CZECH TECHNICAL UNIVERSITY IN PRAGUE

Faculty of Electrical Engineering

MASTER'S THESIS

Design, Localization and Position Control of a Specialized UAV Platform for Documentation of Historical Monuments

Pavel Petráček

Thesis supervisor: **Ing. Martin Saska, Dr. rer. nat.**

Department of Cybernetics

I hereby declare that I wrote the presented thesis on my own and that I cited all the used information sources in compliance with the Methodical instructions about the ethical principles for writing an academic thesis.

In Prague on ....

MASTER‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name:	**Petráček Pavel**	Personal ID number:	**434753**
Faculty / Institute:	**Faculty of Electrical Engineering**		
Department / Institute:	**Department of Control Engineering**		
Study program:	**Cybernetics and Robotics**		
Branch of study:	**Cybernetics and Robotics**		

II. Master’s thesis details

Master’s thesis title in English:

Design, localization and position control of a specialized UAV platform for documentation of historical monuments

Master’s thesis title in Czech:

Návrh, lokalizace a stabilizace specializované bezpilotní helikoptéry pro dokumentaci historických objektů

Guidelines:

The goal of the thesis is to design a specialized unmanned aerial vehicle (UAV) for documentation of historical monuments without access to an external localization service (such as GPS) and to develop a method for its stabilization and localization in a map using onboard sensors.
1) Design and construct a UAV that respects requirements of its deployment in historical buildings.
2) Design and implement a system for self-localization and stabilization of the platform in a known map.
3) Verify the system using datasets recorded during real-world flights with a precise positional ground-truth. The localization will be realized offline by post-processing the data.
4) Integrate the localization system into the position feedback control loop and test it in the Gazebo simulator (an online experiment with real system with the onboard position feedback is not part of mandatory tasks, since several critical system components are currently under development and student cannot influence their realization in time).

Bibliography / sources:

[1] T. Furukawa, L. Dantanarayana, J. Ziglar, R. Ranasinghe and G. Dissanayake: "Fast global scan matching for high-speed vehicle navigation," 2015 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), San Diego, CA, 2015, pp. 37-42.
[2] M. Kam, Xiaoxun Zhu and P. Kalata: "Sensor fusion for mobile robot navigation," in Proceedings of the IEEE, vol. 85, no. 1, pp. 108-119, Jan. 1997.
[3] V Spurny, T Baca, M Saska, R Penicka, T Krajnik, J Thomas, D Thakur, G Loianno and V Kumar: Cooperative Autonomous Search, Grasping and Delivering in a Treasure Hunt Scenario by a Team of UAVs. Accepted in Journal of Field Robotics, 2018.

Name and workplace of master’s thesis supervisor:

Ing. Martin Saska, Dr. rer. nat., Multi-robot Systems, FEE

Name and workplace of second master’s thesis supervisor or consultant:

Date of master’s thesis assignment: **13.02.2019** Deadline for master's thesis submission: **24.05.2019**

Assignment valid until:
by the end of summer semester 2019/2020

Ing. Martin Saska, Dr. rer. nat. Supervisor’s signature	prof. Ing. Michael Šebek, DrSc. Head of department’s signature	prof. Ing. Pavel Ripka, CSc. Dean’s signature

Acknowledgments

I would like to thank my thesis adviser, Dr. Martin Saska, from the Department of Cybernetics of the Faculty of Electrical Engineering at the Czech Technical University in Prague. I value highly your patient guidance, friendly attitude, and supply of interesting projects, without which this work wouldn't be possible.

Besides my adviser, I thank colleagues from the Multi-Robot Systems group. I deeply appreciate the knowledge and skills gained through our collective efforts. Your valuable advice helped immensely during development of this work.

I wish to thank Ing. Matěj Petrlík for spending his time by reviewing this work. From alumni members of the group, I am particularly grateful to Dr. Nicolas Staub for his push in early stages of the work.

I would also like to express my inmost gratitude to my family, particularly my mother Dana and brother Norbert, whom never questioned my choices and always stood by my side.

Nevertheless, my thanks also go to my friend Martin for his honest friendship and shared fondness in beer tasting.

Finally, I express my profound gratitude to my long-term partner Pavla for her unfailing support, brisk humor and mutually shared values. You remind me daily, what is important in life.

Sincerely, thank you

Abstract

This thesis deals with design, autonomous localization and position control of unmanned multirotor aircraft for deployment in large historical monuments without access to global navigation systems. A specialized aerial platform respecting significant safety requirements was designed and manufactured for applications of this work. The main focus of this thesis lies in design and implementation of an active self-localization system with onboard multimodal sensory setup and a priori generated map. For that, we employ fusion of a global Monte Carlo Localization state estimation with local refinement by Iterative Closest Point algorithm, and an inertial measurement unit by Kalman Filter. The localization system is derived, implemented, integrated into the aircraft position control feedback, and evaluated in simulation and on real data obtained from experiments, conducted in an interior of a physical church. These experiments verified capability of the system to accurately estimate and autonomously control state of the aircraft in real time.

Keywords: unmanned aerial vehicle, GNSS-denied environments, indoor localization, active localization, multimodal sensor fusion, scan matching, Monte Carlo Localization, Iterative Closest Point, Kalman filtering, LiDAR, Point Cloud Library

Abstrakt

Tato práce se zabývá návrhem, autonomní lokalizací a pozičním řízením specializované bezpilotní helikoptéry sloužící k dokumentaci historických památek bez přístupu ke globálnímu navigačnímu systému. Pro účely této práce byla navržena a kompletně vyrobena specializovaná letová platforma, plně respektující důležité nároky na bezpečnost její aplikace. Hlavním přínosem této práce je návrh a implementace aktivního lokalizačního systému za pomoci palubních senzorů a předem vygenerované mapy. Lokalizační přístup zpracovává globální odhad stavu helikoptéry založený na metodách Monte Carlo, lokálně zpřesněným pomocí algoritmu Iterative Closest Point, a inerciální měřící jednotce pomocí Kalmanova filtru. Lokalizační systém byl navržen, implementován, integrován do zpětné vazby pozičního řízení helikoptéry, a posouzen v simulaci i na reálných datech získaných při experimentech v reálném kostele. Tyto experimenty potvrzují schopnost systému přesně odhadovat, sledovat a řídit stav helikoptéry v reálném čase bez zásahů operátora.

Klíčová slova: bezpilotní helikoptéra, prostředí bez přístupu GNSS, aktivní lokalizace, více-senzorová fúze, zarovnání skenů, Monte Carlo lokalizace, Iterative Closest Point, Kalmanův filtr, LiDAR, Point Cloud Library

Contents

List of Figures

List of Tables

Chapter 1: Introduction

Contents

In recent years, a massive advances have emerged in the technology of aerial vehicles capable of vertical landing and takeoff in terms of control, reliability, and autonomy. These multirotor vehicles, commonly remarked as Unmanned Aerial Vehicles (UAVs) or Micro Aerial Vehicles (MAVs), became extremely popular for their flexibility, diversity, and potential for both, amusement and functionality.

Typical multirotor consists of at least three motors mounted with fixed-pitch propellers, a rigid platform for electronics and a payload, and landing gear. The dimensions diversity of multirotors starts at just a few centimeters and goes up to the size of a car. Number of applications of multirotors is countless. To list a few examples, the applications include hobbyist/professional aerial photography, 3D mapping, precision agriculture, cargo delivery (including healthcare applications) and even drone racing. Furthermore, unmanned multirotors are capable of deployment in hazardous scenarios (search & rescue), as a security and emergence response, for inspections (tailing dams, overhead power lines, mines), surveying, exploration, and even in less known applications of urban planning, conservation of wildlife and nature, or telecommunications.

Since a multirotor is a dynamically unstable system, it constantly requires action inputs even for the simplest scenarios such as altitude control. With exclusive utilization of its own inertial measurement unit, a UAV is capable of self-stabilization, although is prone to drift since an integration error is accumulated over time. To provide a control assistance (e.g., automatic hover), a global navigation satellite system (GNSS) receiver is commonly included to provide a global position reference. However, usage of a GNSS is constrained by external conditions and its availability in the operating environment. In GNSS-denied environments, including most indoor and underground situations, the localization is not straightforward. To assist the UAV control or even introduce autonomy in these situations, three localization approaches can be employed: outer motion capture system referencing, relative localization, and onboard self-localization.

First, an outer reference system can track motion of the UAV and transfer information over an uplink connection. The inconvenience of these systems is their need to pre-set external devices. Two examples are Vicon Motion Capture [1, 2] and MarvelMind Robotics indoor "GPS" [3]. The Vicon tracks motion of infrared reflections of an onboard object using a set of infrared cameras. The MarvelMind Robotics indoor "GPS" triangulates a position of an onboard ultrasonic beacon by a set of static beacons according to the time-of-flight principle.

Second, a relative localization approach can be employed onboard, where a sensor (typically a camera) tracks a salient object in an image stream. Two examples are WhyCon [4, 5, 6] localizing black and white patterns from RGB cameras and UVDAR [7, 8, 9] capturing blinking ultra-violet diodes by an ultra-violet sensitive camera. These systems can be utilized both ways – as a ground motion capture system, or onboard the UAV to track motion of an onboard sensor relative to a sensed object.

Third, a passive or active self-localization can be employed to estimate a UAV state autonomously by processing solely onboard data. This approach is suitable for deployment in an unknown environment, with typical sensors being passive cameras and active LiDARs (Light Detection and Ranging). Methods utilizing cameras estimate motion of an onboard camera by finding correlations of consecutive frames in the image stream, while LiDAR-based methods estimate transformation between successive scans. Disadvantages of camera-based approaches, referred to as visual-odometry, lay in high computing power demand and need for feasible lighting conditions. Since an onboard computer needs to generate behavior for multiple subsystems (e.g., trajectory planning, collision avoidance, data processing), this approach might not be suitable for platforms with low processing power. In environments with bad lighting conditions, like large historical buildings, the usage of visual-odometry is likewise restricted, as discussed in Section 1.1. On the other hand, LiDARs measure time-of-flight of near-infrared light to determine distance. This principle is not restricted by lighting conditions, making it feasible for applications in dark areas. Overview of state-of-the-art LiDAR-based techniques is discussed in detail in Section 1.2.

1.1 Motivation

Restorers and conservators monitor states of historical monuments to study short and long term influence of time and restoration works on the monuments. Nowadays, during regular study services of influence of restoration works, the scaffolding is necessary to monitor conditions of a building. A UAV platform can supply the same documentation and inspection techniques used by the experts in locations inaccessible by people without the need of a large and expensive scaffolding installation, or in locations which had never been documented before. The UAV platform can reduce, improve, and significantly speed up the duration of the restoration works while scaling down their expenses.

The term historical monuments encompasses ancient or modern, war-damaged, dilapidated or restored cathedrals, chapels, churches, mausoleums, and temples of size varying from small chapels up to large cathedrals. Although the type of environments is diverse, the objects share common characteristics of bad lighting conditions and dust whirling due to an aerodynamic influence of deployed UAVs, especially in medium and high altitudes.

The end-users (restorers, conservators, historians) lack documentation of hardly accessible places in order to assess conditions of historical objects. Hence, in collaboration with

National Heritage Institute[1] of the Czech Republic, a set of historical monuments across the Czech Republic was selected for initial deployment of a UAV platform. Examples of already documented objects are Church of St. Mary Magdalene in Chlumín, Church of St. Maurice in Olomouc or Chateau Plumlov. The full list of objects to document can be found in [10]. Apart from the evaluation of structural conditions of a historical building, other objects of interest can be effectively documented for restoration or presentation purposes. That includes paintings, altars, statues, mosaics, frescoes, stained glass, pillars, or pipe organs.

The objective of an aerial platform is to autonomously document these objects and convey the acquired data to the end-users. However, the type of the acquired data may vary from the ordinary high-quality photographs taken in the visible spectrum to more exotic approaches – UV/IR spectrum photographs, radiography data, photogrammetry, or 3D reconstruction outcomes.

To overcome the problem of bad lighting conditions, the proposed solution is a multi-robotic system consisting of multiple UAVs. The first and main platform is a central UAV equipped with onboard sensors for data acquisition and self-localization, which is complemented by a set of supporting UAVs. Purpose of the supporting UAVs is to carry onboard lights to highlight details of documented objects of interest. Therefore, the supporting UAVs provide a mechanism to change illumination of the scene in order to highlight the object surface topography or its relief. The illumination techniques (raking light, three point lighting), together with initial results of the trajectory planning for formations of UAVs with supporting lighting approaches, are described in [11].

The main motivation of this work focuses on linking solutions of multiple robotic problems in order to create a robust robotic system providing the end-users a valuable tool to complement their work. A set of robotic problems in the application of UAV deployment for documentation of historical monuments consists of the UAV control, GNSS-denied localization, path and trajectory planning, sensor fusion, data processing, 3D mapping, formation flying, and data acquisition. An engineering part of this work contains design and manufacture of the application-tailored UAV platform developed with respect to critical safety requirements of the application. Second part of this work introduces reliable active GNSS-denied self-localization system tailored for deployment in the presented environments.

[1] `https://www.npu.cz/en`

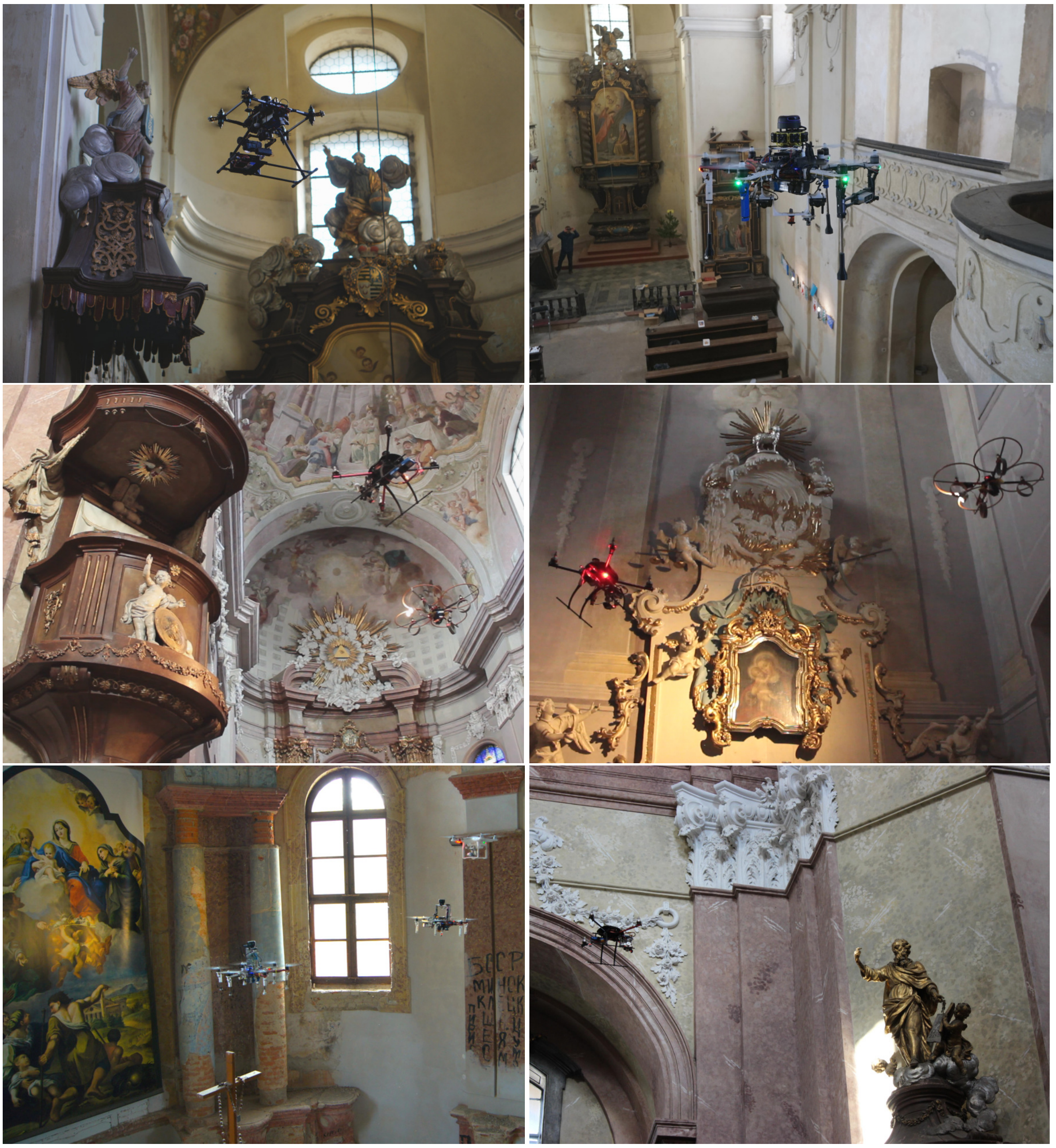


Figure 1.1: Documentation of historical monuments by a platform of unmanned aerial vehicles in cooperation with experts from restoration and conservation fields of study

1.2 Related Work

UAVs are being utilized for airborne documentation of archaeological excavations [12, 13], 3D surveying of archaeological sites and landscapes [14, 15, 16], 3D recording of cultural heritage [17], photogrammetry [15, 16], or visual inspecting [18, 19] for years. Most of these airborne systems are deployed outdoors and georeference its data using a global positioning system, which considerably simplifies the problem. The deployed UAVs for photogrammetry purposes carry mainly lightweight LiDARs.

Although simultaneous localization and 3D mapping techniques (SLAM) are well-studied for a single UAV [20, 21], and even for teams of UAVs [22, 23], only one work using UAVs in context of documentation of interiors of historical buildings was found [24]. This manuscript is based on a state-of-the-art visual SLAM with offline postprocessing to obtain a 3D model of the historical site. Vision-based SLAM systems can be found [25, 26], however unstable lighting conditions prevent to use exclusively vision-based approaches in the proposed architecture.

