrated to allow faster training and improved policy performances [27]. Using the raw image from the sensor directly makes the training cumbersome and generalization of the policy harder, as the same semantic class can have different textures and visual appearances (e.g. in the cases of terrain, vegetation, and cars).

The training of the policy is then performed based on the data and the rewards collected in each episode. Since we aim to reduce the localization error and increase the chances of getting to the destination, the reward function received by the RL agent at step t is defined as

$$R_t(\mathbf{p}(t), e(t)) := R_S + w_E R_E(e(t)) + w_G R_G(\mathbf{p}(t)), \tag{7.9}$$

where R_S is the survival reward, R_E is associated to the localization error $e(t)$ and R_G to the progress towards the goal position. The survival reward is assigned at every step, until tracking is lost:

$$R_S := \begin{cases} 0 & if\ lost\ track \\ 1 & otherwise \end{cases}. \tag{7.10}$$

Note that we do not penalize explicitly with a negative final reward when the VO system loses track, in order not to penalize promising actions that lead to high errors due to faulty initialization of the visual tracks at the beginning of the episode.

The reward associated to the localization error is also assigned in every step, and encourages to take actions that reduce the drift in the VO system:

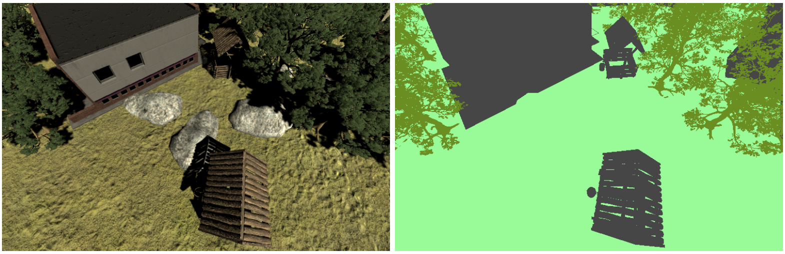
$$R_E(e(t)) := \begin{cases} R_E^{max} & if\ e(t) \leq e_{min} \\ 0 & if\ e(t) \geq e_{max} \\ R_E^{max} exp(-(e(t) - e_{min})) & otherwise \end{cases}, \tag{7.11}$$

where e_{min} and e_{max} are the minimum and the maximum acceptable errors, respectively, and R_E^{max} is the maximum reward value. Finally, the last component of the reward function favours the progress towards the goal position $\mathbf{p}_G(t)$ and is inversely proportional to the distance between the current robot position and the destination:

$$R_G(\mathbf{p}(t)) := R_G^{max} \frac{1}{\|\mathbf{p}(t) - \mathbf{p}_G(t)\|}, \tag{7.12}$$

where R_G^{max} is the maximum achievable reward value. When the agent reaches the goal, it receives a final reward equal to R_G^{max}.

At the beginning of each episode, we place the robot at a given starting position, initialize the VO tracking system, and set an end target position. The agent navigates towards the goal generating trajectories by optimizing the cost functions defined in Eq. (7.2) and Eq. (7.7), given the set of weights output by the current policy. During the flight, the agent monitors the localization error and the episode ends when either the goal is reached or the VO system loses track.



(a) Visual Image. **(b)** Semantic Mask.

Figure 7.4: Example of visual image and its corresponding semantic mask rendered by the Unity engine at training time. The training scenes are generated by the game engine, placing objects (e.g. trees, rocks, cars, houses) randomly in the space.

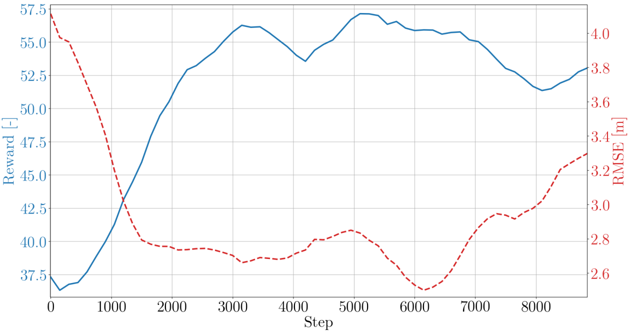


Figure 7.5: Reward and RMSE trends over the training steps, shown as blue solid and red dashed curves, respectively. In the initial phase of training, the sharp increase in the reward demonstrates that the RL agent learns quickly the most reliable semantic classes for localization, followed by a plateau, where the optimal policy is reached.

4 Experiments

4.1 Policy Training

In order to maximize the generalization of the policy and to avoid overfitting to a specific scene, we train the agent in a set of randomly generated environments using the game engine Unity and the Flightmare framework [132]. The simulated UAV is equipped with a front-looking camera mounted with a pitch of 60°. The Unity engine provides the images required by the

VO systems, the semantic masks, as well as the depth images to perform obstacle avoidance. Based on [56], the depth images are corrupted by zero-mean Gaussian noise in order to mimic the noise in real sensors, such as stereo or depth cameras.

At the beginning of each episode, a new scene is generated, and a goal destination is placed randomly in the scene (Fig. 7.4). The UAV starts navigating towards the goal position, and the episode ends when either the goal is reached or the VO system loses track. The agent outputs actions at fixed time intervals (or *steps*), communicates them to the perception-aware planner, and collects the reward as feedback. In the first episode of the training process, the policy is initialized randomly. The training continues until the maximum number of steps across all episodes is reached. Here, we set this maximum to 9000 steps.

All scenes are characterized by the same semantic classes (pavement/road, terrain, water, trees, buildings and cars). We use common classes used by several state-of-the-art semantic-segmentation algorithms, but our system can handle any set of classes. The reward parameters are set to $R_E^{max} = 5$ and $R_G^{max} = 50$, with minimum and maximum localization errors $e_{min} = 0.5\,\mathrm{m}$ and $e_{max} = 2.5\,\mathrm{m}$. To compute the total reward as in Eq. (7.9), the weights for components associated to the localization error and to the goal-reaching task are set to $w_E = 3$ and $w_G = 0.1$, respectively.

The training performance of the agent is shown in Fig. 7.5, where the reward and the Root Mean Square Error (RMSE) of the VO system with respect to ground-truth position are reported. As depicted by the initial sharp increase in the reward curve, the agent learns quickly to identify the semantic classes that allow robust localization, resulting in a decrease in the pose estimation error. The training performance successively decreases, as visible from the plateau in the reward curve and the small increase in the translational error. Despite the decrease due to slightly higher RMSE, the reward does not drop, as the agent is able to reach the target destination more frequently. This indicates that the optimal behavior is reached and that the oscillations in performance are linked more to the randomness of the scene and consequently, of the VO algorithm's performance rather than to wrong action selection.

4.2 Tests in Previously Unseen Environments

A series of experiments in challenging scenes are conducted to show the effectiveness of the proposed method. The test environments are not experienced by the policy at training time, and, similarly to [12], they are specifically built to create problematic situations for camera-based odometry algorithms, such as water with moving waves, cars, and trees moving in the wind. The scenes are either built from scratch using the Unity game engine or obtained from 3D models of real-life places from photogrammetry[2]. In total, we test six different scenes as shown in Fig. 7.6.

Test Set-Up

In each scenario, we command the UAV to fly to a set of 20 goal positions placed randomly within a target area. To evaluate the approach, we use the state estimation error as a metric since we do not encounter crashes against obstacles, despite high localization errors, thanks to

---

[2] `https://sketchfab.com`

Experiment	Ours	[12]	[162]	Reactive
Racetrack				
Success Rate [%]	**100**	80	35	40
RMSE [m]	**2.4 ± 0.2**	2.5 ± 0.2	8.4 ± 4.0	6.4 ± 0.9
Missed Distance [m]	**4.5 ± 0.7**	4.6 ± 0.4	17.3 ± 11.8	6.5 ± 1.3
Path Length [m]	458.4 ± 11.1	464.1 ± 7.4	476.4 ± 56.0	**402.4 ± 1.3**
Villages				
Success Rate [%]	**90**	85	10	75
RMSE [m]	**1.2 ± 0.1**	1.3 ± 0.4	2.8 ± 1.0	5.8 ± 1.3
Missed Distance [m]	**3.8 ± 0.9**	4.5 ± 1.2	9.8 ± 1.5	10.8 ± 2.5
Path Length [m]	378.2 ± 12.8	377.5 ± 10.5	343.7 ± 3.5	**338.4 ± 6.5**
Highway				
Success Rate [%]	**95**	90	35	80
RMSE [m]	**2.1 ± 0.4**	6.1 ± 1.4	3.1 ± 2.1	2.7 ± 0.5
Missed Distance [m]	**6.4 ± 1.2**	8.6 ± 1.1	8.6 ± 3.9	**6.4 ± 0.6**
Path Length [m]	**583.7 ± 33.3**	638.0 ± 26.8	673.2 ± 11.3	660.0 ± 88.1
Baxall				
Success Rate [%]	**100**	**100**	45	70
RMSE [m]	**0.9 ± 0.2**	2.2 ± 0.2	3.7 ± 2.3	1.8 ± 0.2
Missed Distance [m]	**1.8 ± 0.3**	2.8 ± 0.7	3.6 ± 0.8	2.5 ± 1.2
Path Length [m]	439.0 ± 10.7	424.3 ± 12.1	738.1 ± 10.9	**397.8 ± 8.3**
Fraser				
Success Rate [%]	**100**	90	10	20
RMSE [m]	**1.3 ± 0.1**	2.3 ± 0.3	2.0 ± 0.1	2.3 ± 0.3
Missed Distance [m]	**5.0 ± 1.4**	5.7 ± 1.4	5.4 ± 0.1	5.2 ± 1.6
Path Length [m]	**644.4 ± 9.6**	652.7 ± 9.8	747.5 ± 10.0	678.9 ± 40.3
House Garden				
Success Rate [%]	**100**	70	15	20
RMSE [m]	**0.8 ± 0.1**	2.6 ± 0.7	1.6 ± 0.5	2.1 ± 0.1
Missed Distance [m]	**2.6 ± 0.4**	5.5 ± 0.9	3.8 ± 0.5	7.1 ± 0.3
Path Length [m]	435.1 ± 17.8	500.0 ± 81.9	**420.3 ± 6.8**	503.1 ± 79.0

Table 7.1: Results of the experiments in the six simulated scenes. We report the success rate of the goal-reaching task for the different planners, as well as the average localization RMSE, the missed distance from the goal and path length, all averaged over 20 runs. The averages are computed only for the experiments where the goal is reached. The first three scenes (*Racetrack*, *Villages* and *Highway*) are built using the game engine Unity, while the others (*Baxall*, *Fraser* and *House Garden*) are 3D photorealistic models of real-life places obtained from photogrammetry. The best performance in each scene is shown in bold.

the usage of local occupancy maps for obstacle avoidance. We compare the performance of our learning-based method with a purely reactive planning strategy, the semantic-based path planner from [12], and the perception-aware method proposed in [162]. The reactive planner is based on our planner without considering perception, i.e. ignoring the action from the RL agent and setting the weight w^j in Eq. (7.2) and λ_l and λ_v in Eq. (7.8) to zero.
The semantic-based planner of [12] uses its original implementation, but we modify it to accommodate an unlimited number of semantic classes, since the original version considers only two (i.e. terrain and water). We tune it once for all scenes to trust only reliable classes for vision-based localization, such as terrain, buildings and roads. As this planner assigns binary weights to each class, it is conceptually equivalent to a supervised learning classifier, trained to discriminate only between reliable and unreliable classes, but unable to adapt dynamically to the environment.
In all the scenes, the goal positions are the same for all the planners for fairness of comparisons, while the initial positions may differ by a couple of meters due to the movements necessary to initialize ORB-SLAM.