Similarly to numerous systems extending their applications with static terrestrial laser scanners to assist with modeling of the scanned sites [27, 28], the proposed system architecture employs a static laser scanner. Similarly to [29, 30], Monte Carlo Localization in 3D is employed from onboard 2D LiDAR. However, the global estimate refinement on a local map by a scan matching technique based on Iterative Closest Point [31] is proposed. Multiple manuscripts [32, 33] employ fusion of a scan matching of LiDAR data, inertial measurement unit and a vertically oriented rangefinder. Kalman filter is used to derive 3D position of a UAV. Although our approach employs similar techniques, it goes beyond by integration of the system into the UAV position control feedback.

Most of the aerial systems use commercial multirotor vehicles, which might not be optimal for their application. In this work, an application-tailored UAV platform suited for environments of historical monuments is introduced to maximize the platform capabilities. A multimodal sensor setup similar to [34] combines minimalist dimensions with maximal payload weight. In comparison with [18], the proposed platform shares similar dimensions, while it is equipped with a collision prevention system and its payload weight capacity is significantly higher.

Preceding works of the MRS group at FEE CTU in Prague developed subsystems, which are utilized throughout this work. In [35, 36], Model Predictive Control (MPC) and SO(3) controllers are proposed to control a UAV along a specified trajectory. These controllers were tested in harsh environments during two challenges of MBZIRC competition in 2017 [37, 38, 39]. Visual documentation of dark areas of interiors of large historical buildings by a formation of UAVs using a model predictive control on the receding horizon is proposed in [11]. Future extensions of this work plan to integrate and fuse state estimation based on optic-flow in onboard camera images [40] and ultra-violet relative localization system UVDAR [9] for mutual localization between UAVs during a formation flight.

1.3 Outline

This thesis is partitioned as follows. Foremost, a detailed description of a custom-built UAV platform suited for deployment in cluttered indoor environments of historical buildings is presented in Chapter 2. Second, an overall architecture of the application-tailored system for documentation of historical monuments is described in Chapter 3. Third, a concept of a global map as a baseline for accurate, reliable and robust localization system is presented. Map generation, interpretation and preprocessing is described in Chapter 4. Fourth, a proposed system for reliable and accurate map-based localization is described in detail in Chapter 5. The system covers combination of a global localization with local refinement, both fused together to provide final state estimation. Fifth, an analysis of the localization system is described in Chapter 6. This chapter presents simulation results, generation of a ground truth reference data, and evaluation of the system on real data taken inside Church of St. Mary Magdalene in Chlumín, Czech Republic. Sixth, the proposed localization system is integrated into the position control feedback of the UAV control in Chapter 7. Finally, the thesis is concluded in Chapter 8 by a discussion of the achieved objectives and future extensions of the work.

1.4 Mathematical Notation

Summary of mathematical notation used throughout the thesis is presented in Table 1.1.

Symbol	Example	Description
upper or lowercase letter	m, M, M	a scalar
bold upper or lowercase letter	$\mathbf{R}$, $\mathbf{h}$	a matrix or set
lowercase letter accented by a right arrow	$\vec{x}$	a column vector
upper index T	$\mathbf{R}^T$, $\vec{x}^{\,T}$	vector and matrix transpose
lower index k	M_k, $\mathbf{R}_k$, $\vec{x}_k$	M, $\mathbf{R}$, $\vec{x}$ at discrete time step k

Table 1.1: Overview of the mathematical notation

1.5 Table of Symbols

Chapter	Symbol	Description
Localization (Chapter 5)	m	Map
	k	Discrete time step
	$\mathbf{O}$, $\mathbf{T}$	Affine transformation matrix
	$\mathbf{R}$	Orientation matrix
	$\vec{t}$	Translation vector
	$\mathbf{I}$	Identity matrix
	$\vec{p}$, $\vec{\Omega}$	Robot position and orientation
	$\vec{q}$	Robot pose
	$\vec{v}$, $\vec{\omega}$	Robot linear and angular velocity
	Φ, Θ, Ψ	Roll, pitch, yaw
	$\vec{x}$	Robot state
	x, y, z, Ψ	State variables
	ω	Importance weight
	$\mathbf{S}$, s	Set of sensors, sensor
	L_s	Sensor s observations length
	$\mathbf{\Sigma}$	Covariance matrix
Monte Carlo Localization (Section 5.2)	$\vec{y}$	Robot observations
	y_s	Single observation of sensor s
	y_{min}, y_{max}	Observation range limits
	$\vec{u}$	Robot control input
	η	Normalization factor
	$\mathbf{H}$, $\mathbf{h}$	Set of hypotheses, hypothesis
	M	Total number of hypotheses
	M_{KLD}, M_{global}, M_{local}	Hypotheses sampling subset size
	M_{min}, M_{max}	Hypotheses set size limits
	$\epsilon_{0:6}$	Motion model noise
	σ_{hit}, λ_{short}, ν	Observation model parameters
	z_{lim}^{min}, z_{lim}^{max}	Sampling z-axis limits
	z_{lim}^{est}	Sampling altitude estimate
	z_{lim}^{δ}	Sampling z-axis offset
	α_{slow}, α_{fast}	Augmented-MCL decay rates
Scan Matching (Section 5.3)	$\mathbf{P}$, $\mathbf{Q}$	Source and reference point clouds
	$\vec{p}$, $\vec{q}$	Points $\vec{p} \in \mathbf{P}$ and $\vec{q} \in \mathbf{Q}$
	ϵ^{icp}	MSE of Iterative Closest Point
	$d_{1/2}$	Selection plane offset
Fusion (Section 5.4)	$\mathbf{A}$, $\mathbf{B}$, $\mathbf{K}$, $\mathbf{Q}$, $\mathbf{R}$, $\mathbf{S}$	Linear Kalman Filter matrices

Table 1.2: Summary of symbols utilized throughout Chapter 5

Chapter 2: Platform Design

Contents

The operational environments of our application are heavily volatile. Let the hereafter list be a summary of characteristics of such environments to account for in the context of an aerial vehicle hardware design.

- Cluttered and narrow parts withing an environment.
- Presence of obstacles difficult to detect (chandelier ropes, lightning cables).
- Presence of dynamic obstacles in low altitudes (people, environment changes).
- Fragile nature of the surveyed objects.
- Presence of wind gust due to the stack effect (opened windows, doors).
- Whirling of dust due to an aerodynamic influence of a UAV.

These difficulties request the platform to be minimal in size, compact, powerful, safe, equipped with onboard sensory equipment and maneuverable with a proportionally heavy payload. The payload weight can differ accordingly to the required data output of the end-user. Although for the data acquisition in the visible spectrum, example weights of the payload are given at the bottom of Table 2.1 (total weight of the example payload is 1129 g).

Furthermore, the maneuverability is an immeasurable variable, thus a relation between the maneuverability and a thrust-to-weight ratio of an aerial vehicle is proposed. The thrust-to-weight defines the ratio between maximal positive thrust (in kilograms) from all the propellers combined to the total weight of the vehicle in the standard atmosphere on Earth. Obviously, the ratio must be greater than 1:1 in order to obtain an ability to take off. It is also clear, that the larger the ratio, the better the maneuverability. However, the exact sufficient ratio is indefinable, since the maneuverability is also indefinable. Based on empirical thumb-rule experience of the author, which correlates with the opinion of the drone community [41], a sufficient ratio is defined to be close to 2:1 thrust-to-weight.

Although frame sets for aerial vehicles are available on the commercial market, only a few of them are suited for indoor applications. Besides, requirements of the application strictly specify minimalist dimensions of the UAV with respect to the onboard payload and sensors weight. Therefore, a custom aerial platform, suited for the specific demands of historical monuments documentation, is designed. The following list presents examples of similarly application-tailored vehicles to the one proposed in this thesis.

In [34], a similar vehicle with 3D omnidirectional sensor coverage is presented. However, this vehicle is larger in all dimensions (square base with 850 mm side) and its payload weight capabilities are lower in comparison to the proposed platform.

In [42], a vehicle for inventory applications and warehouse inspections using RFID markers is presented. These environments are characteristic by narrow passages between warehouse structures and large similar locations in the same environment. Historical objects share some of the environment characteristics similarities, which makes the application analogous. However, with 1700 mm diameter dimension, the vehicle in [42] is significantly larger (commercial DJI Matrice 600 frame).

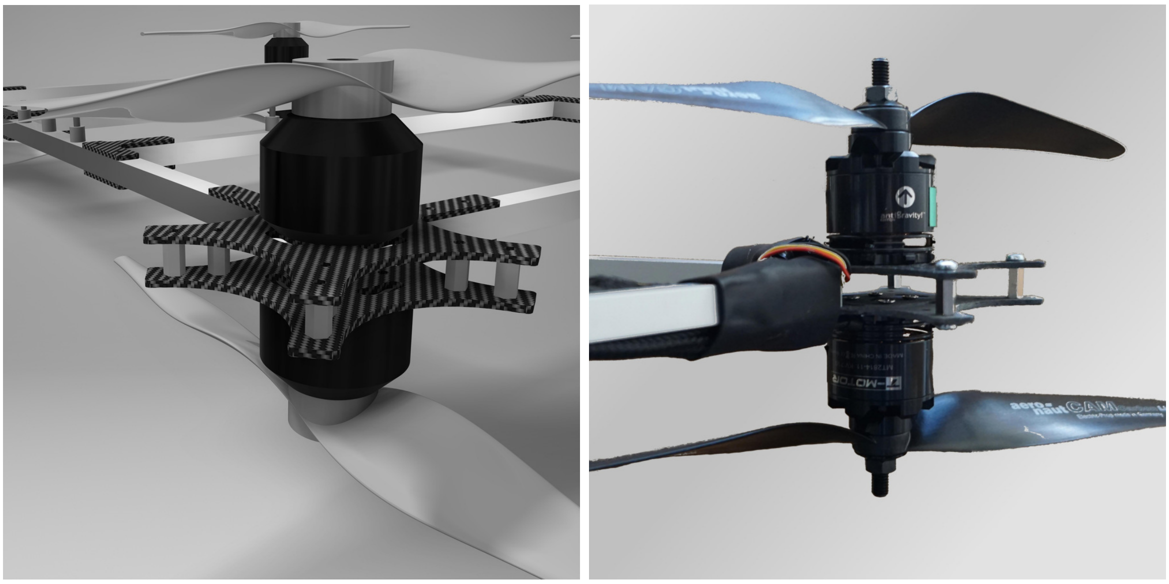
Authors in [18] presented an application-tailored system for inspection of chimneys. Apart from similar dimensions to our platform (800 mm), the chimney inspection application shares a considerable amount of characteristics. Identically to our application, the output of the system is data to be inspected by an expert or an end-user. However, their application environment - chimneys - is highly predictable and homogeneous, which makes it easier to apprehend.

2.1 Hardware Design

The first objective of this thesis is to design a physical body for an autonomous drone with capabilities of safe flight in an indoor aerial operational space. A basic multirotor vehicle consists of a physical body (frame), drive (motors, ESCs, an accumulator and propellers), an autopilot, and a radio controller receiver. As a consequence of autonomy, additional sensory equipment is necessary to obtain capabilities of an environment sensing, self-localization, and a behavior generation.

As aforementioned, the cluttered environment demands a balance between the dimensions of the vehicle and a payload weight limit. A number of rotors, together with their particular configuration, and length of attached propellers are aspects correlating the most with final dimensions of the multirotor vehicle. A drive system of coaxial rotors is selected based on the hereafter introduced methods. A coaxial rotor consists of two rotors on the same axis of rotation, which are contra-rotating, as shown in Figure 2.1. Usage of coaxial rotors slightly reduces the propulsion system efficiency [43]. On the other hand, it significantly reduces the dimensions of the vehicle and the motors redundancy adds disturbance and single point of failure resistance. Having an expected total weight of the vehicle, four coaxial motors in the octocopter X8 configuration shown in Figure 2.2 provide sufficient thrust.

Fixed-pitch propellers with the length of 12 inches and 5 inches pitch are used. The pitch parameter defines the displacement of a propeller after one complete revolution in a solid environment. Typically, the larger the propeller pitch, the higher torque is needed, since more air resistance is produced on the propeller surface area. Selection of the propellers is influenced by the minimalist requirement for the vehicle's dimensions and commercial availability of these particular propellers. In the octocopter X8 rotors configuration, enough motor thrust is



(a) Designed coaxial mount

(b) Physical coaxial mount

Figure 2.1: A coaxial rotor mount of the proposed design

provided, while propellers length is minimized and the electric power input of the motors is kept under a producer-defined margin.

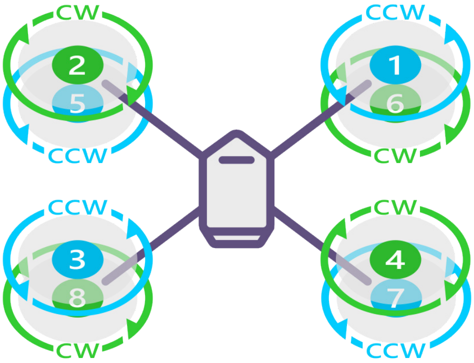


Figure 2.2: Multirotor with four coaxial rotors in X8 configuration [44]

The first step of a robot design is selection of suitable drive components. Firstly, the expected total weight of the robot is approximated, and the appropriate drive parameters are selected accordingly. To design an optimal drive subsystem, an online tool *eCalc* [45] is utilized to estimate suitable drive parameters and limitations. The total robot weight approximation is specified in Table 2.1 and the drive design from *eCalc* is specified in Figure 2.3a.

Two in parallel connected LiPo accumulators, each with four in series connected cells, are utilized to provide power to onboard electronics and the drive system. Nominal voltage for a LiPo accumulator is 3.7 V per one cell, which results in the main power source of the vehicle being a voltage source with a nominal voltage of 14.8 V. The reason for this choice is our remnant possession of these accumulators from previous research projects and MBZIRC 2017 competition [37, 38, 39].

Proposed design of the drive system, shown in Figure 2.3a, yields a thrust-to-weight ratio of 1.8:1. This configuration leads to a controllable vehicle with slightly lowered maneuverability in comparison with the recommended ratio. For the final version of the vehicle, the power source upgrade to one LiPo accumulator with six in series connected cells is intended, which shall enhance the flight characteristics. As shown in Figure 2.3b, replacing the power supply while swapping for suitable motors leads to longer flight time, increase of payload weight limit and thrust-to-weight ratio improvement.

Physical dimensions of the designed vehicle without propellers are specified in Figure 2.8 and respectively in Figure 2.10 with propellers. The resulting parameters of the vehicle are 10 minutes time of flight with a load equal to the example payload weight in Table 2.1.

2.2 Components

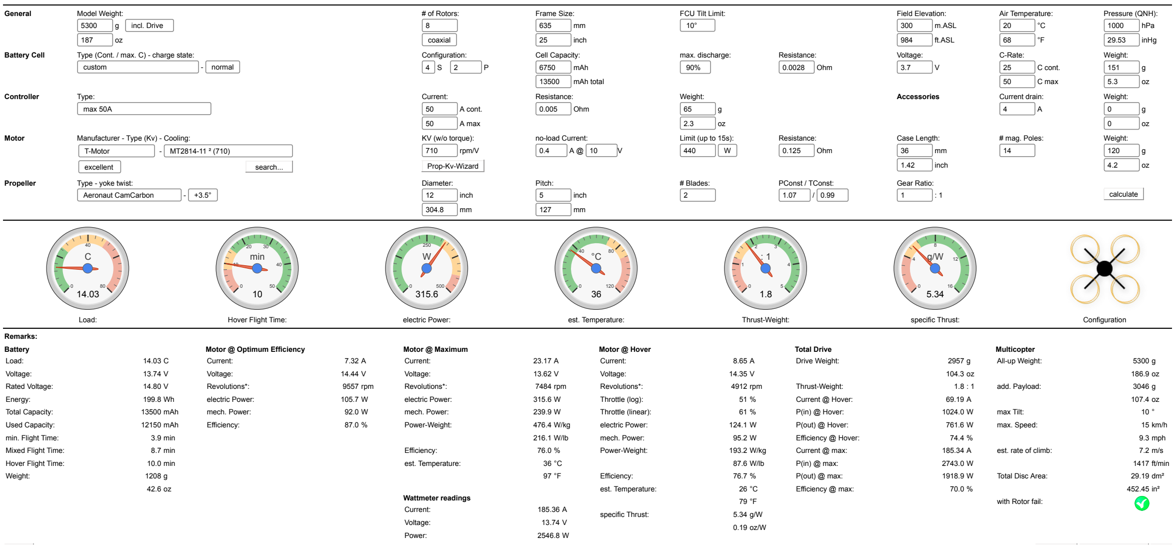
In this section, a list of the components of the designed platform is introduced. The list is also summarized by Table 2.1.

Motors MT2814 from company T-Motor suit well the voltage range of the selected accumulators. These motors are part of the DC brushless category, as mostly used in the drone industry. Important aspect of a DC brushless motor in the field of multirotors is unit kV. This unit specifies theoretical number of motor revolutions per minute per volt (RPM/V) of an unloaded motor. Obviously, adding a propeller to a motor slows it down and the theoretical value can never be reached. Generally, lower kV value leads to slower rotation rise time and larger torque, which the motor is able to produce and hence the larger propellers are suitable. The selected motors are characterized by 710 kV.

Electronic Speed Controller is a link between motors and an autopilot. It translates signals from autopilot to trigger the three coil segments in a DC brushless motor with three-phase DC pulses. The requirements for precision, durability, and functionality of these electronic controllers are extremely strict, thus buying these parts from verified manufacturers is recommended. Important parameters of ESCs are the allowed continuous and burst current values, and a communication protocol they employ.

Propellers produce lift in the forward direction, which is referred to as thrust. The aerodynamic mechanism of a lift production by an air pressure difference on the propeller blade side surface is a well-studied principle in the field of aeronautics. Without loss of generality, a detailed explanation of this principle is ignored. For historical reasons, wooden propellers were regularly utilized. However, modern plastic and carbon materials took over in the multirotor field and nowadays is rare to find a wooden propeller on an aerial vehicle, with an exception of airplanes.

Accumulator serves as a power supply for each electronic part on the vehicle. It powers the motors, ESCs, an onboard computer, all the sensory equipment and possibly even the payload. Together with tethered drones, where a tether is attached to a drone providing a power supply and a data link, accumulators are nowadays the only technology used as a power source for an aerial vehicle. However, with the emergence of larger multirotors for cargo consignment or man transport, even petrol powered systems might become common. From the spectrum of various accumulator chemistries and types, particularly LiPo accumulators



General
Model Weight: 5300 g incl. Drive / 187 oz
# of Rotors: 8 coaxial
Frame Size: 635 mm / 25 inch
FCU Tilt Limit: 10°
Field Elevation: 300 m.ASL / 984 ft.ASL
Air Temperature: 20 °C / 68 °F
Pressure (QNH): 1000 hPa / 29.53 inHg

Battery Cell
Type (Cont. / max. C) - charge state: custom - normal
Configuration: 4 S 2 P
Cell Capacity: 6750 mAh / 13500 mAh total
max. discharge: 90%
Resistance: 0.0028 Ohm
Voltage: 3.7 V
C-Rate: 25 C cont. / 50 C max
Weight: 151 g / 5.3 oz

Controller
Type: max 50A
Current: 50 A cont. / 50 A max
Resistance: 0.005 Ohm
Weight: 65 g / 2.3 oz

Accessories
Current drain: 4 A
Weight: 0 g / 0 oz

Motor
Manufacturer - Type (Kv) - Cooling: T-Motor - MT2814-11 ² (710) / excellent / search...
KV (w/o torque): 710 rpm/V
Prop-Kv-Wizard
no-load Current: 0.4 A @ 10 V
Limit (up to 15s): 440 W
Resistance: 0.125 Ohm
Case Length: 36 mm / 1.42 inch
# mag. Poles: 14
Weight: 120 g / 4.2 oz

Propeller
Type - yoke twist: Aeronaut CamCarbon - +3.5°
Diameter: 12 inch / 304.8 mm
Pitch: 5 inch / 127 mm
# Blades: 2
PConst / TConst: 1.07 / 0.99
Gear Ratio: 1 : 1
calculate

Load: 14.03
Hover Flight Time: 10
electric Power: 315.6
est. Temperature: 36
Thrust-Weight: 1.8
specific Thrust: 5.34
Configuration

Remarks:

Battery

Load:	14.03 C
Voltage:	13.74 V
Rated Voltage:	14.80 V
Energy:	199.8 Wh
Total Capacity:	13500 mAh
Used Capacity:	12150 mAh
min. Flight Time:	3.9 min
Mixed Flight Time:	8.7 min
Hover Flight Time:	10.0 min
Weight:	1208 g
	42.6 oz

Motor @ Optimum Efficiency

Current:	7.32 A
Voltage:	14.44 V
Revolutions*:	9557 rpm
electric Power:	105.7 W
mech. Power:	92.0 W
Efficiency:	87.0 %

Motor @ Maximum

Current:	23.17 A
Voltage:	13.62 V
Revolutions*:	7484 rpm
electric Power:	315.6 W
mech. Power:	239.9 W
Power-Weight:	476.4 W/kg
	216.1 W/lb
Efficiency:	76.0 %
est. Temperature:	36 °C
	97 °F

Wattmeter readings

Current:	185.36 A
Voltage:	13.74 V
Power:	2546.8 W

Motor @ Hover

Current:	8.65 A
Voltage:	14.35 V
Revolutions*:	4912 rpm
Throttle (log):	51 %
Throttle (linear):	61 %
electric Power:	124.1 W
mech. Power:	95.2 W
Power-Weight:	193.2 W/kg
	87.6 W/lb
Efficiency:	76.7 %
est. Temperature:	26 °C
	79 °F
specific Thrust:	5.34 g/W
	0.19 oz/W

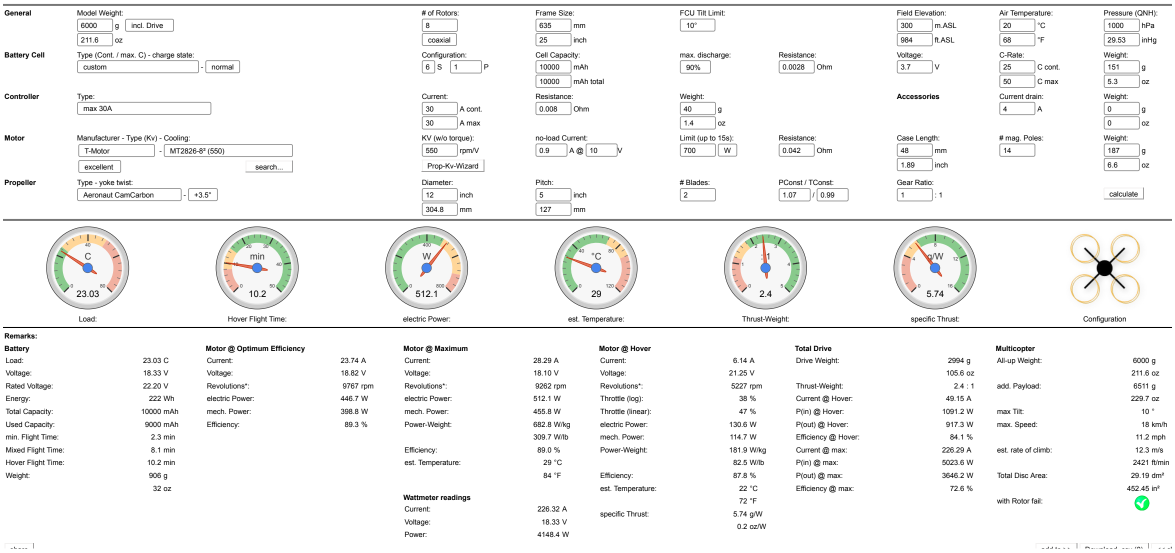
Total Drive

Drive Weight:	2957 g
	104.3 oz
Thrust-Weight:	1.8 : 1
Current @ Hover:	69.19 A
P(in) @ Hover:	1024.0 W
P(out) @ Hover:	761.6 W
Efficiency @ Hover:	74.4 %
Current @ max:	185.34 A
P(in) @ max:	2743.0 W
P(out) @ max:	1918.9 W
Efficiency @ max:	70.0 %

Multicopter

All-up Weight:	5300 g
	186.9 oz
add. Payload:	3046 g
	107.4 oz
max Tilt:	10 °
max. Speed:	15 km/h
	9.3 mph
est. rate of climb:	7.2 m/s
	1417 ft/min
Total Disc Area:	29.19 dm²
	452.45 in²
with Rotor fail:	✓

(a) Two in parallel connected LiPo accumulators, both with 4 in series connected cells (4SP2)



General
Model Weight: 6000 g incl. Drive / 211.6 oz
# of Rotors: 8 coaxial
Frame Size: 635 mm / 25 inch
FCU Tilt Limit: 10°
Field Elevation: 300 m.ASL / 984 ft.ASL
Air Temperature: 20 °C / 68 °F
Pressure (QNH): 1000 hPa / 29.53 inHg

Battery Cell
Type (Cont. / max. C) - charge state: custom - normal
Configuration: 6 S 1 P
Cell Capacity: 10000 mAh / 10000 mAh total
max. discharge: 90%
Resistance: 0.0028 Ohm
Voltage: 3.7 V
C-Rate: 25 C cont. / 50 C max
Weight: 151 g / 5.3 oz

Controller
Type: max 30A
Current: 30 A cont. / 30 A max
Resistance: 0.008 Ohm
Weight: 40 g / 1.4 oz

Accessories
Current drain: 4 A
Weight: 0 g / 0 oz

Motor
Manufacturer - Type (Kv) - Cooling: T-Motor - MT2826-8² (550) / excellent / search...
KV (w/o torque): 550 rpm/V
Prop-Kv-Wizard
no-load Current: 0.9 A @ 10 V
Limit (up to 15s): 700 W
Resistance: 0.042 Ohm
Case Length: 48 mm / 1.89 inch
# mag. Poles: 14
Weight: 187 g / 6.6 oz

Propeller
Type - yoke twist: Aeronaut CamCarbon - +3.5°
Diameter: 12 inch / 304.8 mm
Pitch: 5 inch / 127 mm
# Blades: 2
PConst / TConst: 1.07 / 0.99
Gear Ratio: 1 : 1
calculate

Load: 23.03
Hover Flight Time: 10.2
electric Power: 512.1
est. Temperature: 29
Thrust-Weight: 2.4
specific Thrust: 5.74
Configuration

Remarks:

Battery

Load:	23.03 C
Voltage:	18.33 V
Rated Voltage:	22.20 V
Energy:	222 Wh
Total Capacity:	10000 mAh
Used Capacity:	9000 mAh
min. Flight Time:	2.3 min
Mixed Flight Time:	8.1 min
Hover Flight Time:	10.2 min
Weight:	906 g
	32 oz

Motor @ Optimum Efficiency

Current:	23.74 A
Voltage:	18.82 V
Revolutions*:	9767 rpm
electric Power:	446.7 W
mech. Power:	398.8 W
Efficiency:	89.3 %

Motor @ Maximum

Current:	28.29 A
Voltage:	18.10 V
Revolutions*:	9262 rpm
electric Power:	512.1 W
mech. Power:	455.8 W
Power-Weight:	682.8 W/kg
	309.7 W/lb
Efficiency:	89.0 %
est. Temperature:	29 °C
	84 °F

Wattmeter readings

Current:	226.32 A
Voltage:	18.33 V
Power:	4148.4 W

Motor @ Hover

Current:	6.14 A
Voltage:	21.25 V
Revolutions*:	5227 rpm
Throttle (log):	38 %
Throttle (linear):	47 %
electric Power:	130.6 W
mech. Power:	114.7 W
Power-Weight:	181.9 W/kg
	82.5 W/lb
Efficiency:	87.8 %
est. Temperature:	22 °C
	72 °F
specific Thrust:	5.74 g/W
	0.2 oz/W

Total Drive

Drive Weight:	2994 g
	105.6 oz
Thrust-Weight:	2.4 : 1
Current @ Hover:	49.15 A
P(in) @ Hover:	1091.2 W
P(out) @ Hover:	917.3 W
Efficiency @ Hover:	84.1 %
Current @ max:	226.29 A
P(in) @ max:	5023.6 W
P(out) @ max:	3646.2 W
Efficiency @ max:	72.6 %

Multicopter

All-up Weight:	6000 g
	211.6 oz
add. Payload:	6511 g
	229.7 oz
max Tilt:	10 °
max. Speed:	18 km/h
	11.2 mph
est. rate of climb:	12.3 m/s
	2421 ft/min
Total Disc Area:	29.19 dm²
	452.45 in²
with Rotor fail:	✓

(b) One LiPo accumulator with 6 in series connected cells (6SP1) and increased total weight

Figure 2.3: Designs of the vehicle's drive components and its flight characteristics estimate from *eCalc* tool [45]

are used for multirotors due to their convenient properties of lightweight, high capacity, large discharge rate, and a customizable shape.

Frame provides a physical body and defines the dimensions of the whole vehicle. Detailed description of the frame component is present in Section 2.3.

Autopilot is a system for an underlying attitude stabilization and control of a vehicle, using onboard accelerometers, barometers, magnetometers and gyroscopes. The selected autopilot Pixhawk Cube, previously featured as Pixhawk 2.1, is an open-hardware autopilot broadly used by the robotic community. Particularly the Cube provides high redundancy for its separated, dampened and thermally stabilized IMU system, which comprises of three independent accelerometers, gyroscopes and magnetometers, and two independent barometers.

Central processing unit serves as the brain of an aerial vehicle. It provides computational power for sensor data processing, trajectory planning, state estimation, mission supervision and many other subsystems of an autonomous vehicle. Based on our experience with onboard computers, Intel NUC7i7 (Intel® Core™ i7-8650U, 1.9 GHz with Turbo Boost and Hyper-Threading technologies, 8 GB RAM) is relied on for its compact size, number of various input ports and powerful hardware components. The connection link between the autopilot and an onboard computer is established over a bidirectional serial line with utilization of MAVLink protocol [46]. The connection diagram between vehicle components and the central processing unit is presented in Figure 2.4.

Radio communication between a ground operator and the vehicle is arranged via a 2.4 GHz frequency receiver-transmitter channels. The autopilot adopts commands from the onboard receiver during a non-autonomous flight mode. This link between a ground operator and the vehicle is exceptionally important since the mission operator is obliged to take over the vehicle control in case of any system malfunction.

Video transmission and telemetry provides a live video feed with current flight parameters to a ground operator. This functionality enables a mission supervision and a visual feedback to the operator based on the video feed from a First Person View (FPV) camera. In consequence of the visual feedback, an end-user can adjust mission objective mid-air. That includes objects of interest specification, mission repetition, detailed data acquisition of a certain surface or change of sensing parameters (e.g., light conditions, camera exposure time).

2.2.1 Sensory Equipment

This section lists down selected sensory equipment to supply enough onboard sensing capabilities forward to an autonomous mission. The equipment, highlighted on an airborne UAV in Figure 2.7, provides vision, laser and ultrasonic-based sensing information in various direction to effectively cover the environment.

Environment scanner is the primary source of information about the vehicle's neighborhood. For our purposes, a planar 360° rotational laser scanner RPLIDAR A3 is employed. The scanner parameters are 25 m range radius, scan rate up to 20 Hz, angular resolution down to 0.3375° and sample rate up to 16 000 samples per second. The scanner is fixed to the UAV frame and therefore the sensing plane orientation corresponds to the orientation of the UAV. Figure 2.5 illustrates data produced by RPLIDAR A3 scanner to visually manifest the output of the sensor. Nevertheless, the sensor choice could be conveniently replaced by a different alternative. For example, a 3D scanner Velodyne Puck Lite can be employed for more robust state estimation in defiance of its heavier dispositions.

A range measurement decay with increasing distance was identified during RPLIDAR A3 evaluation in Section 5.2.2. Hence throughout the thesis, RPLIDAR A3 distance decay of a range measurement x is corrected by relation

$$f_{corr}(x) = x + 0.00512x^2 \tag{2.1}$$

determined by quadratic least squares regression, as illustrated in Figure 2.6.

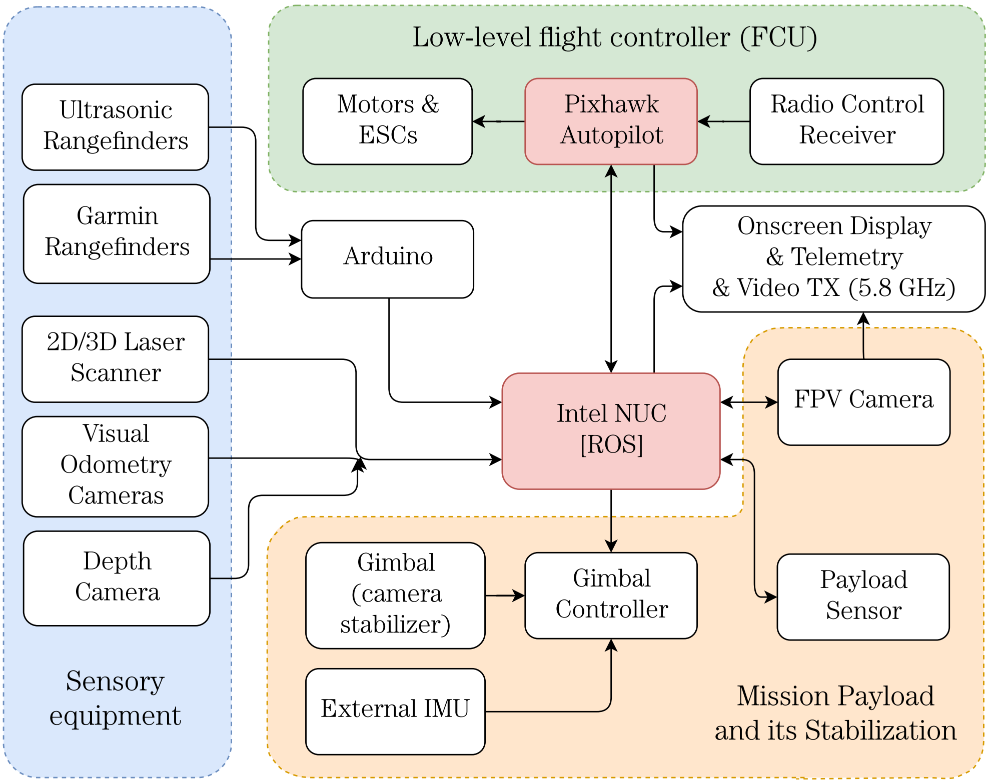
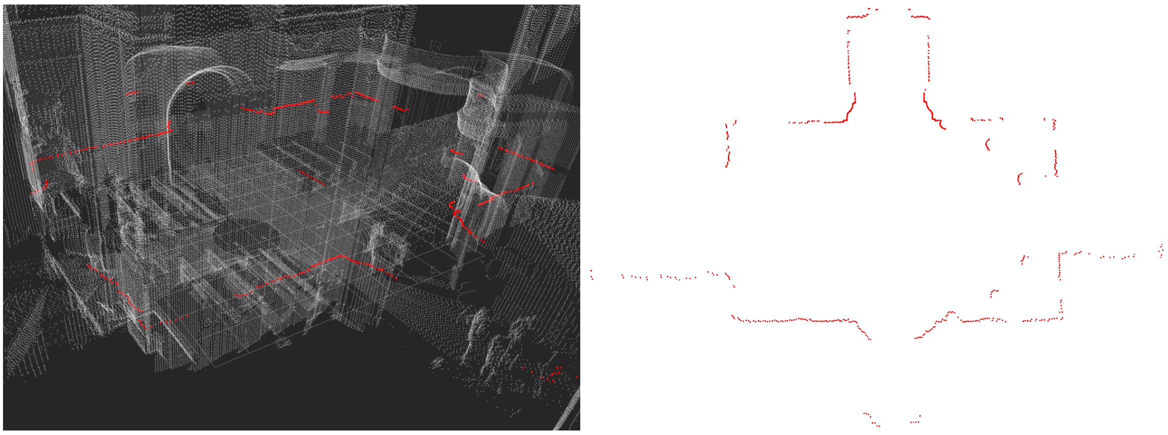