Result Metrics

For each experiment, we report the success rate, the translational RMSE of the VO system, the traveled path length, and the missed distance, i.e. the distance between the end position of the robot and the real position of the goal. We define a run to be successful if the goal is reached and the VO system does not lose track. Notice that we do not explicitly include the missed distance in the reward function at training time, as it is included implicitly in the minimization of the estimation error. The results are summarized in Table 7.1, where the numerical averages are computed only for the successful runs. Fig. 7.6 shows the most representative trajectories for the different planners in the test scenes.

Result Discussion

Our deep RL-based planner reaches the highest success rate across all the experiments, showing an increase in robustness compared to the other planning strategies. More importantly, we demonstrate that our policy network can safely transfer from the randomly generated training scenes to the test environments without the need of additional fine-tuning. We also experience a general decrease in the estimation errors, especially in the experiments for the *Villages*, *Highway*, *Baxall* and *Fraser* scenes. The trajectories generated by our deep RL-based planner favor flights through the most reliable regions of the environments, such as terrain and roads, avoiding moving water and trees. However, as our approach and [12] have the same path-generation back-end, the localization errors of these pipelines are similar, since the performance of vision-based state estimators is directly affected by the path followed by the camera. This behavior is expected, since a qualitative comparison of the trajectories generated by the two planners shows their similarity, especially in *Racetrack* (Fig. 7.6a) and *Villages* (Fig. 7.6b). Moreover, [12] is tuned manually to assign a higher weight on the most reliable semantic classes in the scene, and, consequently, we expect that the behavior of their planner is close to optimal. Nonetheless, our RL-based approach reaches higher robustness, as the possibility to change the weighting scheme of the semantic classes online allows to better adapt to changes in the scene and to avoid areas with unreliable texture for localization. This indicates that, while semantics are a powerful source of information, a binary approach, as in the case of supervised learning-based planners, is not capable of handling more complex scenes and safety requirements necessary in real-world applications.

Instead, the reactive planning strategy takes the shortest path to the goal positions, forcing the robot to fly through areas, such as water and woods that decrease the success rate and the localization accuracy. This effect is exacerbated in the experiments *Villages* and *House Garden*.

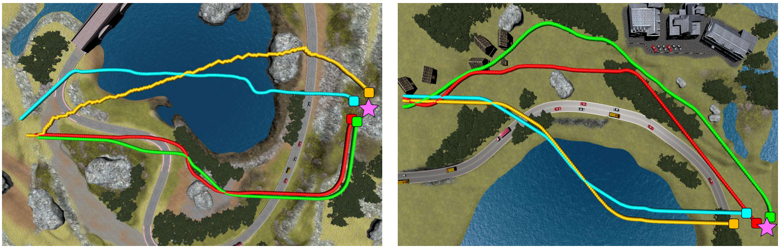
Finally, the planner by Zhang and Scaramuzza [162] has the lowest success rate in almost all the scenes. As this planner uses the landmark concentration to find the best route, it follows trails of landmarks without differentiating them on their appearance. Thus, it flies the robot to unnecessary detours trusting unreliable textures for localization (*Baxall*, *Fraser*), causing larger drift and failures in the state estimation algorithm.

Overall, while the different planning strategies give comparable performance of the vision-based state estimation (in case of a successful run), our method exhibits a remarkable and consistent increase in robustness and success rate for goal-reaching tasks in challenging scenarios. Thanks to its capacity of adapting dynamically to the scene, our deep RL-based planning pipeline can fly the UAV towards the destination safely and accurately, demonstrating a noticeable boost in performance compared to the state of the art.

5 Conclusion and Future Work

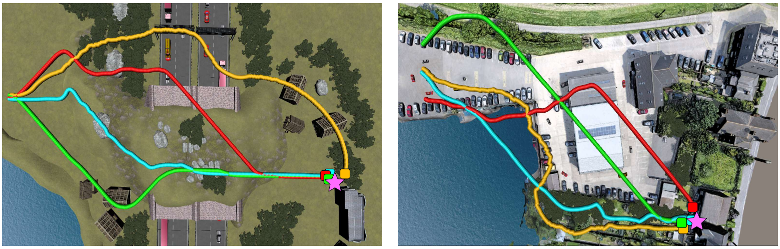
In this paper, we propose a deep RL-based perception-aware path-planning architecture for goal-reaching tasks. Our approach pushes the limits of the state of the art by incorporating reinforcement learning into semantic-based active perception, allowing the robot to learn how to adapt dynamically to changes in the navigation area. Our RL agent can map from semantic to perceptual informativeness by assigning importance weights to each semantic class in the scene. We demonstrate that our policy architecture can be easily generalizable without fine-tuning to a set of testing environments not experienced at training time. Our design allows us to navigate through areas with more challenging perceptual conditions, showing an increase in success rate and robustness with respect to purely reactive planning and the latest available solutions in state-of-the-art of active perception.

Future directions include investigating the incorporation of semantic segmentation networks in the pipeline in replacement of high-quality masks, and the adaptation of the policy network to handle the noise and uncertainty of labels in semantic segmentation.



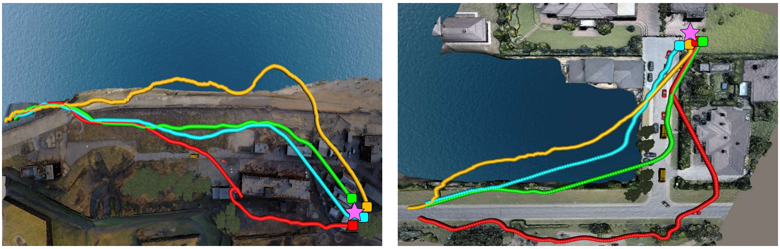
(a) Top-view of the scene *Racetrack*.

(b) Top-view of the scene *Villages*.



(c) Top-view of the scene *Highway*.

(d) Top-view of the scene *Baxall*.



(e) Top-view of the scene *Fraser*.

(f) Top-view of the scene *House Garden*.

Figure 7.6: Top-views of the test scenes with example trajectories for the different planners, with the goal position represented with a magenta star and final positions as colored squares. As an illustration, we select one good representative successful trajectory for each planner in every scene. While our planner (green) is consistently able to avoid dangerous areas by adapting dynamically to the environment, the semantic-aware planner [12] (red) adopts a binary weighting scheme for the different semantic classes, leading to worse success rates. The reactive planner (cyan) instead finds the shortest path to the goal passing through dangerous areas (e.g. lakes, woods), while the perception-aware planner of [162] (orange) follows trails of unsuitable landmarks for localization, causing large detours and failures in the camera-based state estimation system.

Paper V

Autonomous Emergency Landing for Multicopters using Deep Reinforcement Learning

Luca Bartolomei, Yves Kompis, Lucas Teixeira and Margarita Chli

Abstract

This work presents a pipeline for autonomous emergency landing for multicopters, such as rotary wing Unmanned Aerial Vehicles (UAVs), using deep Reinforcement Learning (RL). Mechanical malfunctions, strong winds, sudden battery life drops (e.g, due to cold weather), failure in localization or GPS jamming are not uncommon and all constitute emergency situations that require a UAV to abort its mission early and land as quickly as possible in the immediate vicinity. To this end, it is crucial for a UAV that is deployed in real missions to be able to detect a safe landing spot efficiently and proceed to land autonomously, avoiding damage to both its integrity and the surroundings. Driven by the advances in semantic segmentation and depth completion using machine learning, the proposed architecture uses deep RL to infer actions from semantic and depth information, flying the robot towards secure areas, while respecting safety constraints. Thanks to our robust training strategy and the choice of these mid-level representations as input to the RL agent, we show that our policy can directly transfer to the real world, without the need for any additional fine-tuning. In a series of challenging experiments both in simulation and with a real platform, we demonstrate that our planner guides a rotorcraft UAV to a safe landing spot up to 1.5 times faster and with double success rate than the state of the art (including a commercially available solution), paving the way towards realistically deployable UAVs.

Published in:
IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2022
DOI: 10.1109/IROS47612.2022.9981152

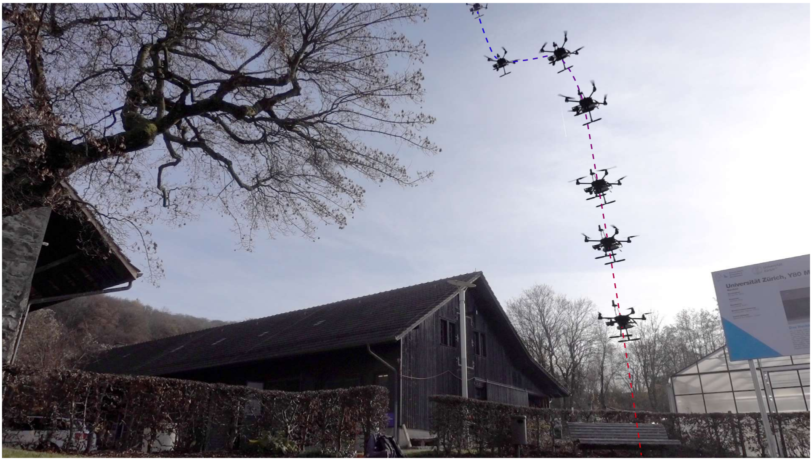


Figure 8.1: Safe emergency landing performed during a real-world experiment. Using semantic and depth information, the multi-rotor UAV finds a safe landing area and autonomously descends towards it, while avoiding collisions with the environment.

1 Introduction

Multicopters are highly agile and versatile Unmanned Aerial Vehicles (UAVs), which can be utilized in a variety of applications, such as search-and-rescue missions, inspection and 3D reconstruction, surveying and goods delivery. However, several dangerous events can occur during deployment. Adverse atmospheric conditions, such as cold weather or strong wind gusts, can unexpectedly shorten the battery life, whereas mechanical malfunctions and localization failures can potentially lead to crashes. Commercial drones, since they are designed to fly over populated areas, remain capable of landing in such critical situations, even with reduced battery voltage or partially working rotors. It is, therefore, fundamental to enable UAVs to detect safe landing spots nearby and autonomously land there in case of an emergency. The problem of detecting a safe landing area assumes greater importance in the case of navigation in urban settings, as damage needs to be avoided not only to the body of the drone and the landscape, but also to moving cars, people, etc. Moreover, given the limited payload capabilities of multirotor drones, such UAVs are generally equipped with a minimal sensor set-up composed of cameras and range sensors, rendering the emergency landing problem even more challenging, as a suitable landing location has to be identified by relying on limited sensory information. Recent solutions proposed in the literature either require manual intervention at the landing area by placing fiducial markers to identify a safe spot on the ground [112, 118], or require dense online 3D mapping of the environment [46, 90]. While the former methods are not suitable for emergency landing in unstructured environments, the latter ones, despite being more flexible, require high-quality sensory inputs. This assumption limits the applicability of these approaches, as their sensing pipeline cannot cope with the high level of noise during outdoor real-world flights, leading to failures. Moreover, these methods reason only about the geometry of the navigation area dismissing more general and relevant context, for example, the fact that some areas can be identified as clear spaces, but are unsuitable for landing, such as rooftops and roads. In order to overcome this issue, pipelines use high-level semantic information, which has been receiving growing attention in recent years [137], as semantic segmentation has been proven to be extremely useful for path-planning applications [12]. However, these approaches have been deployed in simulation only, raising questions about their applicability in real-world missions.