Figure 2.4: High-level connection diagram of the designed UAV electronics, where the sensory equipment is described in detail in Section 2.2.1



(a) View on data embedded in a map obtained by a 3D scanner, as described in Section 4.1

(b) Top-view on a raw data

Figure 2.5: Example of data from planar 360° scanner RPLIDAR A3 taken onboard an airborne UAV in Church of St. Mary Magdalene in Chlumín introduced in Chapter 4

Laser rangefinders oriented vertically to measure point distance in positive and negative z-axis of the autopilot provide an estimate of distance from the ground (altitude) and distance from the ceiling respectively. Specifications of the selected Garmin Lidar Lite sensors are

Component group	Component	Specifications	Weight [g]
Drive	Motors	T-Motor MT2814	8×120
	Electronic Speed Controller	Foxy Multi Opto	8×31
	Propellers	CAM-Carbon 12 × 5	8×10
	Accumulator	Tattu 4S LiPo	2×605
Frame	Base	Carbon and aluminium parts	630
	Collision Prevention System	Propeller guards	300
	Landing gear	Carbon tubes	50
Electronics	Central processing unit	Intel NUC7i7	150
	Autopilot	Pixhawk Cube (2.1)	39
	Sensors data acquisition board	Arduino Nano	7
	Radio control receiver	Optima 9, 2.4 GHz	22
Sensors	Optical rangefinders	Garmin Lidar Lite v3	2×22
	Ultrasonic rangefinders	HC-SR04	4×9
	Front-facing depth camera	RealSense D435	72
	Visual odometry camera	mvBlueFOX	20
	Rotational 2D laser scanner	RPLIDAR A3	190
Video transmission & telemetry	Analog camera	RunCam 2	49
	Analog video transmitter	Boscam 5.8 GHz	20
	Onscreen display	MinimOSD	10
Payload	Camera	Sony Alpha A6000	344
	Fixed focal length lens	Sony 16 mm f/2.8 SEL	67
	Stabilization unit	2-axes Dragon Gimbal	358
	Stabilization unit controller	SimpleBGC 32 bit Tiny	100
	LED light	Aputure LED AL-F7	260
			5266

Table 2.1: List of the proposed platform components with their specification and weight, and an example of a payload for documentation in visible light spectrum

according to theirs datasheet 40 m measurement range with 1 cm resolution and ±2.5 cm accuracy under 5 m, and ±10 cm accuracy over 5 m.

Visual odometry camera with a suitable lens provides fast image information for a vision based state estimation. In our case, an onboard-running optical flow algorithm [40] is utilized for the vehicle velocities estimation from the motion of objects in a visual scene. For redundancy of independent localization sources, a fusion of multiple velocity estimates is proposed from the optical flow field of the scene below and in front of the UAV, if enough processing capacity is provided. Hence, two mvBlueFOX cameras (front and down facing) are integrated into the platform. Their parameters are 25 Hz frame rate, rolling shutter, 1/3″ optical sensor size and up to 1280 × 960 resolution. The optical flow algorithm implementation [40] is not color-based, hence a greyscale camera version with camera sensor sensitive solely

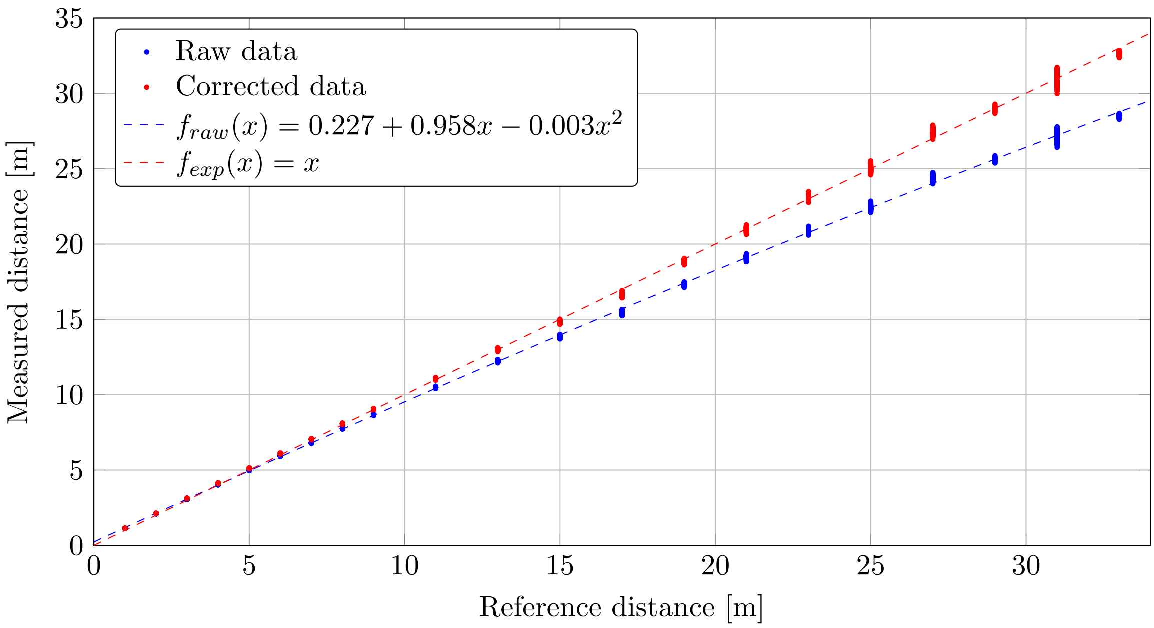


Figure 2.6: Bias correction of RPLIDAR A3 range measurement according to Equation 2.1

to light intensity (without Bayer mask) is preferred to maximize information sensitivity.

Ultrasonic rangefinders HC-SR04 provide a low-level safety mechanism, as they are used directly in the lowest levels of the UAV control. In other words, they serve as virtual bumpers to prevent undesirable collisions with the environment. These sensors with 4 m maximal measurement range, 3 mm resolution, and 15° field of view are mounted on diagonal axes of the vehicle to roughly cover the possible areas of collisions.

Depth camera oriented forward provides depth information in the vehicle's x-axis. The employed camera Intel RealSense D435 produces syncrhonized RGB-D data stream with resolution of 1280 × 720 px up to rate of 90 Hz. The depth information can be further utilized for attitude estimation, localization, map building, and 3D reconstruction.

2.3 Frame Design

A full design draft of the frame was modeled in a 3D modeling software[1] in order to expose potential drawbacks of a custom design. Then a prototype was assembled from aluminum square tube profiles and carbon fiber composites. Carbon fiber composite is a modern material in the aerial field, where its main advantage stands in a great proportion between lightweight, and high stiffness and strength. All parts from carbon fiber composite (henceforth carbon parts) were cut on a milling machine either by the thesis author or by an external manufacturer. The design of the vehicle is divided into three components - frame, base and collision prevention system.

[1] Autodesk Inventor Professional 2019 with a standalone student license provided by the Czech Technical University

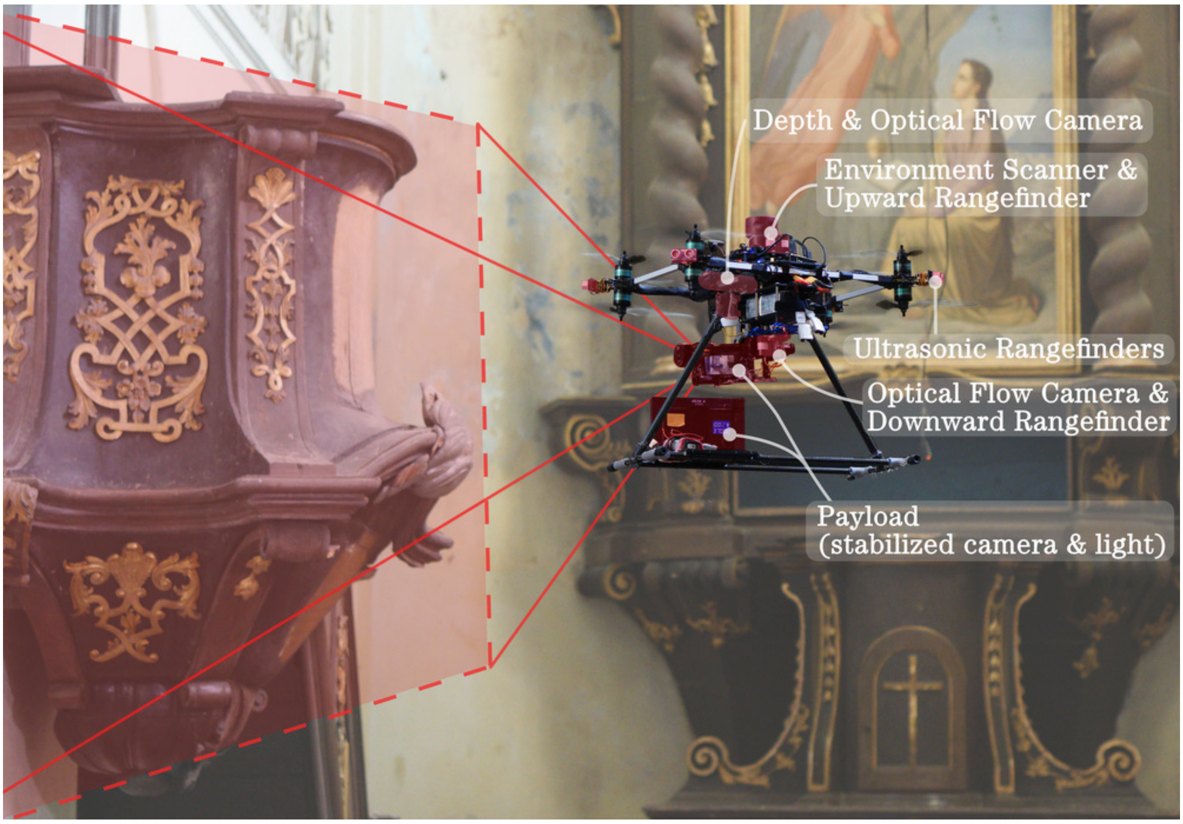

Depth & Optical Flow Camera
Environment Scanner & Upward Rangefinder
Ultrasonic Rangefinders
Optical Flow Camera & Downward Rangefinder
Payload (stabilized camera & light)

Figure 2.7: Visualization of onboard sensory equipment and a payload field of view

Purpose of the frame component is to provide a rigid and connected platform to mount the coaxial motors on. The frame is an H-shape with reinforcements at the side segments. These reinforcements serve as a necesary precaution to handle twist momentums, which occur at the motor mounts segments, hence to prevent material bends and wear outs. The final shape is defined as a rectangle with the shorter side dimension designed to fit the propellers and longer side dimension extended by a width of the base component, as illustrated in Figure 2.8. The frame incorporates aluminum square bars serving as general shape connection links joined by carbon motor mounts and L-shaped links. Furthermore, the frame involves soft mount dampers as a connection link between the frame and base, which reduce vibrations transfer from the motors to base components.

Base component serves as a platform for attachment of onboard electronics, power distribution, batteries, and even a payload. It is designed as a stack of carbon plates with 2 mm thickness, as illustrated in Figure 2.9, where each layer serves its own functionality. Starting from the bottom, the leading layer is a platform for accumulators attachment. On the bottom side of this layer, downward looking sensors are located. One layer above, power distribution is handled and non-sensory electronics is fixed. Looking from below at this layer, the landing gear is attached to its central part and the frontal side is reserved for a payload hinge. One more layer above, a general platform for the central processing unit, radio receiver, and alternative sensors are located. This layer is the only one connected to the frame component, whereas the rest is stacked either below or above it. The base-frame connection is established

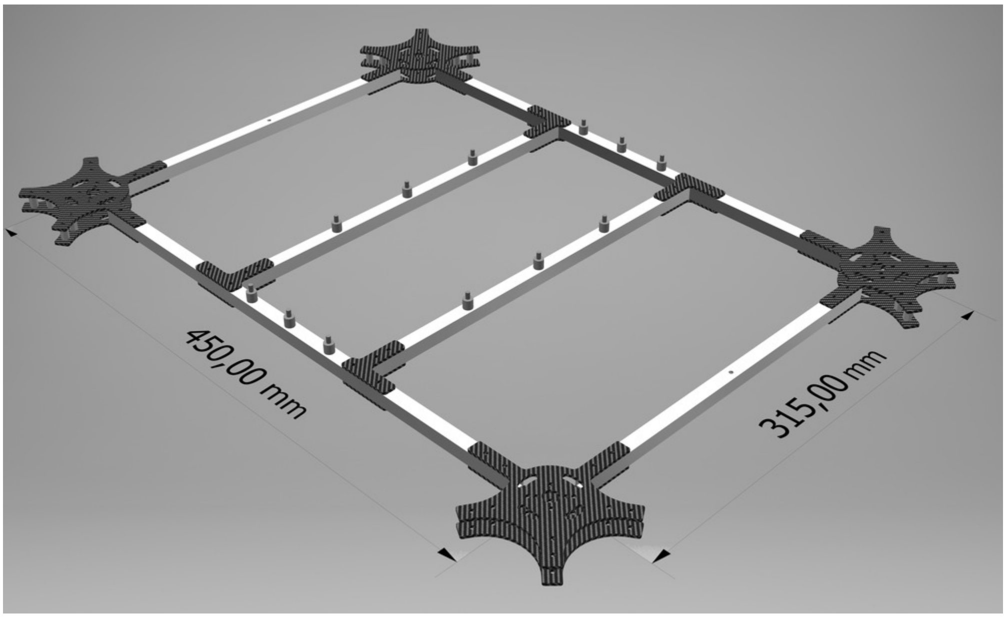
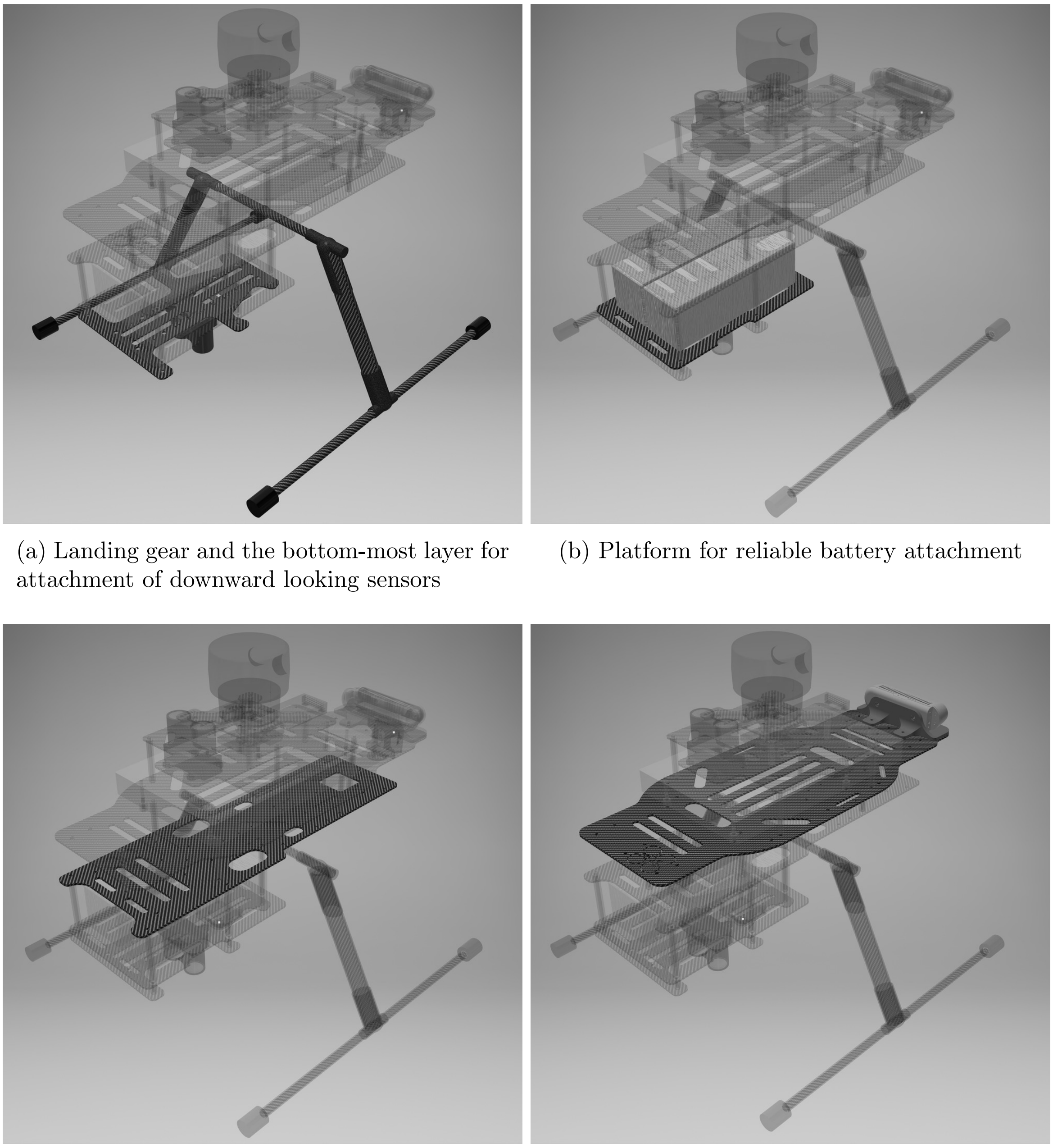


Figure 2.8: Visualization of the frame component with motor platforms and soft mounts for the base component

via the aforementioned soft mount dampers. One more layer above is a smaller platform for the autopilot and upward looking sensors. And finally on the top is a layer reserved for sensors requiring unobscured visibility.

The third component, the collision prevention system, is a propeller guard system isolating an outer environment and hastily rotating propellers. These systems particularly common on multirotor vehicles with operational spaces in close proximity to people and obstacles. For deployment in a historical monument, the collision prevention system is compulsory due to the requirement of absolute safety. The system protects conceivable fragile objects of interest and prevents possible property damages. On the other hand, it likewise protects the propellers from outer sources, since a fractured propeller could destabilize the ensemble vehicle, leading to possibilities of extensive property damages. The collision prevention system was designed to be removable and robust enough for low-speed aerial movements, as shown in Figure 2.10. Nonetheless, the system is highly sparse to allow air to flow freely. The designed system certainly isolates large objects incoming from sides of the vehicle, although it could still collide with small hanging objects during ascending or descending movements. Therefore, the mission operators are obliged to decide, whether it is safe to fly in environments, where this system could not provide sufficient safety guarantee.

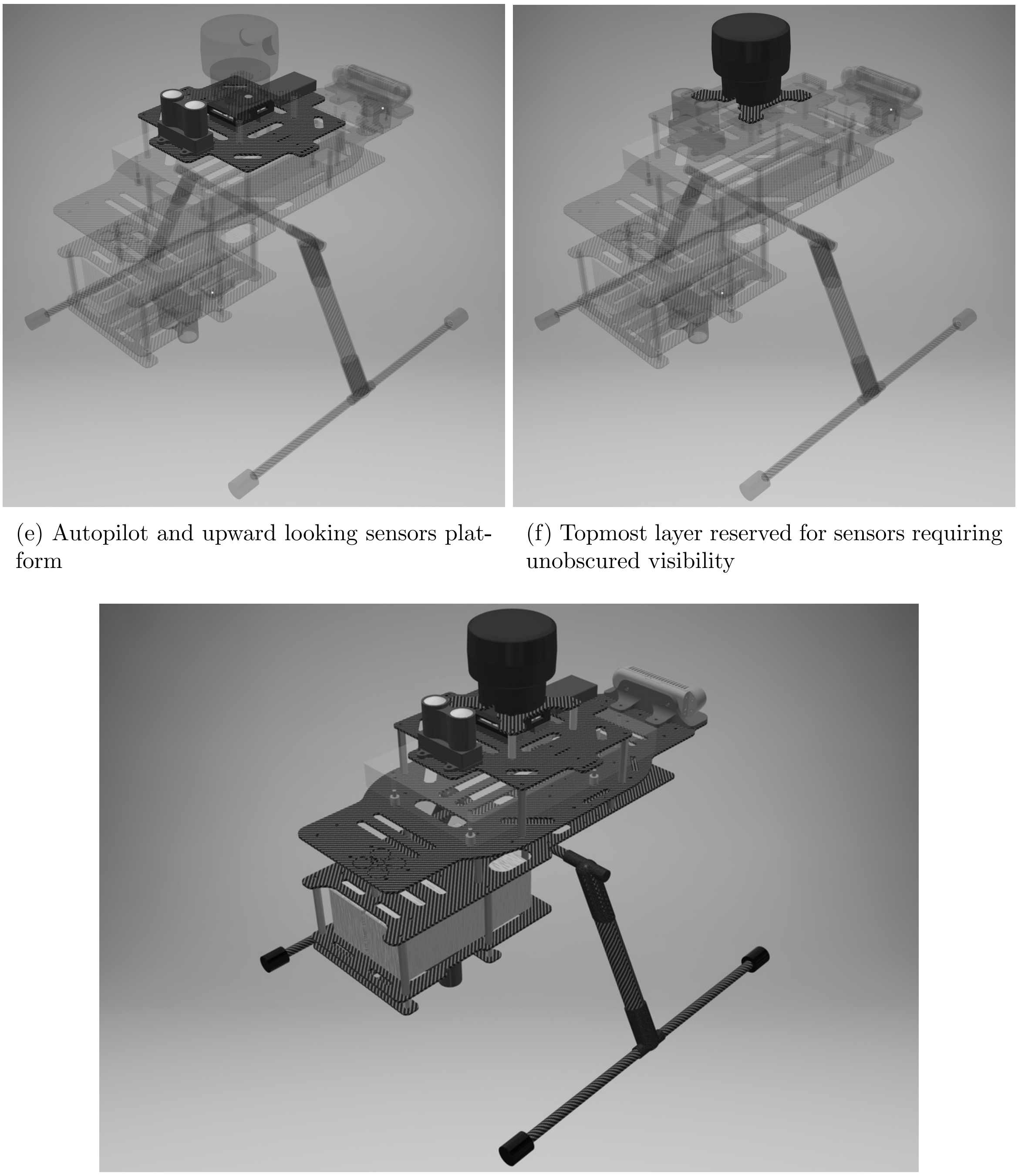


(a) Landing gear and the bottom-most layer for attachment of downward looking sensors

(b) Platform for reliable battery attachment

(c) A location for power distribution management and non-sensory equipment fixation

(d) A layer for central processing unit, radio receiver and fixation of other sensors

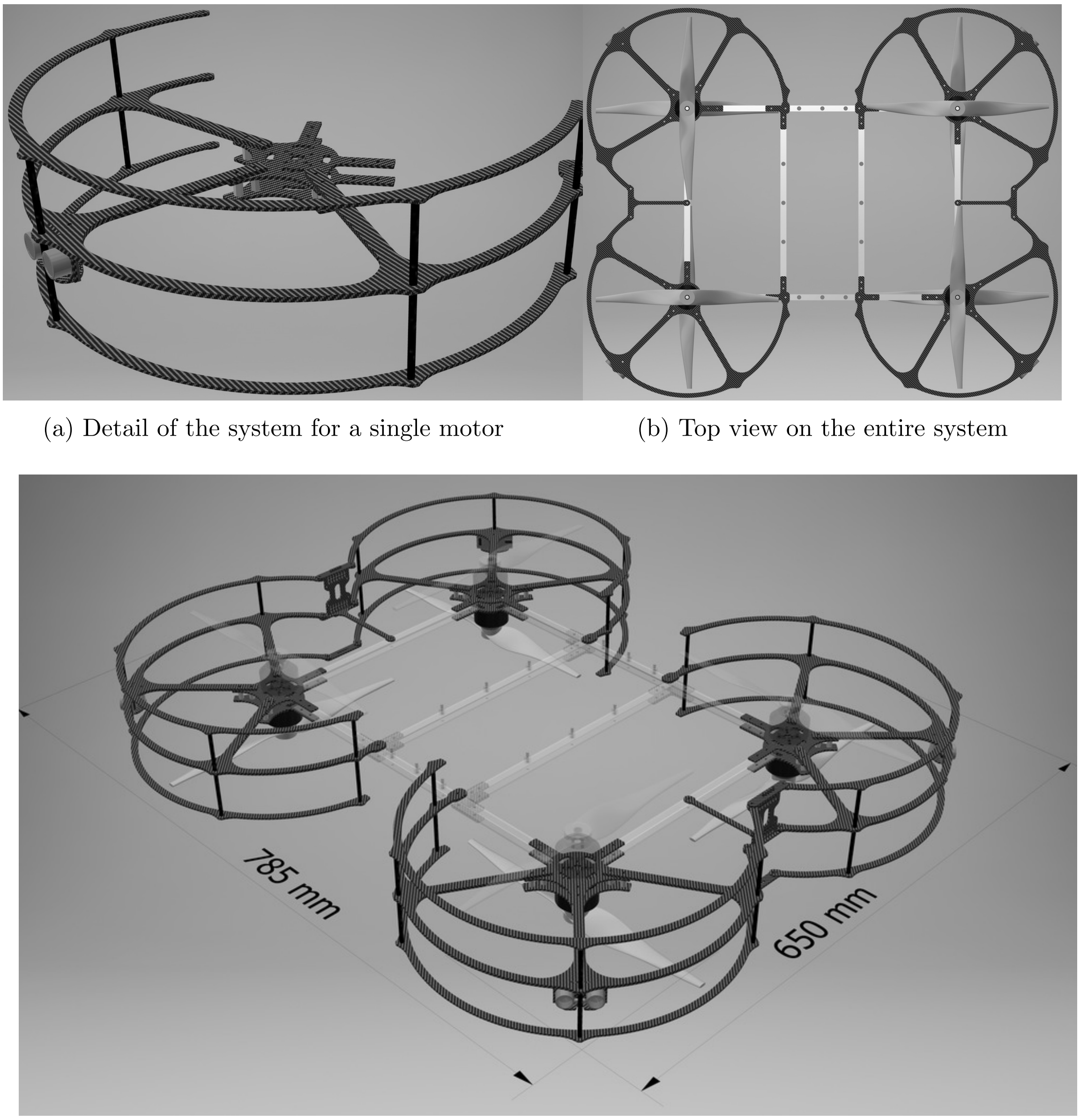


(e) Autopilot and upward looking sensors platform

(f) Topmost layer reserved for sensors requiring unobscured visibility

(g) Aggregated base component with all its layers and a landing gear

Figure 2.9: Visualizations of the individual base component parts

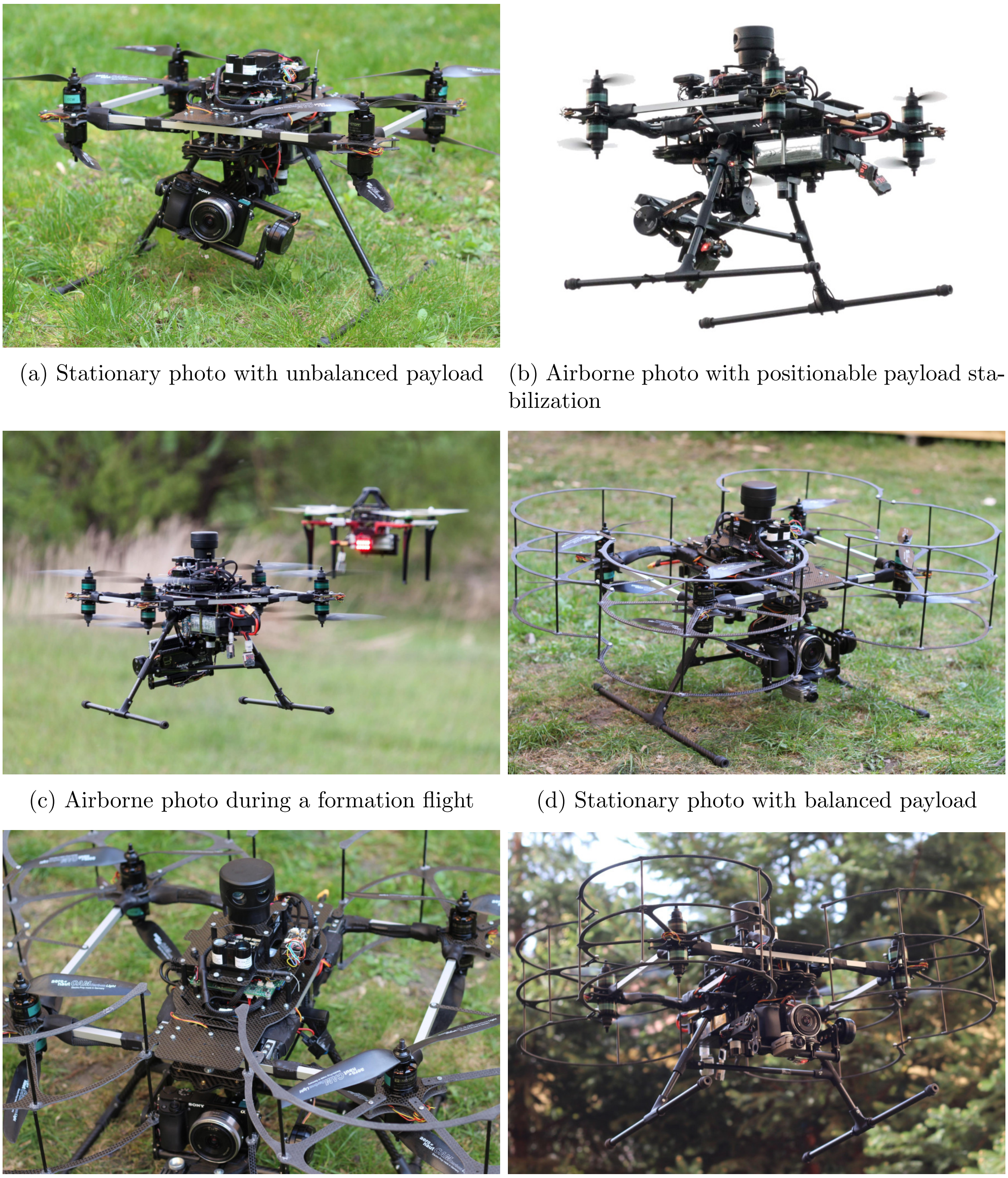


(a) Detail of the system for a single motor

(b) Top view on the entire system

(c) A perspective scene of the aggregated system

Figure 2.10: Visualizations of the collision prevention system



(a) Stationary photo with unbalanced payload

(b) Airborne photo with positionable payload stabilization

(c) Airborne photo during a formation flight

(d) Stationary photo with balanced payload

(e) Detail of the top layer

(f) Airborne photo of the complete system

Figure 2.11: Photos of the finalized platform during a testing flight with and without the collision prevention system, and a stabilized payload of either a camera or a light

Chapter 3: System Architecture

A brief insight into the system's control architecture is presented to empathize a reader with the application-tailored system for documentation of historical monuments. The proposed solution is based on the well-studied hybrid control paradigm of an autonomous system [47]. The paradigm combines hierarchical (deliberative) with the reactive paradigm in a Plan and then Sense-Act way, as illustrated in Figure 3.1.

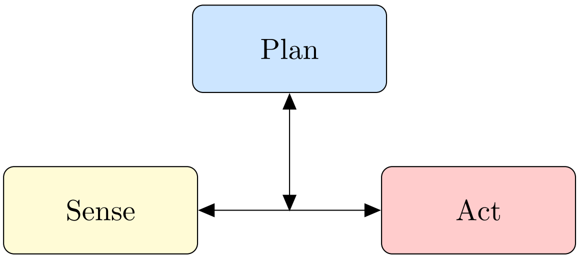


Figure 3.1: Robotic hybrid paradigm

Commonly, robotic system architectures consist of stacked layers (or modules), each providing data or information for layers above, or inducing a system behavior. This approach provides effective development and security performance, which is particularly essential for projects wrapping multiple subsystems. An architecture of the proposed system is described in Figure 3.2, where the general layer architecture is introduced, together with a rough update rate approximation of each particular layer.

Clearly, the proposed system consists of an abundant number of robotic problems. As for the Act system (red color in figures), frequently tested UAV control approach [36, 35] is employed, whose functionality was already proved in harsh outdoor environments [37, 38, 39]. For purposes of the proposed system, the control system is perceived as a black box, which is supplied with vehicle's state estimation and setpoint references, and it produces accurate reference tracking in sufficient finite time.

Furthermore, a proposition of the Plan system is already presented in [11], where the approach for documentation of dark areas of historical buildings using a formation of unmanned aerial vehicles is presented. The Plan system is being developed in parallel by our group and is not further discussed in this thesis.

Additionally, a System Fault Detection is introduced in Figure 3.2. This system, which is extremely crucial for our application with tremendous safety requirements, monitors critical components of the vehicle. That includes monitoring of hardware and likewise the software components - e.g., sensory data acquirement, sensors and algorithms anomalies check, etc. In case of any malfunction, it has the rights to override control layers and initiate an appropriate

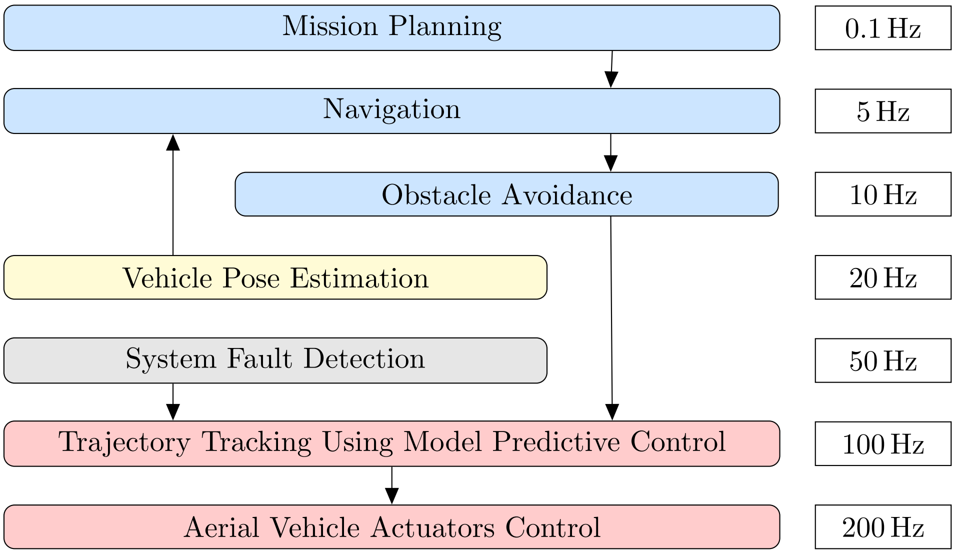


Figure 3.2: General system architecture comprised of dependent layers with an approximation of their update rate

safety procedure. Examples of these safety procedures are immediate landing, return to takeoff location, maintain hovering at the current location or total system shutdown. The System Fault Detection system is likewise being developed in parallel by our group and is not further discussed in this thesis.