Motivated by these challenges, we propose a deep Reinforcement Learning-based (RL) emergency landing pipeline that detects safe landing areas from both depth and semantic information, by relying on a minimal sensor set-up composed of a monocular RGB camera. Thanks to recent findings in depth estimation and semantic segmentation using deep learning, a single camera snapshot is sufficient to recover the necessary information to land a UAV safely. The proposed RL agent, taking as input depth and semantic images, outputs high-level commands to find a suitable landing spot and navigates the UAV towards it as fast as possible, while avoiding collisions (Fig. 8.1). We demonstrate that we can train the proposed policy only in simulation and directly deploy it in real-world experiments, where other state-of-the-art planners fail. This work represents a fundamental step towards more secure and reliable autonomous drones, able to detect safe areas and land in case of an emergency respecting safety constraints.

In brief, the contributions of this work are the following:

- the design of a semantic-aware policy architecture that can be trained in simulation only and then deployed on a real UAV,
- an extensive evaluation of the proposed system in both simulation and real-world experiments,
- the design of a photo-realistic simulator that takes into account the drone's dynamics, and
- the source code of the proposed system upon acceptance of this work.

2 Related Works

Despite the increasing importance given to safety, emergency landing of drones still remains an open problem, especially in unknown environments [58, 158]. Detecting a safe landing area and approaching it are both challenging tasks, since factors such as obstacles, uncertainties in state estimation, and the platform's dynamics must be accounted for. From a control theory perspective, this problem is cast as an optimization, where the optimal motor commands are found with respect to a final objective, e.g. minimizing fuel usage [109]. A plethora of approaches have been proposed, ranging from optimization-based nonlinear control strategies [146] to more complex solutions, built upon optimal policy search using deep reinforcement learning [109]. These solutions are demonstrated to be extremely effective in controlling a range of different platforms, such as quadrotors [146] and spacecrafts [109], even under noisy state estimation by using learning-based strategies to model the robot's dynamics [127]. However, as the focus is solely on the control problem, they assume that a landing target is given and do not deal with obstacle avoidance.

In the literature, as well as in commercially available solutions such as Google Wing and Amazon Prime Air, a common approach is to manually mark suitable landing spots using highly-distinctive fiducial markers. These are also employed to enable small quadcopters to land on top of other moving platforms, such as ground robots [118], or on decks of maritime vehicles [113]. In these cases, the presence of the fiducial marker is fundamental, as its simple tracking facilitates the task greatly. In this direction, the recent work by Polvara *et al.* [112] proposes a reinforcement learning-based system, using domain randomization to enable a quadrotor to land on top of a fiducial marker under perceptual aliasing caused by different background textures, including floor tiles, grass, terrain and concrete. While they demonstrate it is possible to reach a human-like level of performance in real-world experiments by relying on RGB images only, their approach cannot be utilized for emergency landing in unknown environments, since it relies on the presence of a marker and does not deal with obstacles and occlusions.

To overcome this limitation, more sophisticated vision-based pipelines that do not rely on fiducial markers have been proposed, using either geometrical [46, 90, 158] or deep learning approaches [58, 145]. These solutions are more flexible and allow robots to land safely in unstructured and unknown environments [158]. To identify a suitable spot without maps known *a priori*, they employ RGB, depth and semantic images, and output the desired landing location [47, 58]. They are able to actively avoid collisions against obstacles by first generating a map of the area online, and then planning in it [158]. The work by Foster *et al.* [46] explicitly creates an elevation map using depth completion to identify a suitable landing spot, and then generates a path to that position in a separate path-planning module. Similarly, Mittal *et al.* [90] identify a safe landing area by processing dense depth images from a stereo camera and extracting terrain information, such as steepness and flatness, while also considering depth accuracy and the estimated energy consumption to reach a candidate location. Once a final position is identified, a safe path to the landing area is computed. While these approaches are validated in real-world experiments, they assume that the dense depth input, either from depth completion or stereo matching, is stable and accurate. However, this hypothesis might not be valid when flying at higher altitudes, as depth uncertainty increases with the squared value of depth [99], and stereo matching can provide accurate depth information only in a short range.

Another line of research utilizes Neural Networks (NNs) for binary classification of the terrain into either safe or hazardous areas [58], while performing landing site detection and tracking of people [137]. The training is performed on either satellite images or synthetic datasets [115], with very few works tested in real-world experiments [47]. Furthermore, this category of approaches performs binary classification of the landing area, not exploiting the complete semantic knowledge extracted from sensor readings. Using the full semantic mask can be beneficial, as it could allow higher level reasoning on the structure of the environment to identify safer landing areas faster.

Inspired by the shortcomings of these approaches and the need of emergency landing strategies for realistically deployable UAVs, in this work we propose a deep RL approach that employs semantically labeled images and depth maps to perform emergency landing of UAVs in urban scenarios. This is performed by leveraging the most recent results in deep learning for depth completion [139] and semantic segmentation [147]. Using these mid-level representa-

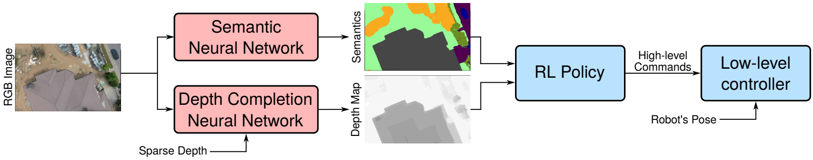


Figure 8.2: Schematic overview of the pipeline. The input to the system is a monocular RGB image, which is used by a semantic segmentation NN [147] to generate a semantic mask of the environment. A depth completion network [139] outputs a dense depth map of the surroundings using the RGB image and sparse depth information. The semantic mask and the depth map are used by the RL agent to generate high-level commands, that are successively fed to the robot's low-level controller.

tions as inputs can alleviate the simulation-to-reality gap, allowing for faster and more stable training of deep RL agents [12, 27]. We demonstrate that our policy, trained exclusively in simulation, can be directly deployed onboard a UAV in real-world missions. To run onboard drones with limited computing capabilities, the proposed pipeline is designed in a modular fashion, that allows to outsource depth estimation and semantic segmentation to cloud-based resources, if needed. We compare our approach against both a state-of-the-art landing pipeline [90] and a commercially available solution, demonstrating that our approach reaches higher success rates, as well as faster landings.

3 Methodology

Our main objective is to identify a safe landing area and perform the descending maneuvers safely and as fast as possible to reduce the flight time. We formulate this as a deep reinforcement learning problem, where an RL agent is trained to identify suitable areas by means of depth maps and semantically labeled images. While depth maps carry information about the shape of the environment (e.g. obstacles, free space), semantics expose the agent to the spatial relationships between different classes (e.g. cars and roads; houses and grass) [12], speeding up the search for valid landing spots. Moreover, using these mid-level representations as input to a deep RL agent is proven to yield better generalization of the policy [27]. Learning directly from raw camera data to command actions requires an implicit semantic segmentation and depth estimation step, which would take an extremely long training time in an RL fashion. Our architecture allows to decouple the problems of depth estimation and semantic labeling from path planning and control. This decoupling is essential for portability and deployability, as by simply providing access to cloud-based computing resources, we can enable any flying platform to employ our learnt policy. Given the recent findings in semantic segmentation [147] and depth completion [139] using deep learning, this work assumes that semantically labeled images and depth maps are available. These inputs are fed to the agent, which outputs high-level commands that are then processed by a low-level controller. The RL agent is trained in photorealistic simulations only, initially by using ground truth semantics and depth, and then

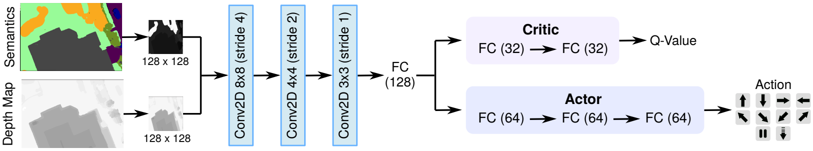


Figure 8.3: Schematic overview of the policy architecture. First, the semantic image, converted into a single channel, and the depth map are downsampled. The resulting images go through the Actor-Critic network, composed of three CNN layers and a linear Fully Connected (FC) layer. The Critic then outputs the Q-Value from two FC layers, while the Actor feeds the high-level command to the controller. Policy optimization is performed with the PPO algorithm [124].

fine-tuned by replacing these high-quality inputs with predictions from NNs. This procedure fills the simulation-to-reality gap and leads to policy generalization, allowing us to test the agent in real-world experiments without additional fine tuning.

3.1 System Overview

An overview of the proposed pipeline is shown in Fig. 8.2, where we assume a stream of RGB images as input. Thanks to the level of maturity of state-of-the-art semantic segmentation and depth completion using deep learning, a monocular camera snapshot and sparse depth seeds are sufficient to extrapolate the semantic masks and the depth maps. The RL agent is then tasked to find a suitable landing spot and descend towards it by generating high-level commands that are sent to the robot's low-level controller. While depth and stereo cameras can provide information in the range of a few meters, depth completion methods can estimate depth with a longer range, albeit at the expense of higher noise levels. Nevertheless, long range information is beneficial in drone landing, enabling a platform to be proactive with respect to collisions. The modular design of the proposed pipeline decouples the RL problem from semantic segmentation and depth estimation, allowing our system to run onboard a small UAV.

3.2 Deep RL Policy

The RL agent maps semantically labeled images and depth information to actions employing an Actor-Critic model with the architecture shown in Fig. 8.3. Here an action consists of a high-level command (lateral movement, halt motion, descending motion) that can be selected from a set of 10 possible actions. In the following, we describe the policy architecture and the reward function used to train the agent in more detail.

Policy Architecture

The Actor and the Critic networks share the first part, composed of a 3-layer Convolutional Neural Network (CNN). The final part of the Critic consists of two Fully Connected (FC) layers composed of 32 units, while the action is output by the Actor from three FC layers with 64 units each. Policy optimization is then performed at fixed-step intervals employing the on-policy algorithm PPO [124]. Moreover, in order to reduce the hyperspace dimension, we convert the color semantic mask into a single channel and downsample both the resulting image and the depth map to 128 × 128 pixels. Semantics and depth are mid-level visual representations that are more generic than raw color images, and they are demonstrated to allow faster training and improved policy performances [27].

Reward Function

The training of the policy is performed based on the data and the rewards collected in each episode. At each timestep, the total reward is composed of different terms:

$$R_{TOT} := \omega_t R_t + \omega_S R_S + \omega_C R_C + \omega_A R_A + R_T, \tag{8.1}$$

where R_t is the step reward, R_S is associated to the semantic classes in view, R_C to collisions between the robot and the environment, R_A to the selected action, and R_T is the terminal reward.

The step reward is assigned at constant intervals until the episode ends:

$$R_t := \begin{cases} 0 & \text{on episode termination} \\ -0.2 & \text{otherwise} \end{cases} . \tag{8.2}$$

The negative contribution to the total reward encourages the policy to land as fast as possible.

The reward associated to semantics, R_S, incites the agent to fly on top of the classes that are considered safe for landing, such as terrain, while penalizing flights on dangerous areas, including roads and houses. Given the semantic mask I_S and the set of possible semantic classes $\mathcal{S}$, this reward is calculated as

$$R_S := \sum_{s \in \mathcal{S}} \gamma_s \frac{count(I_S^s)}{count(I_S)} , \tag{8.3}$$

where $count(I_S^s)$ the number of pixels in I_S associated to class s, $\gamma_s \in \{-1, 1\}$ is the weight associated to s, and $count(I_S)$ the total number of pixels in I_S. A negative weight γ_s represents a penalty associated to flying over class $s \in \mathcal{S}$ and leads to more exploratory actions, while a positive value encourages the agent to start landing on top of s.