In this thesis, the Sense system is described in details. The purpose of this system is the extraction and interpretation of valuable information from raw data taken onboard the vehicle. Its objective is to extract information about the world around the UAV and estimate its state. The proposed methods for the Sense system are described in Chapter 5.

General nature of large buildings that are required to be documented restrain an operation of a global positioning system. Also, presence of practically inaccessible locations prevents usage of any indoor pre-set localization system (e.g., a motion capture system). Therefore, the Sense system is designed specifically for indoor environments. Thoroughly studied approaches for indoor UAV localization are based on onboard cameras, rotational laser scanners or point distance measurement sensors. This equipment leads to usage of variants of simultaneous localization and mapping (SLAM), scan matching or optic-flow mechanisms for reasoning and self-localization in an indoor environment.

However, available vision-based methods become impractical due to their need for suitable lighting conditions, which might be harsh in large historical objects. This disadvantage causes solely vision-based techniques to be non-applicable and further enhancements are required. Furthermore, to overcome a lack of reliability in state-of-the-art localization or pose tracking algorithms employing solely onboard information, the system architecture is enhanced with a concept of a global a priori generated map. A map serves as a backbone supporting localization algorithm by maximizing robustness capabilities of the localization system and heavily decreasing the need for onboard sensing abilities. To further support reliability of the autonomous system demanded by the enormous safety requirements of the application, the maximal velocity of the UAV is limited to $1\,\mathrm{m\,s^{-1}}$, with optimal velocity being even below $0.5\,\mathrm{m\,s^{-1}}$.

In Figure 3.3, high-level system architecture of the proposed approach with the concept of global map is presented. Map generating, utilization and enhancement processes are introduced further in Chapter 4.

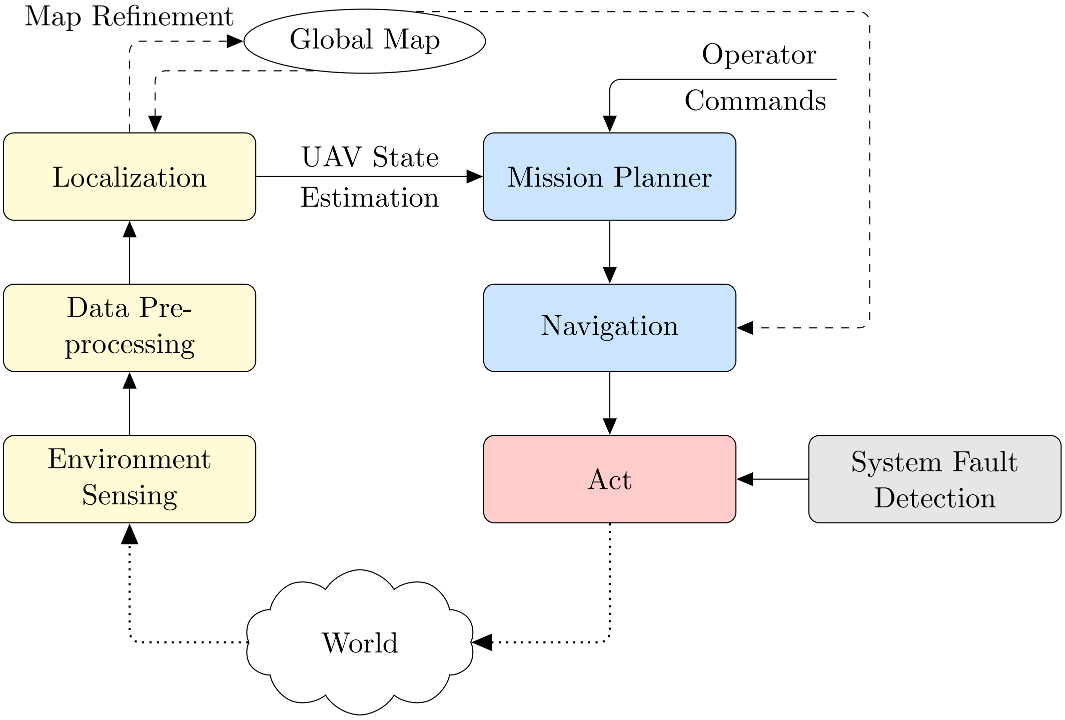


Figure 3.3: System architecture based on a control scheme for a mobile robot

Chapter 4: Global Map

Contents

A map provides a global reference utilized in localization, mapping and navigation modules. Its utilization adds straightforwardness to the robotic problem, supplies additional robustness to the system and supports system reliability. An available global map yields an opportunity to associate captured onboard data with a 3D map, which provides a well-arranged data output to an end-user. For an example of a model of a historical object with photos visually associated with the objects of interest, see Figure 4.1.

In Figure 4.1, a high-level application workflow is presented, where an end-user specifies objects of interest, which are further labeled and applied during mission planning and optimization. Each of the included photographs and snapshots in this figure was taken from ground or airborne locations in Church of St. Mary Magdalene in Chlumín, the Czech Republic, which is introduced in the following sections.

4.1 Map Generation

To generate a map, a reliable device for capturing a 3D map is employed. To accurately scan interiors of historical buildings, a professional 3D scanner is operated. Two professional environment scanners were analyzed on two historical objects.

The first device, Leica Nova MS60 [48], is classified as a multistation. Multistation combines accurate measuring with other Leica technologies, including 3D point cloud generation, imaging, motorization, automatic timing, data storage, and others. This multistation is competent to produce a 3D point cloud including RGB information, intensity and signal-to-noise ratio. The 3D points precision depends on the scanning frequency, where the fastest scan mode (1000 Hz mode) yields 300 m maximal range and 1 mm range accuracy. Furthermore, multiple scans can be generated and then post-processed to generate registered scan of the whole object. However, scanning time of Leica Nova MS60 during a single scan procedure with the fastest scan mode reaches 60 minutes of scan time.

In virtue of these abilities to generate a workable 3D point cloud, which can be utilized as a map, an interior of Church of St. Mary Magdalene in Chlumín, the Czech Republic, was scanned. In consequence of extensive scanning time, only one particular scan of the St. Mary

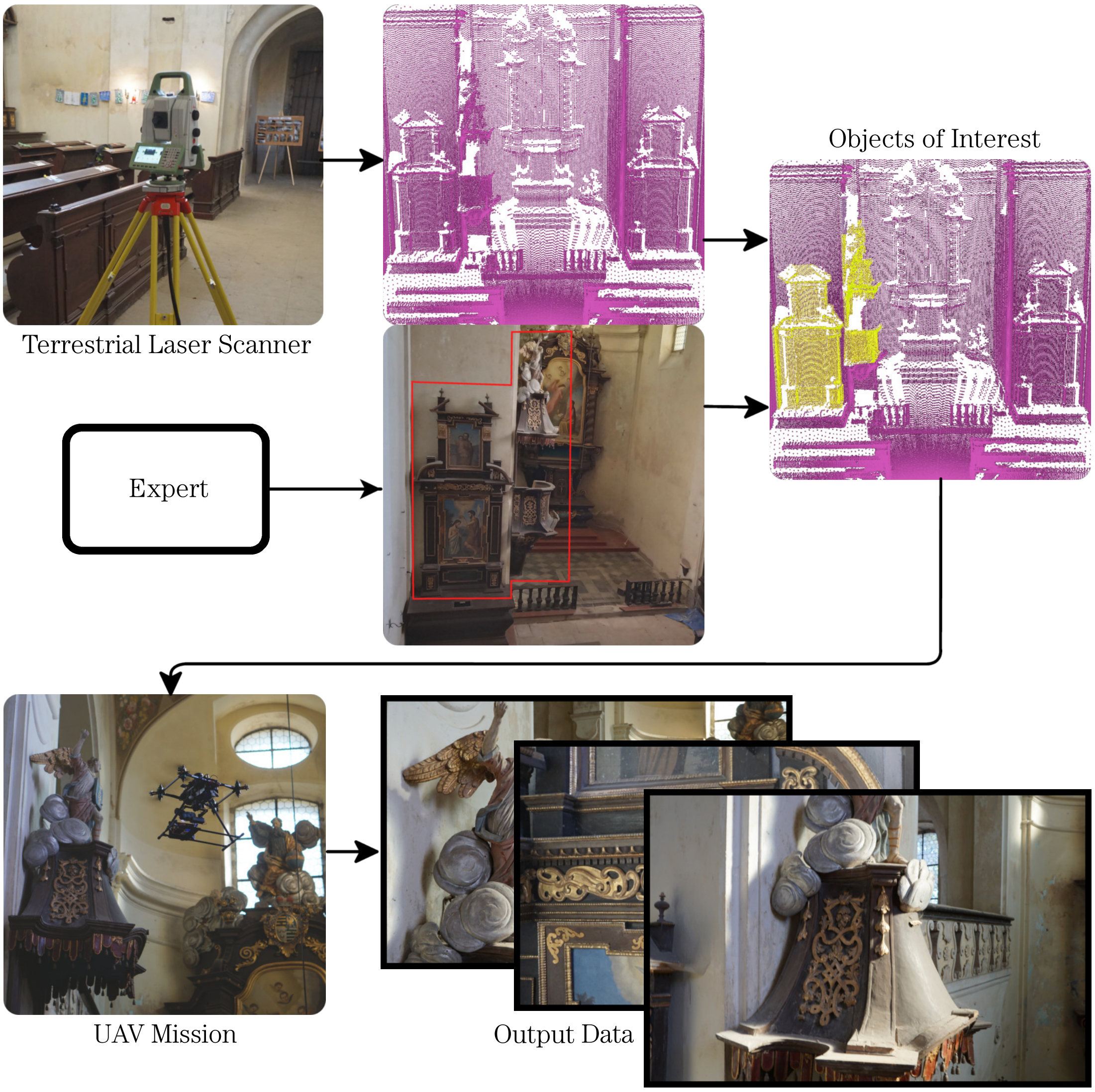
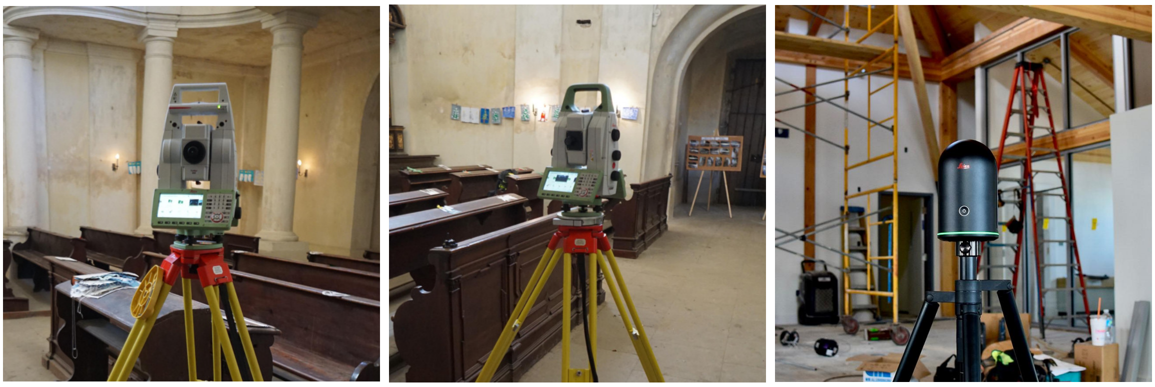


Figure 4.1: Workflow diagram of the entire mission, starting from specification of objects of interest in a scanned map, followed by a UAV or formation of UAVs mission and resulting to a payload data output associated with model of the map.

Magdalene church was taken, and only in XYZ format (no color information to expedite the scanning process). In addition to the 3D scan, a localization dataset with a position ground truth data was taken within this church by multi station Leica Nova MS60 and in parallel by a total station Leica Viva TS16 (shown in Figure 4.2a). Description of the localization dataset is presented in Section 6.3. For these essential assets, Church of St. Mary Magdalene is further used for simulation and real experiments, which are described in Chapter 6. The exterior and interior insights, together with a raw scan produced by Leica MS60 multistation, are presented in Figure 4.3.

The second examined device is a specialized laser scanner Leica BLK360 Imaging Laser Scanner [49]. It captures the environment with RGB-color panoramic images overlaid on a high-accuracy point cloud using 830 nm wavelength laser and distance measurement system based on time of flight principle enhanced by Waveform Digitizing technology. Table 4.1 specifies parameters of the scanner for each of its available scan modes. Its superior advantage lays in scanning time of 3 minutes for a full-dome scan (360° horizontal and 300° vertical field of view) in standard resolution, 150 MP spherical image generation and 60 m maximal range. The compressed time allows generation of scans at various locations and their registration into a single 3D point cloud shortly before deployment of the aerial system. However, the fast scanning time handicaps the 3D point accuracy to 6 mm at 10 m, and 8 mm at 20 m. This accuracy is regarded as sufficient for the robotic application.



(a) Leica Viva TS16 (b) Leica Nova MS60 (c) Leica BLK360 [49]

Figure 4.2: Overview of examined Leica stations and 3D scanners

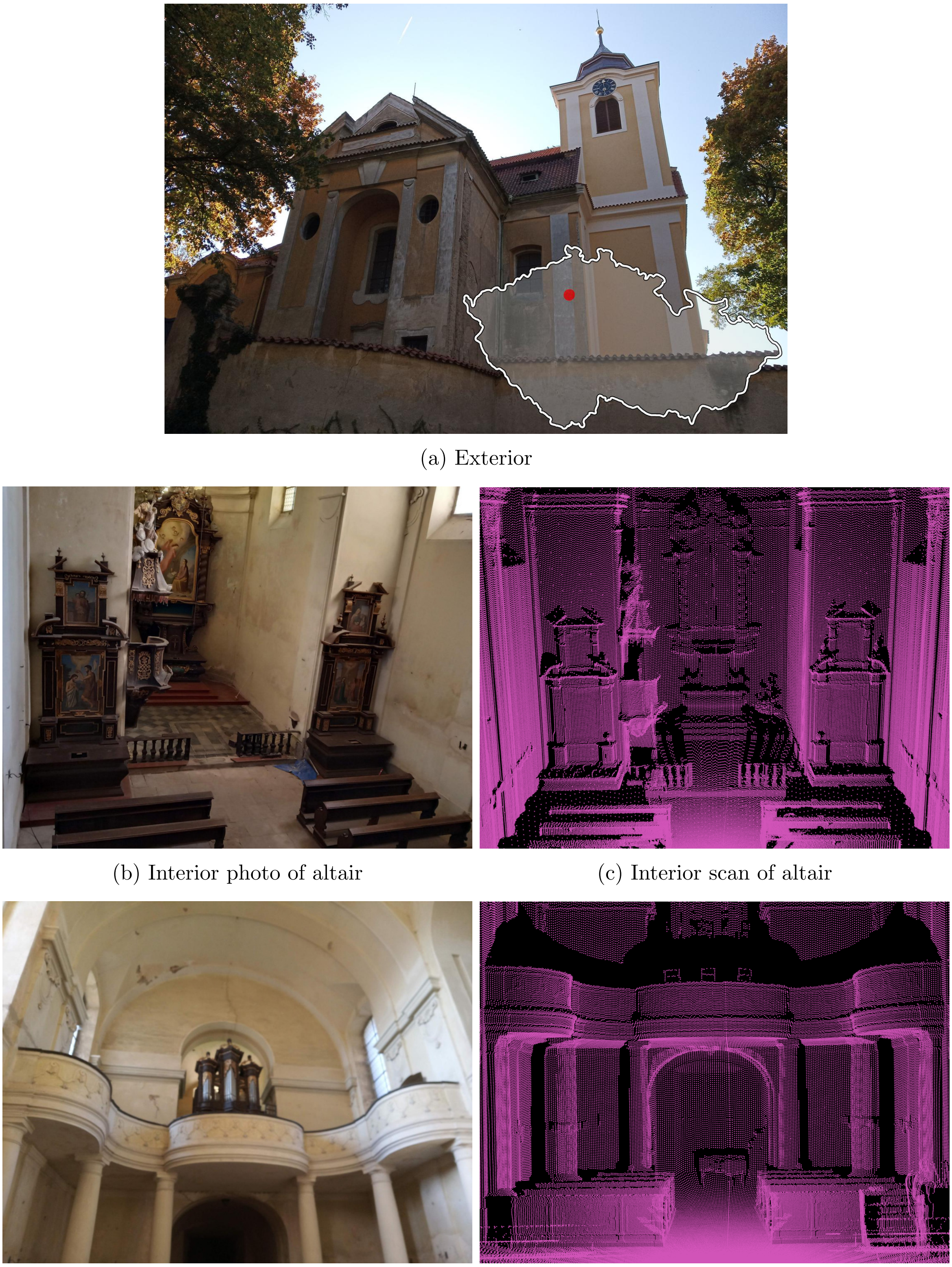
Scan mode	Resolution [mm @ 10 m]	Estimated scan duration [mm : ss]	Approx. scan size [millions of points]
Fast	20	0:40	3
Standard	10	1:50	18
High density	5	3:40	65

Table 4.1: Essential parameters of Leica BLK360 scanner for each particular scan mode

To evaluate the competence of the Leica BLK360 scanner, the interior of Church of St. Wenceslas in Smíchov, Prague, was scanned. The exterior and interior insights, together with colored scan registered from six different church locations by Leica BLK360 scanner, are shown in Figure 4.4. In the output of the Leica BLK360 scanner in Figure 4.4, yellow tetrahedrons represent the six scan locations. Individual scans were registered onto each other in behalf of their mutual overlaps during post-processing[1], and a complete point cloud was created from the collective merge. The resulting point cloud is a collection of 331 millions of RGB-XYZ points, as specified in Table 4.2. The detail of a column capital in Figure 4.4h implies the immense granularity of points.