While the objective of this reward is to identify a suitable landing spot, the collision reward R_C deals with the problem of obstacle avoidance. In this case, we model the UAV as a sphere with radius r and this reward pushes the agent away from obstacles:

$$R_C := \begin{cases} 0 & \text{if } d_{min} \geq d^* \\ -\frac{1}{d_{min} - r} & \text{if } r < d_{min} < d^* \end{cases} , \tag{8.4}$$

where d_{min} is the distance to the closest obstacle and d^* is a threshold to check if the UAV is too close to an object, with $d^* > r$. When the robot is closer than d^* to an obstacle, the policy is rewarded negatively and, in case of a collision ($d_{min} \leq r$), the episode ends and the agent receives a negative terminal reward R_T.

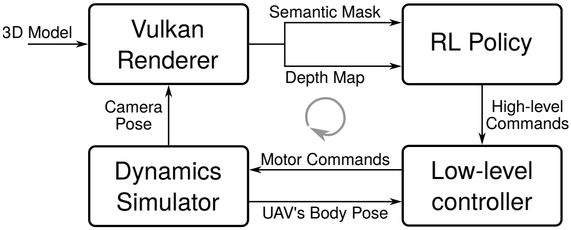


Figure 8.4: Schematic overview of the simulation used for training the deep RL policy. Given a 3D model and a camera pose, a Vulkan-based renderer generates semantic and depth images that are fed to the agent. The policy outputs high-level commands that are transformed into motor commands by a low-level controller, and the UAV position is updated according to the dynamic model for quadrotors introduced in [132]. At test time, the ground-truth semantic and depth images are substituted by the output of the semantic segmentation and depth-completion NNs from [147] and [139], respectively.

The reward R_A is associated to the action selected by the policy. As our pipeline is built for emergency landing and our objective is to reach the ground quickly, we reward the descending movements, while slightly penalizing lateral moves and halting the motion:

$$R_A := \begin{cases} -0.05 & \text{if lateral movement} \\ -0.5 & \text{if halt motion} \\ 1 & \text{if descending movement} \end{cases} . \tag{8.5}$$

At the end of each episode, we assign the terminal reward R_T. The episode terminates when the robot either lands safely, collides against an obstacle or when the maximum allowed time to land is reached. In case of a successful run, for example when landing on a safe area, such as flat terrain or grass, we assign a positive terminal reward, while in case of collisions, exhausted time budget or landing on a hazardous area, the policy receives a penalty as follows:

$$R_T := \begin{cases} 50 & \text{if landing on safe areas} \\ -20 & \text{if landing on dangerous areas} \\ -20 & \text{if in collision} \\ -50 & \text{if maximum time reached} \end{cases} . \tag{8.6}$$

3.3 Training Environment

Fig. 8.4 shows the pipeline used to train the deep RL policy. We assume that the robot is equipped with a downward-looking camera, and we render photorealistic RGB images, semantic masks and depth maps using a custom pipeline based on the Vulkan library[1]. The Vulkan

---

[1] `https://www.vulkan.org`

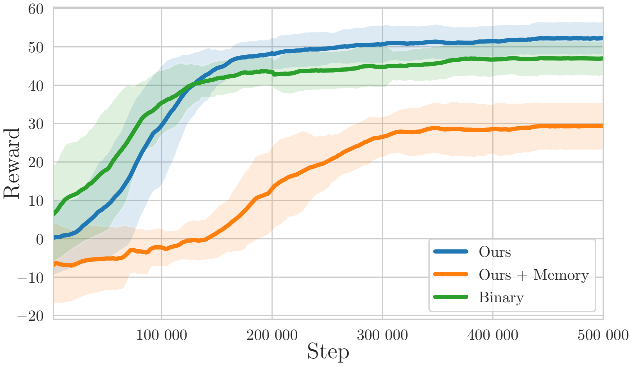


Figure 8.5: Reward curve over the training steps. The proposed policy is depicted in blue. In the initial phase of training, the sharp increase in the reward demonstrates that the RL agent quickly learns the actions to land on safe areas. In green, we show the training performance of the policy using binary images as input, and in orange our policy with an additional memory module after the fully connected layer.

API allows to create a fast and lightweight rendering scheme that, given a camera pose and a 3D model with associated textures, generates semantics and depth information that is then fed to the RL policy. The communication between the renderer and the policy is established with TCP sockets, in order to mimic the pipeline used in real-world experiments, where semantic segmentation and depth completion are outsourced to web computing resources. The high-level actions from the policy are then fed to the low-level controller that, given the current robot's pose, generates motor commands. By adopting the dynamic UAV model proposed in the Flightmare framework [132] and assuming a fixed integration step, the quadrotor flies to the target location. This cycle repeats until the end of the episode.

Notice that, in order to have a stable training process, ground truth semantics and depth images are fed to the agent, since the outputs from the semantic segmentation and depth-completion NNs are subjected to a high level of noise. The training performance of a policy when exposed to stochastic inputs can degrade drastically, and it is a well-known problem in RL. Only after the initial training phase, we fine-tune our policy by exposing the agent to the complete pipeline. In Sections 4.1 and 4.2, the pipeline used for fine-tuning and testing is described in more detail.

4 Experiments

4.1 Policy Training

We select a series of photorealistic models generated from photogrammetry, where the policy is trained (Fig. 8.6). The UAV is equipped with a downward-looking camera, and at each episode

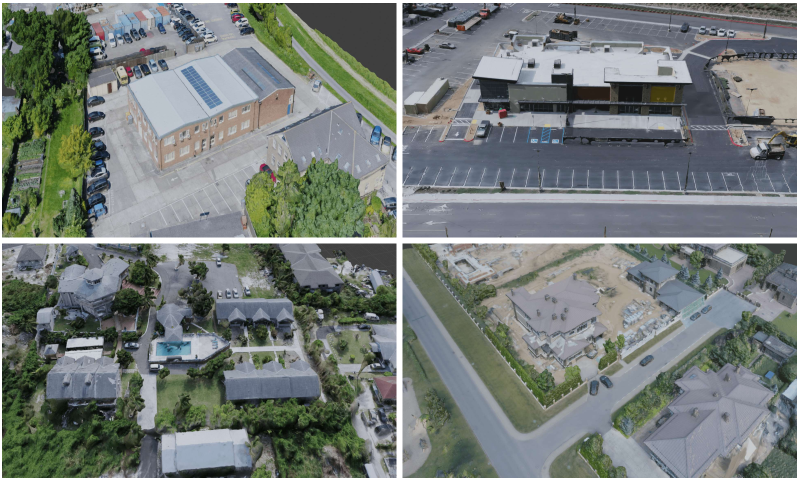


Figure 8.6: Views of some of the 3D models experienced by the RL agent at training time. The models, extracted from urban scenes only, are characterized by the same semantic classes (buildings, roads and pavements, terrain, cars and vegetation).

the robot is placed at a random initial position and it is tasked to land safely on the ground. All the scenes are characterized by the same semantic classes (pavement/road, terrain, water, trees, buildings and cars). We use common classes used by several state-of-the-art semantic-segmentation algorithms, but our system can handle any set of classes.

To have a faster and more stable training, the process is parallelized by utilizing four drones at a time. Each agent outputs actions at fixed time intervals (or *steps*), communicates them to the robot's controller and collects the reward as feedback. The 3D model of the environment changes at constant time intervals, allowing the policy to experience different scenes and avoiding overfitting. In the first episode of the training process, the policy is initialized randomly. The training continues until the maximum number of steps across all episodes is reached. To compute the total reward as in (8.1), the weights are set to $\omega_t = 1$, $\omega_S = 1$, $\omega_C = 0.5$ and $\omega_A = 1$, while d^* and r are set to $3\,\mathrm{m}$ and $0.5\,\mathrm{m}$, respectively.

Fig. 8.5 reports the training performance of the agent over 500 000 steps (blue curve), with a linearly decaying learning rate. The policy is initially trained with ground truth inputs and shows a converging reward curve. The agent is successively fine-tuned by exposing it to noisy inputs coming from semantic segmentation [147] and depth completion [139] NNs, where the sparse depth seeds are obtained by sampling ground truth depth images. This is a necessary step to deal with the simulation-to-reality gap, which is of extreme importance in robotic applications. While mid-level representations can help to mitigate this problem, most of the works in the literature feed only ground truth semantic masks and depth maps corrupted by mild noise. In reality, the outputs from deep learning-based methods can introduce severe noise and inconsistencies in the pipeline, especially when NNs are exposed to inputs that differ from the training data. These inconsistencies can be potentially harmful for the policy, as RL is notoriously sensitive to noise on the inputs and on the reward. By fine-tuning the policy for 100 000 additional steps with a fixed learning rate of 10^{-4}, the policy is adjusted to improve its performance when exposed to noisy data.

4.2 Tests in Previously Unseen Environments

A series of experiments in challenging scenes are conducted to show the effectiveness of the proposed method. We select 7 models of real-life places not experienced at training time, all created from photogrammetry (Fig. 8.7). The test models are chosen to create hard challenges for the pipeline (e.g. number of obstacles, small valid landing areas) and present the same semantic classes experienced at training time.

Test Set-Up

We select 15 initial positions in the scenes and we report the number of successful landings, as well as the time to land. We consider a run to be successful if the drone lands without collisions on terrain and if the landing does not exceed a maximum allowed time, set to 30 s. Timings provide insights about the efficiency of the planning solution, as in emergency situations the objective is to reach the ground as fast as possible, avoiding large detours to find a suitable landing spot. At test time, we fix the weights of the NN modelling the policy and we replace the ground truth inputs with the outputs from the semantic network [147] and the depth completion method [139]. The depth completion network receives sparse depth seeds by sampling ground truth depth maps available from the simulator.

We compare our approach against a naive baseline strategy, the planner proposed by Mittal *et al.* [90] and the commercially available solution by PX4[2]. While the naive solution lands the UAV by guiding it straight towards the ground, [90] detects a safe landing spot processing the depth images to find a flat area with the best characteristics in terms of inclination, estimated depth uncertainty and expected energy consumption to reach that position. Then, RRT* is used to find a suitable path from the current UAV location to the target checking for collisions in a 3D map.

Similarly, the commercial solution by PX4 detects suitable landing spots using depth information. To decide whether a location is a good candidate landing spot, normal vectors in the area are estimated and the terrain gets classified as either safe or dangerous. This planner then triggers the UAV's descent until an obstacle is sensed in its path. If the distance to the obstacle is smaller than a threshold, the UAV flies in an exploratory pattern until a suitable landing spot is found, and then descends towards it. This cycle repeats until the ground is reached.

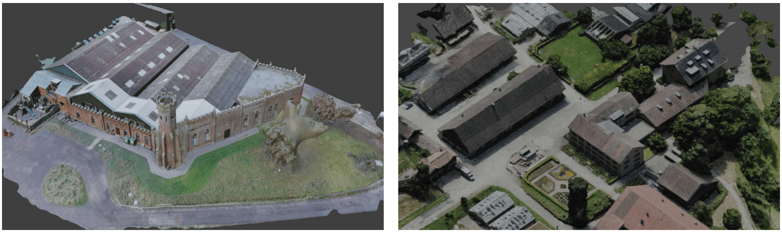
Both path planners by [90] and PX4 assume accurate depth measurements from an onboard stereo camera. However, the decreasing stereo disparity with increasing flight altitude renders this assumption unrealistic in practice. Moreover, following the trend in Robotic Vision literature, as in this work, replacing stereo depth estimation with deep learning-based depth comple-

---

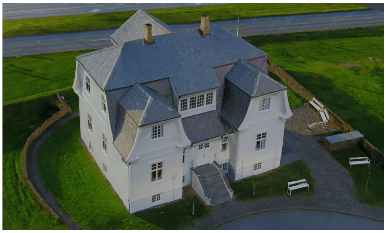
[2] `https://github.com/PX4/PX4-Avoidance`

(a) *Essex Castle*. **(b)** *Fraser Gunnery Range*.