As a consequence of atrocious color calibration of the scanner during the first two scans, the colors in the figure do not match reality, as a reader can distinguish from a comparison of Figure 4.4b and Figure 4.4e. Point cloud visualization of Church of St. Wenceslas in Figure 4.4 was carried out in JetStream Viewer.

[1] Post-processed in Cyclone REGISTER 360 with floating trial license provided to the MRS group



(a) Exterior

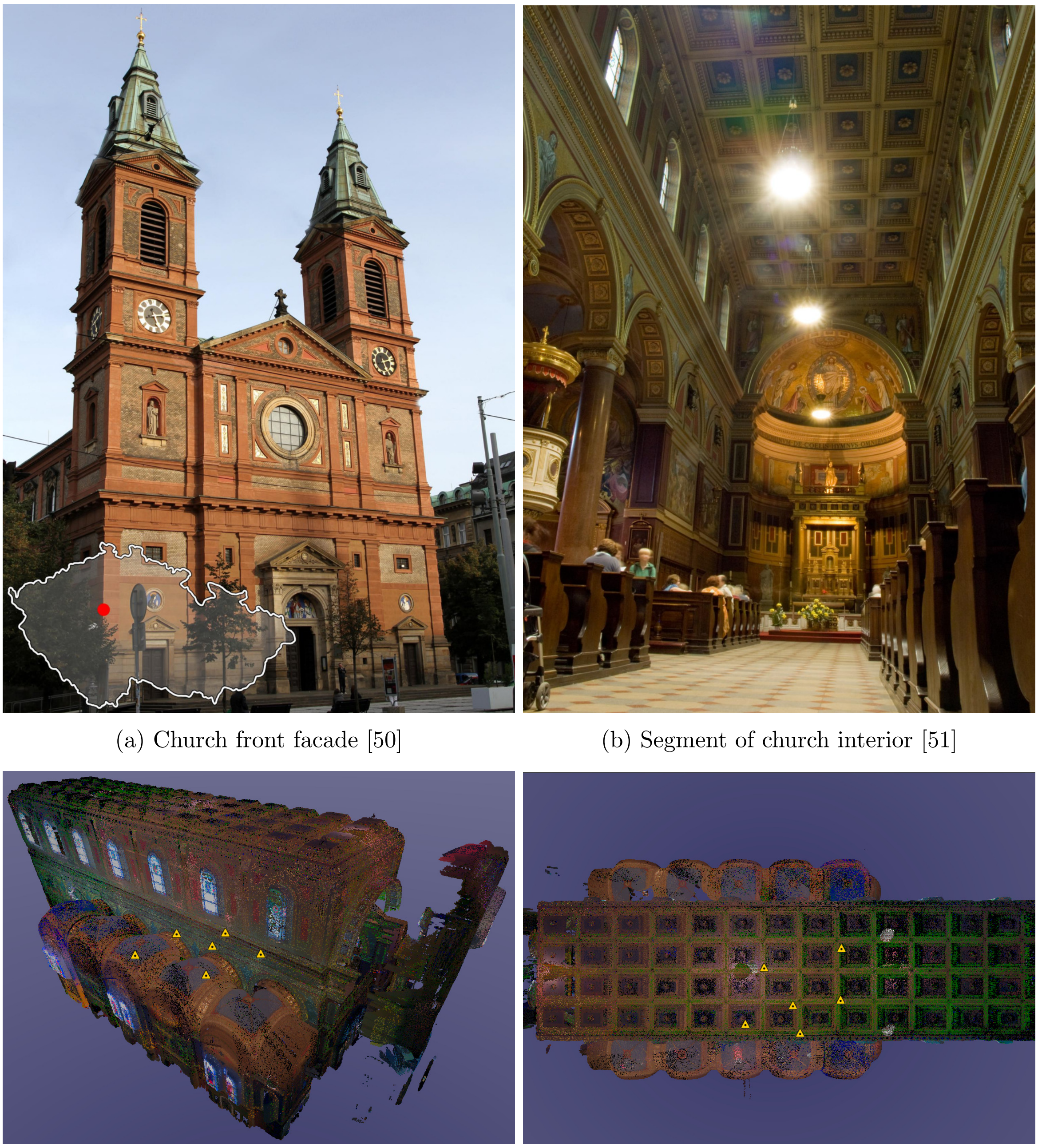
(b) Interior photo of altair

(c) Interior scan of altair

(d) Interior photo of balcony

(e) Interior scan of balcony

Figure 4.3: Church of St. Mary Magdalene in Chlumín, the Czech Republic

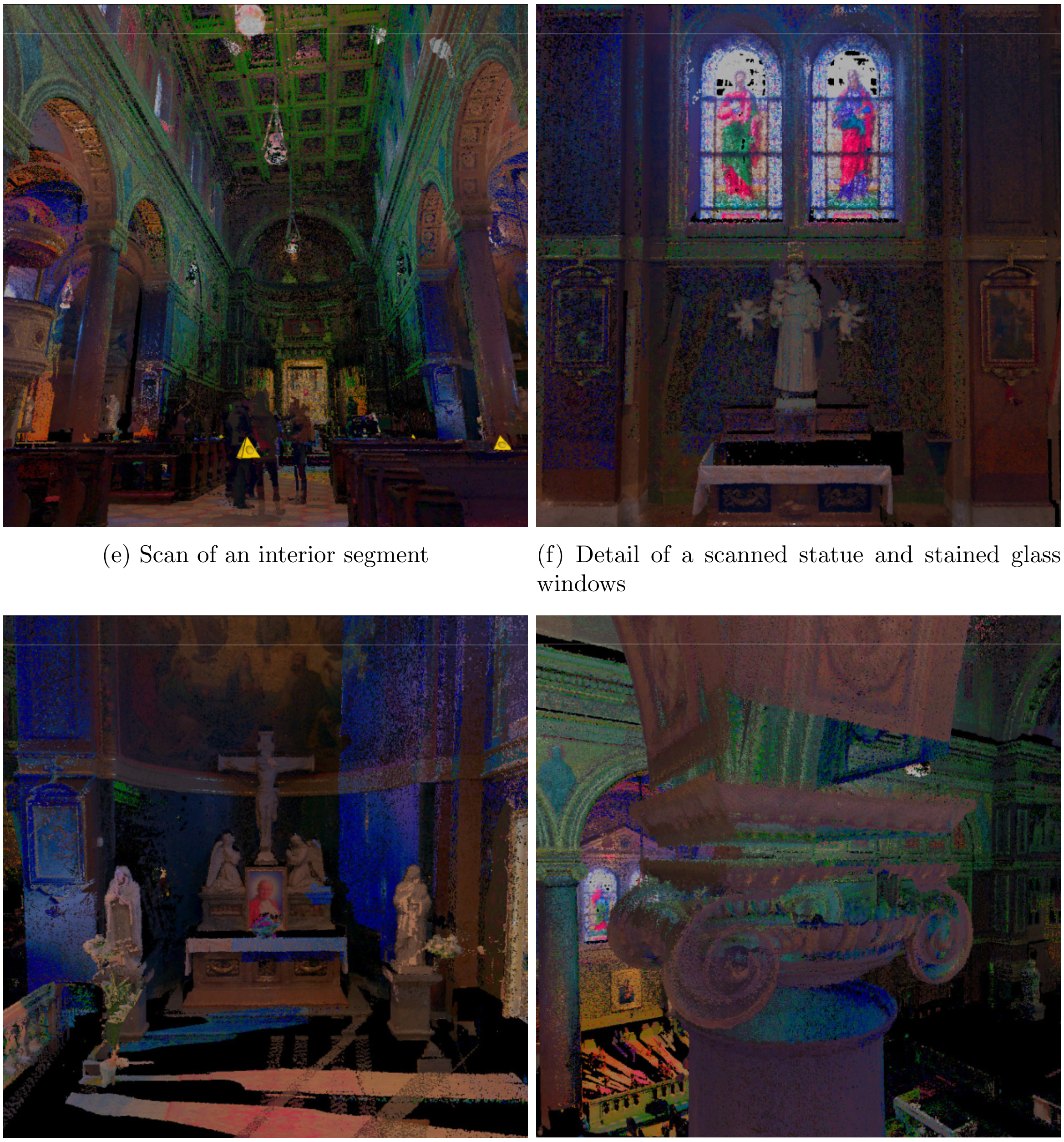


(a) Church front facade [50]

(b) Segment of church interior [51]

(c) 3D view of the registered scan

(d) Top-view on the entire site



(e) Scan of an interior segment

(f) Detail of a scanned statue and stained glass windows

(g) Detail of a scanned remembrance site

(h) Detail of a scanned column capital

Figure 4.4: Church of St. Wenceslas in Smíchov, Prague, the Czech Republic

4.2 Map Interpretation

In the previous section, a map in the concept of excessively dense point clouds was introduced. Further in Chapter 5, an approach for UAV map-based self-localization is presented. The approach disregards excessive details, completely omits available RGB information and excludes endeavor to match high-detail features. For these reasons, a way to efficiently abolish map details, and store and employ a map onboard a UAV is a necessity.

To optimize the efficiency and performance of a map management with respect to computational resources of a UAV, octrees are utilized. Octrees, introduced in [52], provide a hierarchical data structure for spatial subdivision of 3D space. This well-studied tree data structure recursively subdivides 3D space into eight octants down to a defined resolution, as illustrated in Figure 4.5. The octree structure ensures fast map transformations, as well as node traversal to find a subset of voxels in an octree pierced by a directed line [53].

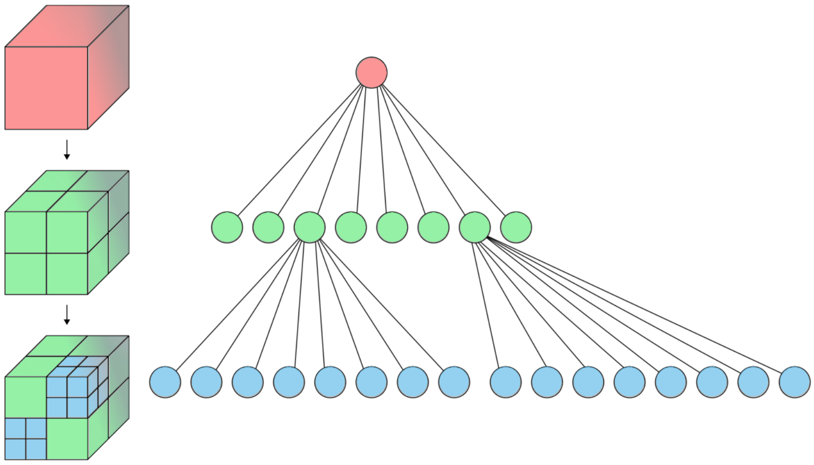
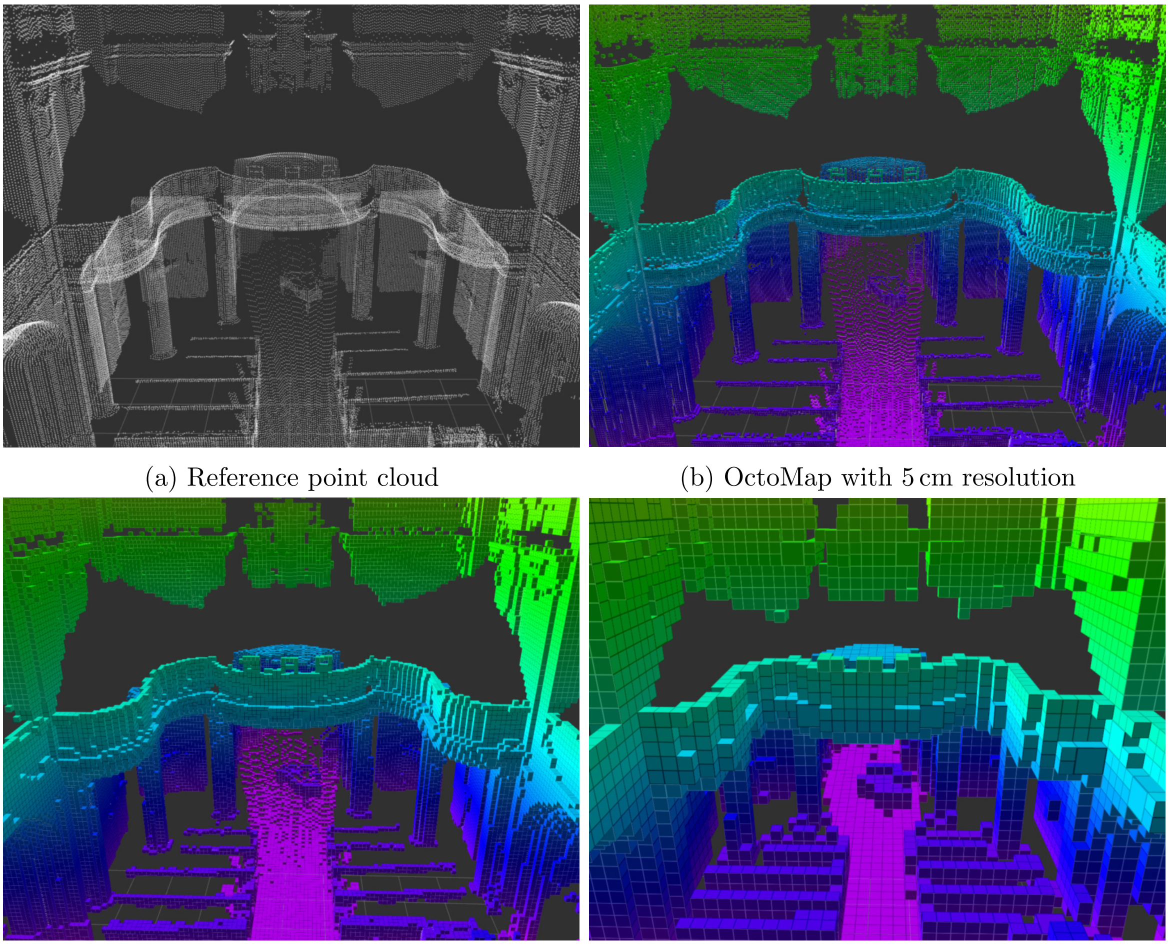


Figure 4.5: Visualization of octree spatial subdivision of 3D space [54]

An efficient open-source OctoMap mapping framework [55] is used to represent a map with inner space representation based on octrees. This implementation extends a regularly used 2D occupancy grid to 3D (hence grid cells convert to voxels), where octree representation provides efficiency, and sparser and downsampled number of points. Having a map represented as an occupancy grid, probabilistic techniques can be utilized to its modeling and updating, where essential requests (sensor measurements integration, data access, collisions evaluation or tree nodes queries) are implemented in an effective way [53, 56]. However, apart from an occupancy status, voxels can be used during navigation and mission planning to store information about the mission itself - i.e., whether a given voxel is part of an object of interest and whether it was already documented.

In the concept of a map-based localization, the choice of map resolution factor needs to respect the performance requirements. Obviously, greater resolution provides larger dropout in octree traversal, searching of nearest neighbors and collision checking performance, since the tree depth is increased. Furthermore, with respect to the localization task, the map resolution is bounded by resolution of onboard sensors used for the task. As presented in Section 2.2.1, the primary sensor is a planar laser scanner. Its range measurement accuracy is not specified by its producer, but its accuracy of ±10 cm at 8 m was derived from real measurement data and demonstrated in Figure 2.6. The accuracy reasons for the choice of map resolution to 10 cm, which disregards the highest indistinguishable and unreliable details captured by the sensor. Influence of a map resolution on OctoMap representation of a balcony in Church of St. Mary Magdalene is presented in Figure 4.6.



(a) Reference point cloud

(b) OctoMap with 5 cm resolution

(c) OctoMap with 10 cm resolution

(d) OctoMap with 30 cm resolution

Figure 4.6: OctoMap representation of Church of St. Mary Magdalene in Chlumín

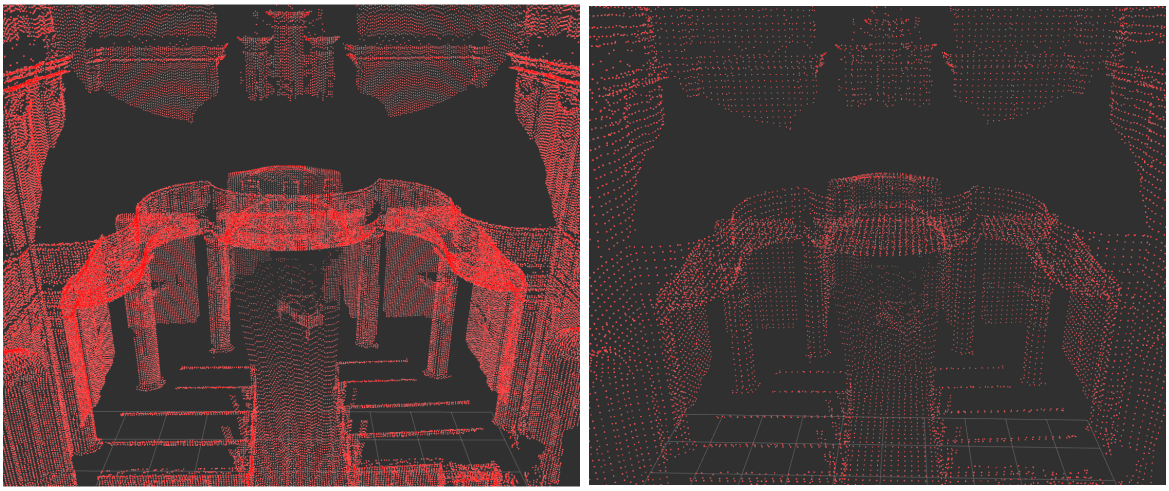
4.3 Map Processing

In order to establish convergence in performance of map operations, the map needs to be processed prior to its utilization. 3D scanner output data volumes are specified in Table 4.2. The data are, especially for the registered scan of Church of St. Wenceslas, intractable to handle. Also, raw scanner data lack information in occluded locations with respect to the scanning positions. These occlusions, observable behind columns or above the balcony in Figure 4.3 or Figure 4.6, cause unpredictable issues during localization task as reference data are missing in these locations. Therefore, in this section, a record of techniques either already utilized or planned to in terms of map pre-processing is presented.