(c) *Hartley Mansion*. **(d)** *Tin Hau Temple*.



(e) *Warehouse*. **(f)** *Irchel*.



(g) *Hofdi House*.

Figure 8.7: Views of the 3D models where the deep RL policy is tested. These models are not experienced by the agent at training time.

Experiment	Baseline	Ours	Mittal *et al.* [90]	PX4
Essex Castle				
Success Rate [%]	13.3	**73.3**	53.3	53.3
Average Time [s]	7.9 ± 2.1	**9.7 ± 2.3**	12.2 ± 3.9	13.8 ± 4.4
Fraser Gunnery Range				
Success Rate [%]	6.7	**86.7**	40.0	20.0
Average Time [s]	10.8 ± 0.0	**16.4 ± 5.9**	17.0 ± 4.1	19.5 ± 2.5
Hartley Mansion				
Success Rate [%]	6.7	**80.0**	60.0	53.3
Average Time [s]	11.5 ± 0.0	**10.5 ± 2.1**	15.9 ± 2.7	18.5 ± 1.9
Tin Hau Temple				
Success Rate [%]	6.7	**86.7**	26.7	13.3
Average Time [s]	10.0 ± 0.0	**10.6 ± 0.7**	17.8 ± 7.1	15.9 ± 0.9
Warehouse				
Success Rate [%]	20.0	**93.3**	26.7	86.7
Average Time [s]	11.2 ± 1.0	14.3 ± 3.1	**14.2 ± 2.4**	19.9 ± 1.8
Irchel				
Success Rate [%]	0.0	**93.3**	73.3	40.0
Average Time [s]	N.A.	**12.9 ± 1.3**	18.2 ± 3.9	22.4 ± 2.5
Hofdi House				
Success Rate [%]	0.0	**100.0**	60.0	80.0
Average Time [s]	N.A.	**8.1 ± 2.3**	13.8 ± 4.2	17.1 ± 5.3

Table 8.1: Results of the experiments in the 7 simulated scenes. We report the success rate over 15 runs, as well as the average time to land. The averages of the timings are computed considering only the successful landings. The best performance in each scene is shown in bold, without considering the results of the baseline planner.

tion as a workaround, can also result in highly noisy depth estimates, limiting the applicability of these planners. Note that, while neither of these planners consider semantic information in their original forms, for fairness of comparison with our pipeline, we extend them to use semantic masks alongside depth maps, performing an additional binary classification of the landing spots on top of their original pipelines, by considering the class ‘terrain’ as the only safe area.

Results and Discussion

The success rates of the test runs, as well as the average time to land, are reported in Table 8.1. The latter is computed by considering only the successful runs. Notice that in the first column we show the performance of the ’Baseline’, which refers to a naive planner guiding the UAV to

Experiment	Ours	Ours + Memory	Binary
Essex Castle	**73.3**	26.7	60.0
Fraser Gunnery Range	**86.7**	6.7	33.3
Hartley Mansion	**80.0**	0.0	46.7
Tin Hau Temple	**86.7**	6.7	46.7
Warehouse	**93.3**	13.3	80.0
Irchel	**93.3**	0.0	66.7
Hofdi House	**100.0**	0.0	66.7

Table 8.2: Success rates (in %) on the test models for the ablation study comparing our architecture with a more complex policy with a memory module (*Ours + Memory*) and against a policy using depth maps and binary semantic masks as inputs (*Binary*). The best results are shown in bold.

land immediately descending straight to the ground. This is the fastest approach, but exhibits the lowest success rates as it does not consider the UAV's surroundings. The proposed policy exhibits the highest success rates and the best timings across almost all cases, with the agent respecting both the safety and the timing constraints, while landing the UAV on valid locations and avoiding collisions. Thanks to the use of full semantic masks, the RL agent learns to leverage the spatial relationships between the semantic classes during training, and thus the policy identifies safe landing spots quickly and effectively, avoiding extensive exploration of the surroundings. On the contrary, the effectiveness of the other planners to land the UAV safely and without crashes seems to be limited due to the noisy input, often guiding the UAV to fly over large detours. In particular, terrain information, such as its flatness or inclination that are extracted from the depth maps generated using depth completion are noisy, rendering the identification of valid landing spots in these methods difficult. In fact, PX4 collides against obstacles, while [90] generates long, intertwined paths.

4.3 Ablation Study

To further analyze our pipeline, we run ablation studies on the policy architecture and type of input to the agent. We compare our approach against two variants; the *Ours + Memory* policy with an additional Long Short-Term Memory (LSTM) component with 256 cells before the Actor-Critic module (similarly to [12]), and a *Binary* policy with the same architecture as in Fig. 8.3, but trained with depth images and binary semantic masks, where the only valid landing area is 'terrain'. The training performances of all variants are shown in Fig. 8.5, and the success rates in the test models are reported in Table 8.2. Extending the policy with the memory module exhibits the worst performances at both training and testing time, with no successful runs in 3 out of 7 test models. While LSTMs could be beneficial in dynamic scenes [12], they require more time to train, leading to lower rewards. Instead, the *Binary* policy converges faster, but to a lower average reward than our policy. This is reflected in the performances during testing, since our method achieves the highest success rates, demonstrating the benefit of using the full

Image Type	Latency [s]	Bandwidth [MB/s]
Depth	3.03 ± 1.21	3.18 ± 0.13
Semantics	1.34 ± 0.47	0.27 ± 0.03

Table 8.3: Latency and bandwidth consumption statistics for the communication between the cloud and the UAV's onboard computer in a real-world experiment.

semantic masks over binary inputs.

4.4 Real-world Experiments

We test the proposed pipeline in a series of real-world experiments, where semantic segmentation and the depth-completion NN are outsourced to cloud-based computing resources (Amazon AWS). We directly deploy the policy trained exclusively in simulation without any additional fine tuning, demonstrating that the use of depth and semantic images as input to the RL agent can fill the simulation-to-reality gap.

System Set-up

The UAV used in the experiments is a *Holybro X500*, equipped with an Intel NUC with an Intel Core i5-1145G7 CPU and a 4G modem stick to communicate with the cloud resources. We use the *Pixhawk* autopilot[3] as a low-level controller, while state estimation is performed by fusing RTK GPS estimates with inertial measurements. RGB images are captured with a down-looking global-shutter camera and are successively sent to the cloud via TCP connections to extract depth and semantic information. Alongside color images, the depth-completion NN receives sparse depth seeds in the form of 3D landmarks estimated by ORB-SLAM3 [24] running onboard the drone. Once semantic and depth information is received from the cloud, it gets fed into the RL agent, which communicates high-level commands to the UAV's controller.

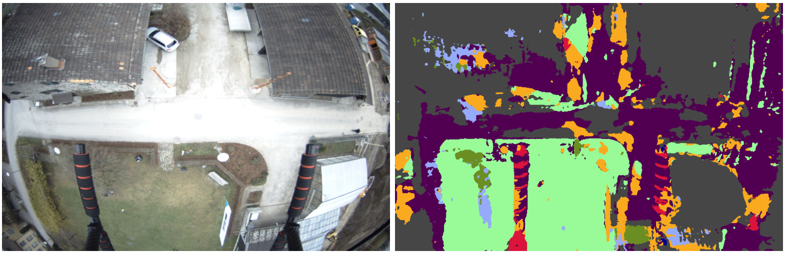
The experiments are carried out in an urban area. The UAV starts at an altitude of $25\,\mathrm{m}$ off the ground over a street, and it is tasked to land on grass, avoiding collisions with the nearby buildings and vegetation. We consider a run to be successful if the drone lands without interventions from the safety pilot. In Table 8.3 we report the latency from sending RGB images to the cloud and receiving depth and semantic information back, as well as the bandwidth consumption for communication. Once the policy receives these inputs, it takes an additional $0.40 \pm 0.07\,\mathrm{ms}$ to output the high-level control commands.

Results and Discussion

Our policy, even if it was trained only in simulation, is the only approach able to land the UAV safely. In contrast, the planners by [90] and PX4 cannot cope with the noisy inputs from real-world data (Fig. 8.8). In particular, due to high noise in the depth, the former fails to find a

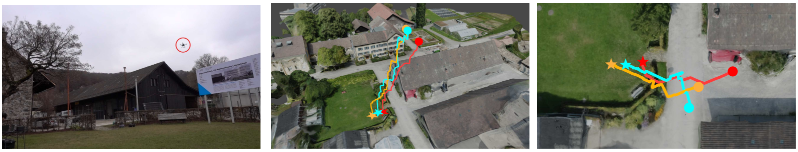
---

[3] `https://pixhawk.org`



(a) Visual Image. **(b)** Semantic Mask.

Figure 8.8: One example image and its corresponding semantic mask during a real-world experiment captured by the downward-looking camera. The light green areas correspond to grass and terrain, and are the only valid landing areas.



(a) External Camera Frame. **(b)** 3D View. **(c)** Top View.

Figure 8.9: View of the environment and examples of trajectories flown by the UAV during successful real-world experiments when guided by our policy. In (a) the drone is highlighted with a red circle. The paths are shown in the 3D reconstruction of the real place where experiments are carried out. The initial position is shown as a colored blob, while the landing spot as a star.

suitable landing position, guiding the UAV to hover over the initial position. Even when a suitable landing spot is found, the path generated is extremely convoluted because of the presence of artifacts in the 3D map used for obstacle avoidance. Similarly, PX4 starts the landing procedure, but, as soon as an obstacle is sensed, it flies the UAV in dangerous detours that force the safety pilot to take over to avoid crashes into the nearby buildings. As in simulation, also in the real experiments, the power of semantics to identify safe landing spots as advocated by the pipeline, is emphasized. Moreover, our policy does not build a full 3D reconstruction of the environment, but instead reacts to potential collisions in a receding horizon fashion, allowing the robot to safely land on grass (Fig. 8.9).

During testing, we experience some failure cases also with our policy. In one run, semantic misclassification causes the UAV to fly towards bushes, wrongly labeled as grass, while in another case, the UAV almost crashes into a lamppost as this is not present as an obstacle in the depth map. While the object is correctly identified in the semantic image, ORB-SLAM3 does not detect any 3D landmarks on the lamppost because of its featureless visual appearance.

Consequently, no sparse seeds are generated and the obstacle is missing in the depth image. Then the policy tries to land unaware of the presence of the obstacle. These experiments show that our pipeline can deal with noisy inputs, but it is still prone to failures in case of systematic errors, especially in semantics.

5 Conclusion and Future Work

In this work, we present a pipeline based on deep RL for autonomous landing of multicopter UAVs in case of an emergency. Our policy maps semantic and depth information onto actions, flying the UAV towards safe areas. We evaluate our system in a series of challenging experiments, both in simulation and reality, demonstrating that we can reach higher success rates and faster landings compared to the state of the art, including a commercially available solution. Moreover, we show that the use of multi-class semantic segmentation is beneficial compared to binary approaches, as it allows the UAV to acquire more context, learning the spatial relationships between the different classes of the scene. In our experiments, using the full semantic knowledge leads to success rates in average 150% higher than cases when binary segmentation is employed. More importantly, our policy transfers directly from simulation to reality, and we demonstrate it can safely land a drone in real-world experiments even when exposed to inputs corrupted by noise.

As future work, we propose to extend the pipeline to consider the uncertainty in semantic classification and in depth prediction to increase robustness against noise, and to further develop the pipeline to work in dynamic scenes.

Bibliography

[1] M. Achtelik, S. Lynen, S. Weiss, M. Chli, and R. Siegwart. Motion- and Uncertainty-aware Path Planning for Micro Aerial Vehicles. *Journal of Field Robotics (JFR)*, pages 676–698, 2014.

[2] A.-A. Agha-Mohammadi, S. Chakravorty, and N. M. Amato. FIRM: Sampling-based Feedback Motion-planning under Motion Uncertainty and Imperfect Measurements. *The International Journal of Robotics Research (IJRR)*, pages 268–304, 2014.

[3] A. Ahmad, V. Walter, P. Petráček, M. Petrlík, T. Báča, D. Žaitlík, and M. Saska. Autonomous Aerial Swarming in GNSS-denied Environments with High Obstacle Density. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[4] J. Aloimonos, I. Weiss, and A. Bandyopadhyay. Active Vision. *International Journal of Computer Vision (IJCV)*, pages 333–356, 1988.

[5] I. Alzugaray, L. Teixeira, and M. Chli. Short-term UAV path-planning with monocular-inertial SLAM in the loop. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[6] F. Amigoni, J. Banfi, and N. Basilico. Multirobot exploration of communication-restricted environments: A survey. *IEEE Intelligent Systems*, pages 48–57, 2017.

[7] N. A. Atanasov. *Active information acquisition with mobile robots*. PhD thesis, University of Pennsylvania, 2015.

[8] R. Bajcsy, Y. Aloimonos, and J. K. Tsotsos. Revisiting Active Perception. In *Autonomous Robots*, pages 177–196. Springer, 2018.

[9] L. Bartolomei, M. Karrer, and M. Chli. Multi-robot Coordination with Agent-Server Architecture for Autonomous Navigation in Partially Unknown Environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[10] L. Bartolomei, Y. Kompis, L. Teixeira, and M. Chli. Autonomous Emergency Landing for Multicopters using Deep Reinforcement Learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[11] L. Bartolomei, L. Teixeira, and M. Chli. Perception-aware Path Planning for UAVs using Semantic Segmentation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[12] L. Bartolomei, L. Teixeira, and M. Chli. Semantic-aware Active Perception for UAVs using Deep Reinforcement Learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[13] L. Bartolomei, L. Teixeira, and M. Chli. Towards Multi-robot Exploration: A Decentralized Strategy for UAV Forest Exploration. *arXiv preprint*, 2023.

[14] G. Best, R. Garg, J. Keller, G. A. Hollinger, and S. Scherer. Resilient Multi-Sensor Exploration of Multifarious Environments with a Team of Aerial Robots. In *Robotics: Science and Systems (RSS)*, 2022.

[15] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart. Receding Horizon

"Next-Best-View" Planner for 3D Exploration. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[16] H. Blum, S. Rohrbach, M. Popovic, L. Bartolomei, and R. Siegwart. Active Learning for UAV-based Semantic Mapping. *RSS - Workshop on Informative Path Planning and Adaptive Sampling*, 2019.

[17] P. O. Bressan, J. M. Junior, J. A. C. Martins, M. J. de Melo, D. N. Gonçalves, D. M. Freitas, A. P. M. Ramos, M. T. G. Furuya, L. P. Osco, J. de Andrade Silva, et al. Semantic segmentation with labeling uncertainty and class imbalance applied to vegetation mapping. *International Journal of Applied Earth Observation and Geoinformation*, 2022.

[18] A. Bry and N. Roy. Rapidly-exploring Random Belief Trees for Motion Planning under Uncertainty. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[19] S. J. Buckley. Fast motion planning for multiple moving robots. In *IEEE International Conference on Robotics and Automation (ICRA)*, 1989.

[20] W. Burgard, M. Moors, C. Stachniss, and F. Schneider. Coordinated multi-robot exploration. *IEEE Transactions on Robotics (T-RO)*, pages 376–386, 2005.

[21] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart. The EuRoC micro aerial vehicle datasets. *The International Journal of Robotics Research (IJRR)*, page 1157–1163, 2016.

[22] M. Burri, H. Oleynikova, M. W. Achtelik, and R. Siegwart. Real-time visual-inertial mapping, re-localization and planning onboard MAVs in unknown environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[23] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard. Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age. *IEEE Transactions on Robotics (T-RO)*, pages 1309–1332, 2016.

[24] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós. ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual–Inertial, and Multimap SLAM. *IEEE Transactions on Robotics (T-RO)*, pages 1874–1890, 2021.

[25] M. Chandarana, D. Hughes, M. Lewis, K. Sycara, and S. Scherer. Planning and Monitoring Multi-Job Type Swarm Search and Service Missions. *Journal of Intelligent & Robotic Systems (JINT)*, pages 1–14, 2021.

[26] B. Charrow, G. Kahn, S. Patil, S. Liu, K. Goldberg, P. Abbeel, N. Michael, and V. R. Kumar. Information-Theoretic Planning with Trajectory Optimization for Dense 3D Mapping. *Robotics: Science and Systems (RSS)*, 2015.

[27] B. Chen, A. Sax, F. Lewis, I. Armeni, S. Savarese, A. Zamir, J. Malik, and L. Pinto. Robust Policies via Mid-Level Visual Representations: An Experimental Study in Manipulation and Navigation. In *Conference on Robot Learning (CoRL)*, 2020.

[28] T. Cieslewski, E. Kaufmann, and D. Scaramuzza. Rapid exploration with multi-rotors: A frontier selection method for high speed flight. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[29] R. G. Colares and L. Chaimowicz. The next frontier: combining information gain and distance cost for decentralized multi-robot exploration. *Annual ACM Symposium on Applied Computing (SAC)*, 2016.

[30] C. Connolly. The determination of next best views. In *IEEE International Conference on Robotics and Automation (ICRA)*, 1985.

[31] M. Corah and N. Michael. Distributed Matroid-Constrained Submodular Maximization for Multi-Robot Exploration: Theory and Practice. *Autonomous Robots*, page 485–501, 2019.

[32] M. Corah, C. O'Meadhra, K. Goel, and N. Michael. Communication-Efficient Planning and Mapping for Multi-Robot Exploration in Large Environments. *IEEE Robotics and Automation Letters (RA-L)*, pages 1715–1721, 2019.

[33] G. Costante, J. Delmerico, M. Werlberger, P. Valigi, and D. Scaramuzza. Exploiting Photometric Information for Planning Under Uncertainty. In *Robotics Research*, pages 107–124. Springer, 2018.

[34] D. L. da Silva Lubanco, M. Pichler-Scheder, and T. Schlechter. A Novel Frontier-Based Exploration Algorithm for Mobile Robots. In *International Conference on Mechatronics and Robotics Engineering (ICMRE)*, 2020.

[35] T. Dang, S. Khattak, F. Mascarich, and K. Alexis. Explore locally, plan globally: A path planning framework for autonomous robotic exploration in subterranean environments. In *International Conference on Advanced Robotics (ICAR)*, 2019.

[36] A. J. Davison and D. W. Murray. Simultaneous Localization and Map-building using Active Vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*, pages 865–880, 2002.

[37] M. Dharmadhikari, T. Dang, L. Solanka, J. Loje, H. Nguyen, N. Khedekar, and K. Alexis. Motion Primitives-based Path Planning for Fast and Agile Exploration using Aerial Robots. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[38] M. Dharmadhikari, H. Nguyen, F. Mascarich, N. Khedekar, and K. Alexis. Autonomous cave exploration using aerial robots. In *International Conference on Unmanned Aircraft Systems (ICUAS)*, 2021.

[39] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel. Path Planning for Autonomous Vehicles in Unknown Semi-structured Environments. *The International Journal of Robotics Research (IJRR)*, pages 485–501, 2010.

[40] S. Dong, K. Xu, Q. Zhou, A. Tagliasacchi, S. Xin, M. Nießner, and B. Chen. Multi-Robot Collaborative Dense Scene Reconstruction. *ACM Transactions on Graphics (TOG)*, pages 1–16, 2019.

[41] A. Dutta, A. Bhattacharya, O. P. Kreidl, A. Ghosh, and P. Dasgupta. Multi-robot informative path planning in unknown environments through continuous region partitioning. *International Journal of Advanced Robotic Systems (IJARS)*, 2020.

[42] T. Elmokadem and A. Savkin. Towards fully autonomous UAVs: A survey. *Sensors*, page 6223, 2021.

[43] J. Faigl, M. Kulich, and L. Přeučil. Goal assignment using distance cost in multi-robot exploration. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[44] D. Falanga, P. Foehn, P. Lu, and D. Scaramuzza. PAMPC: Perception-aware model predictive control for quadrotors. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[45] P. Fankhauser, M. Hutter, C. Gehring, M. Bloesch, M. A. Hoepflinger, and R. Siegwart. Reinforcement learning of single legged locomotion. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[46] C. Forster, M. Faessler, F. Fontana, M. Werlberger, and D. Scaramuzza. Continuous on-board monocular-vision-based elevation mapping applied to autonomous landing of

micro aerial vehicles. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2015.

[47] P. Fraczek, A. Mora, and T. Kryjak. Embedded Vision System for Automated Drone Landing Site Detection. In *Computer Vision and Graphics*, pages 397–409. Springer, 2018.

[48] F. Furrer, M. Burri, M. Achtelik, and R. Siegwart. RotorS – A Modular Gazebo MAV Simulator Framework. In *Studies in Computational Intelligence*, pages 595–625, 2016.

[49] D. Gálvez-López and J. D. Tardós. Bags of Binary Words for Fast Place Recognition in Image Sequences. *IEEE Transactions on Robotics (T-RO)*, pages 1188–1197, 2012.

[50] S. Gangapurwala, M. Geisert, R. Orsolino, M. Fallon, and I. Havoutis. Rloc: Terrain-aware legged locomotion using reinforcement learning and optimal control. *IEEE Transactions on Robotics (T-RO)*, pages 2908–2927, 2022.

[51] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, P. Martinez-Gonzalez, and J. Garcia-Rodriguez. A survey on deep learning techniques for image and video semantic segmentation. *Applied Soft Computing*, pages 41–65, 2018.

[52] A. Gawel, H. Blum, J. Pankert, K. Krämer, L. Bartolomei, S. Ercan, F. Farshidian, M. Chli, F. Gramazio, R. Siegwart, M. Hutter, and T. Sandy. A Fully-Integrated Sensing and Control System for High-Accuracy Mobile Robotic Building Construction. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.

[53] GeekWire. Incoming! amazon patents a plan to turn its drone delivery packages into gliders. www.geekwire.com/2017/amazon-patent-drone-packages-gliders, 2017. Accessed: 2023-01-23.

[54] E. J. Griffith and S. Akella. Coordinating Multiple Droplets in Planar Array Digital Microfluidic Systems. *The International Journal of Robotics Research (IJRR)*, pages 933–949, 2005.

[55] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard. A tutorial on graph-based SLAM. *IEEE Intelligent Transportation Systems Magazine (ITSM)*, pages 31–43, 2010.

[56] S. Gu, S. Guo, W. Zuo, Y. Chen, R. Timofte, L. Van Gool, and L. Zhang. Learned Dynamic Guidance for Depth Image Reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*, pages 2437–2452, 2020.

[57] S. Gu, L. Yang, Y. Du, G. Chen, F. Walter, J. Wang, Y. Yang, and A. Knoll. A review of safe reinforcement learning: Methods, theory and applications. *arXiv preprint*, 2022.

[58] X. Guo, S. Denman, C. Fookes, and S. Sridharan. A robust UAV landing site detection system using mid-level discriminative patches. In *International Conference on Pattern Recognition (ICPR)*, 2016.

[59] Y. Guo, Y. Liu, T. Georgiou, and M. S. Lew. A review of semantic segmentation using deep neural networks. *International Journal of Multimedia Information Retrieval (IJMIR)*, pages 87–93, 2018.

[60] D. Hanover, A. Loquercio, L. Bauersfeld, A. Romero, R. Penicka, Y. Song, G. Cioffi, E. Kaufmann, and D. Scaramuzza. Autonomous Drone Racing: A Survey. *arXiv preprint*, 2023.

[61] G. Hardouin, J. Moras, F. Morbidi, J. Marzat, and E. M. Mouaddib. Next-Best-View planning for surface reconstruction of large-scale 3D environments with multiple UAVs. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[62] Z. Huang, J. Wu, and C. Lv. Efficient deep reinforcement learning with imitative expert priors for autonomous driving. *IEEE Transactions on Neural Networks and Learning*

Systems, 2022.

[63] J. Hwangbo, I. Sa, R. Siegwart, and M. Hutter. Control of a Quadrotor With Reinforcement Learning. *IEEE Robotics and Automation Letters (RA-L)*, pages 2096–2103, 2017.

[64] M. Hüppi, L. Bartolomei, R. Mascaro, and M. Chli. T-PRM: Temporal Probabilistic Roadmap for Path Planning in Dynamic Environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[65] V. Indelman, L. Carlone, and F. Dellaert. Towards Planning in Generalized Belief Space. In *Robotics Research: The International Symposium (ISRR)*, pages 593–609. Springer, 2016.

[66] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and Acting in Partially Observable Stochastic Domains. *Journal of Artificial Intelligence (JAI)*, pages 99–134, 1998.

[67] G. Kahn, A. Villaflor, B. Ding, P. Abbeel, and S. Levine. Self-Supervised Deep Reinforcement Learning with Generalized Computation Graphs for Robot Navigation. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[68] G. Kahn, A. Villaflor, V. Pong, P. Abbeel, and S. Levine. Uncertainty-Aware Reinforcement Learning for Collision Avoidance. *arXiv preprint*, 2017.

[69] M. Kamel, J. Alonso-Mora, R. Siegwart, and J. Nieto. Robust collision avoidance for multiple micro aerial vehicles using nonlinear model predictive control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[70] S. Karaman and E. Frazzoli. Sampling-based Algorithms for Optimal Motion Planning. *International Journal of Robotics Research (IJRR)*, pages 846–894, 2011.

[71] M. Karrer, P. Schmuck, and M. Chli. CVI-SLAM—Collaborative Visual-Inertial SLAM. *IEEE Robotics and Automation Letters (RA-L)*, pages 2762–2769, 2018.

[72] A. Khamis, A. Hussein, and A. Elmogy. Multi-robot task allocation: A review of the state-of-the-art. *Cooperative robots and sensor networks*, pages 31–51, 2015.

[73] D. Kim, G. C. Kim, Y. Jang, and H. J. Kim. Topology-Guided Path Planning for Reliable Visual Navigation of MAVs. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[74] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. Al Sallab, S. Yogamani, and P. Pérez. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems (T-ITS)*, 2021.

[75] T. Koch, M. Körner, and F. Fraundorfer. Automatic and semantically-aware 3D UAV flight planning for image-based 3D reconstruction. *Remote Sensing*, page 1550, 2019.

[76] Y. Kompis, L. Bartolomei, and M. Chli. Fully Autonomous Live 3D Reconstruction with an MAV: Hardware- and Software-Setup. In *International Conference on 3D Vision (3DV)*, 2021.

[77] Y. Kompis, L. Bartolomei, R. Mascaro, L. Teixeira, and M. Chli. Informed Sampling Exploration Path Planner for 3D Reconstruction of Large Scenes. *IEEE Robotics and Automation Letters (RA-L)*, pages 7893–7900, 2021.

[78] T. Kusnur, S. Mukherjee, D. M. Saxena, T. Fukami, T. Koyama, O. Salzman, and M. Likhachev. A planning framework for persistent, multi-UAV coverage with global deconfliction. In *Field and Service Robotics (FSR)*, pages 459–474. Springer, 2021.

[79] B. Liu. Lifelong machine learning: a paradigm for continuous learning. *Frontiers of Computer Science*, pages 359–361, 2017.

[80] F. Liu, J. Shan, B. Xiong, and Z. Fang. A Real-Time and Multi-Sensor-Based Landing

Area Recognition System for UAVs. *Drones*, 2022.

[81] X. Liu, W. Ye, C. Tian, Z. Cui, H. Bao, and G. Zhang. Coxgraph: Multi-Robot Collaborative, Globally Consistent, Online Dense Reconstruction System. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[82] S. Lynen, M. W. Achtelik, S. Weiss, M. Chli, and R. Siegwart. A Robust and Modular Multi-sensor Fusion Approach applied to MAV Navigation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[83] Y. Lyu, G. Vosselman, G.-S. Xia, A. Yilmaz, and M. Y. Yang. UAVid: A semantic segmentation dataset for UAV imagery. *ISPRS Journal of Photogrammetry and Remote Sensing (P&RS)*, pages 108–119, 2020.

[84] A. Mannucci, S. Nardi, and L. Pallottino. Autonomous 3D Exploration of Large Areas: A Cooperative Frontier-Based Approach. In *Modelling and Simulation for Autonomous Systems*, pages 18–39. Springer, 2018.

[85] F. Mascarich, P. De Petris, H. Nguyen, N. Khedekar, and K. Alexis. Autonomous distributed 3D radiation field estimation for nuclear environment characterization. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[86] D. Maturana, S. Arora, and S. Scherer. Looking forward: A semantic mapping system for scouting with micro-aerial vehicles. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[87] L. Meier, D. Honegger, and M. Pollefeys. PX4: A node-based multithreaded open source robotics framework for deeply embedded platforms. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2015.

[88] D. Mellinger and V. Kumar. Minimum snap trajectory generation and control for quadrotors. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[89] T. Miki, M. Popović, A. Gawel, G. Hitz, and R. Siegwart. Multi-agent Time-based Decision-making for the Search and Action Problem. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[90] M. Mittal, R. Mohan, W. Burgard, and A. Valada. Vision-Based Autonomous UAV Navigation and Landing for Urban Search and Rescue. In *Robotics Research*, pages 575–592. Springer, 2022.

[91] A. Mohammadi, Y. Feng, C. Zhang, S. Rawashdeh, and S. Baek. Vision-based autonomous landing using an MPC-controlled micro UAV on a moving platform. In *International Conference on Unmanned Aircraft Systems (ICUAS)*, 2020.

[92] D. Morilla-Cabello, L. Bartolomei, L. Teixeira, E. Montijano, and M. Chli. Sweep-Your-Map: Efficient Coverage Planning for Aerial Teams in Large-Scale Environments. *IEEE Robotics and Automation Letters (RA-L)*, pages 10810–10817, 2022.

[93] L. Mou, Y. Hua, and X. X. Zhu. A relation-augmented fully convolutional network for semantic segmentation in aerial scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[94] M. Mousaei, J. Geng, A. Keipour, D. Bai, and S. Scherer. Design, Modeling and Control for a Tilt-rotor VTOL UAV in the Presence of Actuator Failure. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[95] B. Mu, M. Giamou, L. Paull, A.-a. Agha-mohammadi, J. Leonard, and J. How. Information-based Active SLAM via topological feature graphs. In *IEEE Conference on Decision and Control (CDC)*, 2016.

[96] M. Mueller, A. Dosovitskiy, B. Ghanem, and V. Koltun. Driving Policy Transfer via

Modularity and Abstraction. In *Conference on Robot Learning (CoRL)*, 2018.

[97] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics (T-RO)*, pages 1147–1163, 2015.

[98] V. Murali, I. Spasojevic, W. Guerra, and S. Karaman. Perception-aware trajectory generation for aggressive quadrotor flight using differential flatness. In *American Control Conference (ACC)*, 2019.

[99] C. V. Nguyen, S. Izadi, and D. Lovell. Modeling Kinect Sensor Noise for Improved 3D Reconstruction and Tracking. In *International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission (3DIMPVT)*, 2012.

[100] J. Nikolic, J. Rehder, M. Burri, P. Gohl, S. Leutenegger, P. T. Furgale, and R. Siegwart. A synchronized visual-inertial sensor system with FPGA pre-processing for accurate real-time SLAM. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[101] H. Oleynikova, M. Burri, Z. Taylor, J. Nieto, R. Siegwart, and E. Galceran. Continuous-time trajectory optimization for online UAV replanning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.

[102] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto. Voxblox: Incremental 3D Euclidean Signed Distance Fields for On-Board MAV Planning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[103] Y. Pan, Y. Kompis, L. Bartolomei, R. Mascaro, C. Stachniss, and M. Chli. Voxfield: Non-Projective Signed Distance Fields for Online Planning and 3D Reconstruction. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[104] C. Papachristos, S. Khattak, and K. Alexis. Uncertainty-aware receding horizon exploration and mapping using aerial robots. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[105] L. E. Parker. Path planning and motion coordination in multiple mobile robot teams. *Encyclopedia of Complexity and System Science*, pages 5783–5800, 2009.

[106] M. Pavone, A. Arsie, E. Frazzoli, and F. Bullo. Distributed algorithms for environment partitioning in mobile robotic networks. *IEEE Transactions on Automatic Control*, pages 1834–1848, 2011.

[107] D. Paz, H. Zhang, Q. Li, H. Xiang, and H. I. Christensen. Probabilistic semantic mapping for urban autonomous driving applications. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[108] M. Peasgood, C. M. Clark, and J. McPhee. A Complete and Scalable Strategy for Coordinating Multiple Robots Within Roadmaps. *IEEE Transactions on Robotics (T-RO)*, pages 283–292, 2008.

[109] M. A. Pereira, C. A. Duarte, I. Exarchos, and E. A. Theodorou. Deep $\mathcal{L}^1$ Stochastic Optimal Control Policies for Planetary Soft-landing. *arXiv preprint*, 2021.

[110] J. Peters and S. Schaal. Reinforcement Learning by Reward-Weighted Regression for Operational Space Control. In *International Conference on Machine Learning (ICML)*, 2007.

[111] K. Pluckter and S. Scherer. Precision UAV landing in unstructured environments. In *International Symposium on Experimental Robotics (ISER)*, 2020.

[112] R. Polvara, M. Patacchiola, M. Hanheide, and G. Neumann. Sim-to-Real Quadrotor Landing via Sequential Deep Q-Networks and Domain Randomization. *Robotics*, 2020.

[113] R. Polvara, S. Sharma, J. Wan, A. Manning, and R. Sutton. Autonomous Vehicular Landings on the Deck of an Unmanned Surface Vehicle using Deep Reinforcement Learning. *Robotica*, pages 1867–1882, 2019.

[114] S. Prentice and N. Roy. The Belief Roadmap: Efficient Planning in Belief Space by Factoring the Covariance. *The International Journal of Robotics Research (IJRR)*, pages 1448–1465, 2009.

[115] L. O. R. Pérez, R. M. Silva, and J. M. Carranza. Real-Time Landing Zone Detection for UAVs using Single Aerial Images. In *International Micro Vehicle Competition and Conference (IMAV)*, 2018.

[116] T. Qin, P. Li, and S. Shen. VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator. *IEEE Transactions on Robotics (T-RO)*, pages 1004–1020, 2018.

[117] J. Redmon and A. Farhadi. YOLOv3: An Incremental Improvement. *arXiv preprint*, 2018.

[118] A. Rodriguez-Ramos, C. Sampedro, H. Bavle, I. G. Moreno, and P. Campoy. A Deep Reinforcement Learning Technique for Vision-Based Autonomous Multirotor Landing on a Moving Platform. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[119] T. Rouček, M. Pecka, P. Čížek, T. Petříček, J. Bayer, V. Šalanskỳ, D. Heřt, M. Petrlík, T. Báča, V. Spurnỳ, et al. DARPA Subterranean Challenge: Multi-robotic Exploration of Underground Environments. In *International Conference on Modelling and Simulation for Autonomous Systems (MESAS)*, 2019.

[120] J. Rückin, L. Jin, F. Magistri, C. Stachniss, and M. Popović. Informative Path Planning for Active Learning in Aerial Semantic Mapping. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[121] L. Schmid, M. Pantic, R. Khanna, L. Ott, R. Siegwart, and J. Nieto. An Efficient Sampling-Based Method for Online Informative Path Planning in Unknown Environments. *IEEE Robotics and Automation Letters (RA-L)*, pages 1500–1507, 2020.

[122] P. Schmuck and M. Chli. CCM-SLAM: Robust and efficient centralized collaborative monocular simultaneous localization and mapping for robotic teams. *Journal of Field Robotics (JFR)*, pages 763–781, 2019.

[123] P. Schmuck, T. Ziegler, M. Karrer, J. Perraudin, and M. Chli. COVINS: Visual-Inertial SLAM for Centralized Collaboration. In *IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*, 2021.

[124] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal Policy Optimization Algorithms. *arXiv preprint*, 2017.

[125] W. Schwarting, J. Alonso-Mora, and D. Rus. Planning and decision-making for autonomous vehicles. *Annual Review of Control, Robotics, and Autonomous Systems*, pages 187–210, 2018.

[126] M. Selin, M. Tiger, D. Duberg, F. Heintz, and P. Jensfelt. Efficient Autonomous Exploration Planning of Large-Scale 3-D Environments. *IEEE Robotics and Automation Letters (RA-L)*, pages 1699–1706, 2019.

[127] G. Shi, X. Shi, M. O’Connell, R. Yu, K. Azizzadenesheli, A. Anandkumar, Y. Yue, and S.-J. Chung. Neural Lander: Stable Drone Landing Control Using Learned Dynamics. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

[128] M. Siam, M. Gamal, M. Abdel-Razek, S. Yogamani, M. Jagersand, and H. Zhang. A Comparative Study of Real-Time Semantic Segmentation for Autonomous Driv-

ing. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2018.

[129] R. Sim and N. Roy. Global A-Optimal Robot Exploration in SLAM. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2005.

[130] A. J. Smith and G. A. Hollinger. Distributed Inference-Based Multi-Robot Exploration. *Autonomous Robots*, page 1651–1668, 2018.

[131] K. Solovey, O. Salzman, and D. Halperin. Finding a needle in an exponential haystack: Discrete RRT for exploration of implicit roadmaps in multi-robot motion planning. *The International Journal of Robotics Research (IJRR)*, pages 501–513, 2016.

[132] Y. Song, S. Naji, E. Kaufmann, A. Loquercio, and D. Scaramuzza. Flightmare: A Flexible Quadrotor Simulator. In *Conference on Robot Learning (CoRL)*, 2020.

[133] F. Stache, J. Westheider, F. Magistri, C. Stachniss, and M. Popović. Adaptive path planning for UAVs for multi-resolution semantic segmentation. *Robotics and Autonomous Systems (RAS)*, 2023.

[134] L. Steccanella, D. D. Bloisi, A. Castellini, and A. Farinelli. Waterline and obstacle detection in images from low-cost autonomous boats for environmental monitoring. *Robotics and Autonomous Systems (RAS)*, 2020.

[135] I. A. Şucan, M. Moll, and L. E. Kavraki. The Open Motion Planning Library. *IEEE Robotics & Automation Magazine (RAM)*, pages 72–82, 2012.

[136] P. Svestka and M. H. Overmars. Coordinated Path Planning for Multiple Robots. *Robotics and Autonomous Systems (RAS)*, pages 125–152, 1998.

[137] C. Symeonidis, E. Kakaletsis, I. Mademlis, N. Nikolaidis, A. Tefas, and I. Pitas. Vision-Based UAV Safe Landing Exploiting Lightweight Deep Neural Networks. In *The International Conference on Image and Graphics Processing (ICIGP)*, 2021.

[138] L. Teixeira, I. Alzugaray, and M. Chli. Autonomous Aerial Inspection using Visual-Inertial Robust Localization and Mapping. In *Field and Service Robotics (FSR)*, pages 191–204. Springer, 2018.

[139] L. Teixeira, M. R. Oswald, M. Pollefeys, and M. Chli. Aerial Single-View Depth Completion With Image-Guided Uncertainty Estimation. *IEEE Robotics and Automation Letters (RA-L)*, pages 1055–1062, 2020.

[140] Y. Tian, K. Liu, K. Ok, L. Tran, D. Allen, N. Roy, and J. P. How. Search and rescue under the forest canopy using multiple UAVs. *The International Journal of Robotics Research (IJRR)*, pages 1201–1221, 2020.

[141] J. Tordesillas and J. P. How. MADER: Trajectory Planner in Multiagent and Dynamic Environments. *IEEE Transactions on Robotics (T-RO)*, pages 463–476, 2022.

[142] M. Tranzatto, T. Miki, M. Dharmadhikari, L. Bernreiter, M. Kulkarni, F. Mascarich, O. Andersson, S. Khattak, M. Hutter, R. Siegwart, et al. CERBERUS in the DARPA Subterranean Challenge. *Science Robotics*, 2022.

[143] V. Usenko, L. von Stumberg, A. Pangercic, and D. Cremers. Real-time trajectory replanning for MAVs using uniform B-Splines and a 3D circular buffer. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[144] J. van den Berg, J. Snape, S. Guy, and D. Manocha. Reciprocal collision avoidance with acceleration-velocity obstacles. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[145] V. Vasilopoulos, G. Pavlakos, S. L. Bowman, J. D. Caporale, K. Daniilidis, G. J. Pappas, and D. E. Koditschek. Reactive Semantic Planning in Unexplored Semantic Environ-

ments Using Deep Perceptual Feedback. *IEEE Robotics and Automation Letters (RA-L)*, pages 4455–4462, 2020.

[146] H. Voos and H. Bou-Ammar. Nonlinear tracking and landing controller for quadrotor aerial robots. In *IEEE International Conference on Control Applications (CCA)*, 2010.

[147] S. Wang, F. Maffra, R. Mascaro, L. Teixeira, and M. Chli. Viewpoint-Tolerant Semantic Segmentation for Aerial Logistics. In *Pattern Recognition: DAGM German Conference (DAGM GCPR)*, 2021.

[148] T. Wang, V. Dhiman, and N. Atanasov. Learning Navigation Costs from Demonstrations with Semantic Observations. In *Conference on Learning for Dynamics and Control (L4DC)*, 2020.

[149] Z. Wang, L. Chen, H. Chen, H. Chen, and X. Jiang. Fast and Safe Exploration via Adaptive Semantic Perception in Outdoor Environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[150] Z. Wang, C. Xu, and F. Gao. Robust Trajectory Planning for Spatial-Temporal Multi-Drone Coordination in Large Scenes. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[151] WEF. 4 ways technology can help us respond to disasters. www.weforum.org/agenda/2018/01/4-ways-technology-can-play-a-critical-role-in-disaster-response, 2018. Accessed: 2023-01-23.

[152] R. E. Weibel and R. J. Hansman. Safety considerations for operation of unmanned aerial vehicles in the national airspace system. Technical report, Massachusetts Institute of Technology (MIT), 2006.

[153] W. Wu, F. Gao, L. Wang, B. Zhou, and S. Shen. Temporal scheduling and optimization for multi-MAV planning. In *Robotics Research: The International Symposium (ISRR)*, pages 813–831. Springer, 2022.

[154] X. Wu, S. Chen, K. Sreenath, and M. W. Mueller. Perception-aware receding horizon trajectory planning for multicopters with visual-inertial odometry. *arXiv preprint*, 2022.

[155] B. Yamauchi. A frontier-based approach for autonomous exploration. In *IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, 1997.

[156] B. Yamauchi. Decentralized coordination for multirobot exploration. *Robotics and Autonomous Systems (RAS)*, pages 111–118, 1999.

[157] J. Yan, X. Lin, Z. Ren, S. Zhao, J. Yu, C. Cao, P. Yin, J. Zhang, and S. Scherer. MUI-TARE: Multi-Agent Cooperative Exploration with Unknown Initial Position. *arXiv preprint*, 2022.

[158] T. Yang, P. Li, H. Zhang, J. Li, and Z. Li. Monocular Vision SLAM-Based UAV Autonomous Landing in Emergencies and Unknown Environments. *Electronics*, 2018.

[159] Y. Yang, K. Caluwaerts, A. Iscen, T. Zhang, J. Tan, and V. Sindhwani. Data efficient reinforcement learning for legged robots. In *Conference on Robot Learning (CoRL)*, 2020.

[160] J. Yu, J. Tong, Y. Xu, Z. Xu, H. Dong, T. Yang, and Y. Wang. SMMR-Explore: SubMap-based Multi-Robot Exploration System with Multi-robot Multi-target Potential Field Exploration Method. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[161] T. Zhang, G. Kahn, S. Levine, and P. Abbeel. Learning Deep Control Policies for Autonomous Aerial Vehicles with MPC-guided Policy Search. In *IEEE International Con-*

ference on Robotics and Automation (ICRA), 2016.

[162] Z. Zhang and D. Scaramuzza. Perception-aware Receding Horizon Navigation for MAVs. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[163] B. Zhou, F. Gao, L. Wang, C. Liu, and S. Shen. Robust and Efficient Quadrotor Trajectory Generation for Fast Autonomous Flight. *IEEE Robotics and Automation Letters (RA-L)*, pages 3529–3536, 2019.

[164] B. Zhou, H. Xu, and S. Shen. RACER: Rapid Collaborative Exploration with a Decentralized Multi-UAV System. *IEEE Transactions on Robotics (T-RO)*, pages 1–20, 2023.

[165] B. Zhou, Y. Zhang, X. Chen, and S. Shen. FUEL: Fast UAV Exploration Using Incremental Frontier Structure and Hierarchical Planning. *IEEE Robotics and Automation Letters (RA-L)*, pages 779–786, 2021.

[166] X. Zhou, X. Wen, Z. Wang, Y. Gao, H. Li, Q. Wang, T. Yang, H. Lu, Y. Cao, C. Xu, and F. Gao. Swarm of micro flying robots in the wild. *Science Robotics*, 2022.

[167] Z. Zhu, E. Schmerling, and M. Pavone. A convex optimization approach to smooth trajectories for motion planning with car-like robots. In *IEEE Conference on Decision and Control (CDC)*, 2015.

Curriculum Vitae

Luca Bartolomei
born May 7, 1993
citizen of Italy

2018–2023	*ETH Zurich, Switzerland* Doctoral studies at the Vision For Robotics Lab Supervised by Prof. Dr. Margarita Chli
09–12.2022	*Magic Leap, Inc., Zurich, Switzerland* Computer Vision Research Intern
2016–2018	*ETH Zurich, Switzerland* Master of Science in Robotics, Systems and Control
06–08.2016	*Schindler Aufzüge AG, Luzern, Switzerland* Software Engineer Intern
2012–2015	*Politecnico di Milano, Italy* Bachelor of Science in Mechanical Engineering
2007–2012	*Liceo Scientifico "G. Ferraris", Varese, Italy* High-school Diploma