Site	Point Cloud	Octree 5 cm	 10 cm	 30 cm
Church of St. Mary Magdalene	644 159	509 293	179 942	25 057
Church of St. Wenceslas	330 884 107	2 633 991	727 159	82 885

Table 4.2: Number of points in available scan data - comparison between raw point cloud and octree representation with various resolutions

An immense number of points strives for reduction during a robotic task. From a robotic point of view, large point clouds with high points granularity contain redundant information leading to performance dropout. Besides, storage of large point cloud data is memory and time dependent, e.g., 331 millions of XYZ points from Table 4.2 takes 6.2 GB of memory. Therefore, two methods for points reduction using uniform sampling are utilized. The first method naturally emerges from the usage of octrees, where space voxels are created. Each voxel represents all points located inside its retained space defined by an octree resolution, which leads to representation of n points by a single voxel. Additionally, initialization of octrees might consume a considerable amount of time, especially for a vast number of points. To speed up octree initialization, a preliminary binary compression followed by a uniform sampling of raw point clouds is performed. This approach is similar to the natural voxel sampling since points below a specified resolution are disposed of, where a choice of the resolution should be equal or greater than the resolution of the octree. Advantage of a raw point cloud reduction is that it is a one-time action and further repetitive processes already work with the sampled version, which marginally speeds up the application. An example of such process profiting from the reduction of a point cloud is presented in Section 5.3. The uniform sampling of a point cloud is demonstrated in Figure 4.7.



(a) Raw

(b) Uniform sampling with resolution of 10 cm

Figure 4.7: Example of uniform sampling of a point cloud during pre-processing phase

Another attribute of a map is a presence of occluded locations, where map data are missing. These occlusions are particularly present in scans, where the resulting scan was not aggregated from multiple scan locations. Additionally, high-ground locations are often occluded by some high-leveled element of the historical object, i.e., a balcony or an upper floor. Both occlusions can be seen in Figure 4.8, where the scan data were taken from one location. Hence, the locations behind columns, above balconies or behind bench seats lack information. Nonetheless, lack of points can likewise occur at certain surfaces, which do not reflect the measurement laser beams at all or do not reflect enough beam energy back, when a particular angle of impact is exceeded.

A typical example of an ambiguous surface is glass. Laser scans from the ground, as well as the onboard scanner, do not return reliable data from a regular transparent glass. However, as shown in Figure 4.4f, stained glass, repeatedly present in historical objects, reflects beams to be successfully captured by the terrestrial laser scanner. Unfortunately, a testing flight in proximity of the utilized laser scanner and stained glass was not performed yet. Because of that, a conclusion of resemblance between the ground and onboard data over stained glass surface is not presented.

Absence of data in certain map locations creates holes of variable sizes. From a localization point of view, these gaps can create disturbances or even destabilize a state estimation. Therefore, appropriate precautions have to be taken into account, when a robot shall work with data, which are practically absent. As future work, a map enhancement based on probabilistic integration of onboard data into the global map is planned, and therefore to some extent reconstruct and refine the absent map segments.

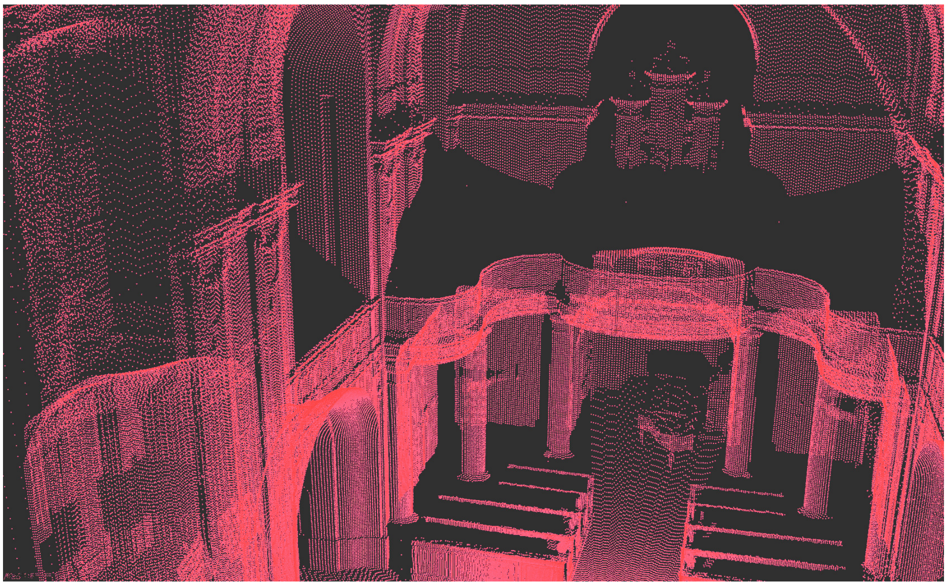
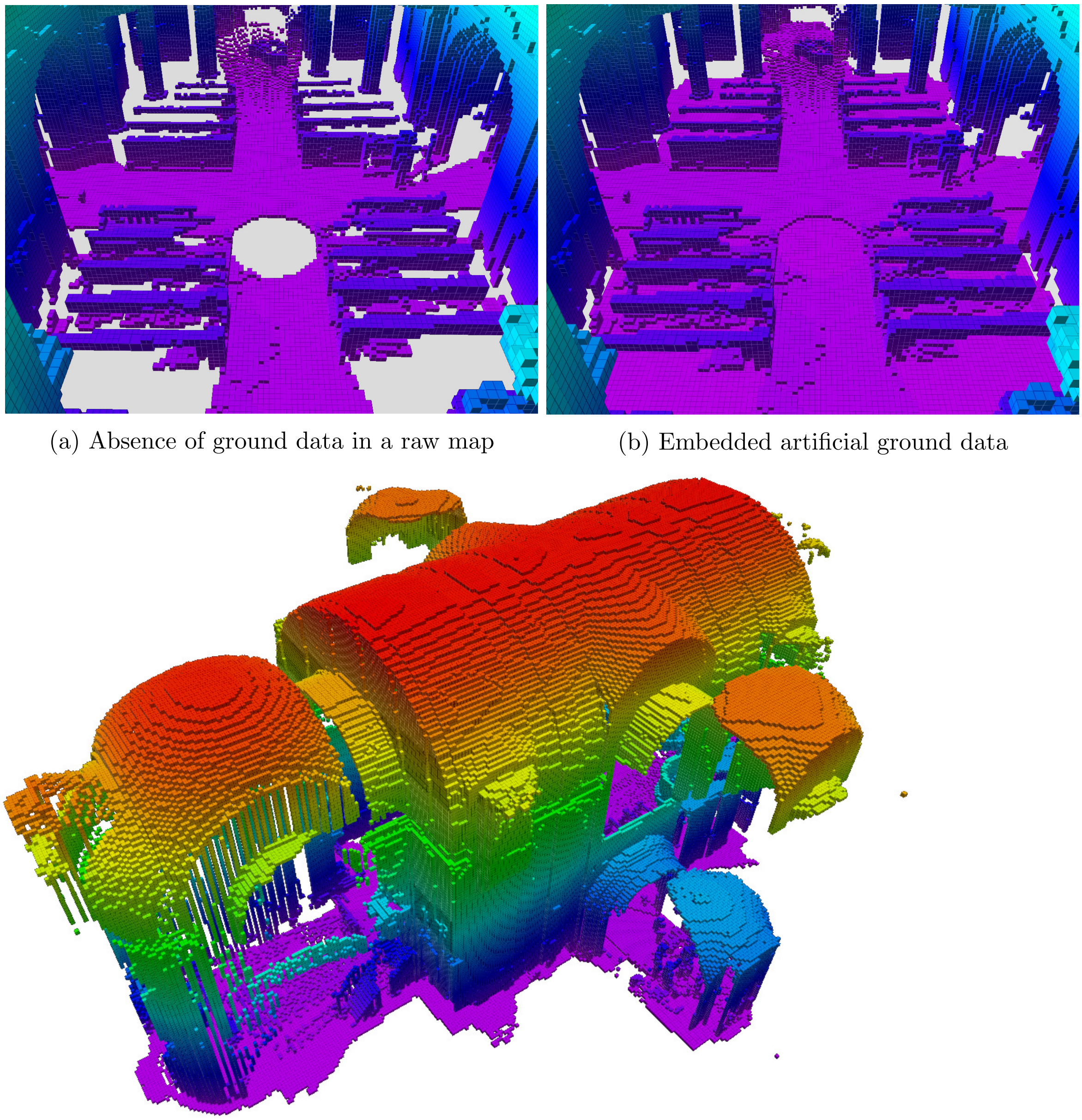


Figure 4.8: Example of missing data (black color) in occluded locations from a single-location scan

Nevertheless, an assumption about absence of data representing ground is already proposed in the presented system, as anticipated in Figure 4.8 or better in Figure 4.6a. These data are essential as a reference for an altitude estimation. Therefore, gaps of ground data are filled by artificial insertion of information, presuming the ground is a plane not containing large and unexpected holes (e.g., stairs down). Gap filling is achieved by adding points uniformly sampled from a plane. The plane parameters are obtained by fitting a plane on a set of points withdrew from the undermost parts of the available map using RANSAC algorithm. Furthermore, the point clouds taken from ground locations share a common feature of incorporating satisfactory information about a ceiling of the scanned objects as a consequence of a clear view of the object scanner. As a consequence, a feature of adding artificial ground data solely at locations allocated under the ceiling is proposed. The final addition of artificial ground data is shown in Figure 4.9, where indoor ground data holes are patched with artificial data.



(a) Absence of ground data in a raw map

(b) Embedded artificial ground data

(c) Church of St. Mary Magdalene in Chlumín with artificial ground data

Figure 4.9: Patching of absent data at foreseen ground locations of a map

Chapter 5: Localization

Contents

5.1 Problem Statement

Online self-localization of a robot is one of the most crucial subsystems of the whole application en route to an autonomous robot. It provides feedback between a robot's representation of a perceptive environment to other subsystems in the form of a state estimate, which we define later in Equation 5.13. The systems relying on the state estimation are mainly robot airborne stabilization, control, navigation, and mission supervision, although map reconstruction or mission evaluation shall likewise include localization history. Table 1.2 contains summary of symbols used throughout this section.

The primary objective of the localization task is to find an affine transformation between map and robot's coordinate system, as embodied in Figure 5.1a. A coordinate system of a map (henceforth *map*) in a matrix form is defined as

$$\mathbf{O}_{map} = \left[\begin{array}{c|c} \mathbf{R}_{map} & \vec{t}_{map} \\ \hline 0 & 1 \end{array}\right], \tag{5.1}$$

and a robot coordinate system being in center of gravity of the robot's flight control unit (henceforth *fcu*) as

$$\mathbf{O}_{fcu} = \left[\begin{array}{c|c} \mathbf{R}_{fcu} & \vec{t}_{fcu} \\ \hline 0 & 1 \end{array}\right]. \tag{5.2}$$

Besides, for a known static $\mathbf{O}_{map}$ and an unknown dynamic $\mathbf{O}_{fcu}$, their mutual relation

$$\mathbf{O}_{fcu} = \mathbf{T}_{fcu,map}\,\mathbf{O}_{map} \tag{5.3}$$

is to be determined, where transformation $\mathbf{T}_{fcu,map}$ is given as

$$\mathbf{T}_{fcu,map} = \left[\begin{array}{c|c} \mathbf{R}_{fcu,map} & \vec{t}_{fcu,map} \\ \hline 0 & 1 \end{array}\right], \tag{5.4}$$

where $\mathbf{R}_{fcu,map} \in \mathbb{R}^{3\times3}$ is the orientation matrix and $\vec{t}_{fcu,map} \in \mathbb{R}^{3\times1}$ is the translation vector of the *fcu* coordinate system with respect to the *map* coordinate system, both in 3D space.

The map coordinate system $\mathbf{O}_{map}$ represents a global coordinate system. Therefore, with anticipation of

$$\mathbf{O}_{map} = \mathbf{I}_{4\times4}, \tag{5.5}$$

where $\mathbf{I}_{4\times4}$ is an identity matrix in order to match origin and orientation of the global coordinate system with *map*, Equation 5.3 can be reformulated to

$$\mathbf{O}_{fcu} = \mathbf{T}_{fcu,map}\,\mathbf{I} = \mathbf{T}_{fcu} = \left[\begin{array}{c|c} \mathbf{R}_{fcu} & \vec{t}_{fcu} \\ \hline 0 & 1 \end{array}\right]. \tag{5.6}$$

The orientation matrix $\mathbf{R}_{fcu}$ represents a rotation of the robot around its x, y and z axes in this particular order. Hence, its thorough description leads to

$$\mathbf{R}_{fcu}(\Psi, \Theta, \Phi) = \mathbf{R}_z(\Psi)\,\mathbf{R}_y(\Theta)\,\mathbf{R}_x(\Phi), \tag{5.7}$$

where Ψ, Θ, Φ are yaw, pitch and roll of the robot (referred to as rotations around an aircraft principal axes), and

$$\begin{aligned} \mathbf{R}_z(\Psi) &= \begin{bmatrix} \cos(\Psi) & -\sin(\Psi) & 0 \\ \sin(\Psi) & \cos(\Psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \\ \mathbf{R}_y(\Theta) &= \begin{bmatrix} \cos(\Theta) & 0 & \sin(\Theta) \\ 0 & 1 & 0 \\ -\sin(\Theta) & 0 & \cos(\Theta) \end{bmatrix}, \\ \mathbf{R}_x(\Phi) &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\Phi) & -\sin(\Phi) \\ 0 & \sin(\Phi) & \cos(\Phi) \end{bmatrix}. \end{aligned} \tag{5.8}$$

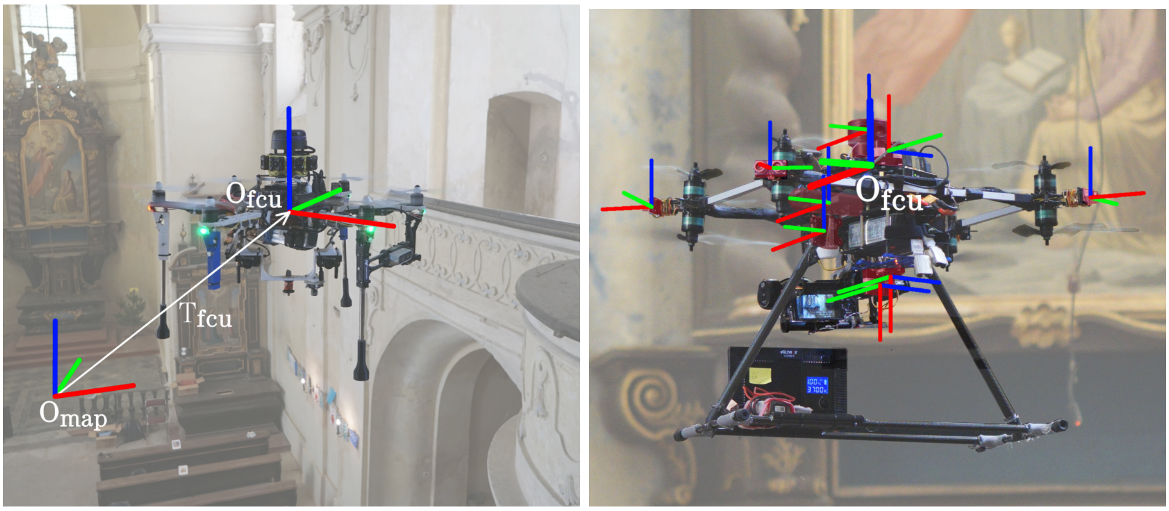
To summarize the objective of the localization task, a transformation defined in Equation 5.6 needs to be found by determining an orientation of the robot described by matrix $\mathbf{R}_{fcu}$ and position of the robot described by translation vector $\vec{t}_{fcu}$. From this point forward, lower index description of variables associated with the description of the robot's coordinate system *fcu* is ignored, which is an utmost subject of examination. Also, all variables are to be defined with respect to a global *map* coordinate system, unless specified otherwise.

Let us define formally concepts of robot pose, position, velocity, and state with regards to discrete time k. The pose of a UAV is defined as

$$\vec{q}_k = \begin{pmatrix} \vec{p}_k \\ \vec{\Omega}_k \end{pmatrix}, \tag{5.9}$$

where

$$\vec{p}_k = \begin{pmatrix} x_k \\ y_k \\ z_k \end{pmatrix}, \quad \vec{\Omega}_k = \begin{pmatrix} \Phi_k \\ \Theta_k \\ \Psi_k \end{pmatrix} \tag{5.10}$$



(a) Objective of the localization task - determining a transformation T_{fcu} between a robot and a map coordination system

(b) Static coordinate systems of onboard sensors depicted in Figure 2.7 with respect to robot coordination system O_{fcu}

Figure 5.1: Essential dynamic and rigid transformations (*rgb* colors specify *xyz* axes in this particular order) between a map, a robot and onboard sensors in a localization task

represent UAV position $\vec{p}_k$ and orientation vector $\vec{\Omega}_k$ at time step k. In matrix representation, pose of a robot is also given by Equation 5.6. Velocities of the UAV are defined as

$$\vec{v}_k = \begin{pmatrix} v_k^x \\ v_k^y \\ v_k^z \end{pmatrix} , \quad \vec{\omega}_k = \begin{pmatrix} \omega_k^x \\ \omega_k^y \\ \omega_k^z \end{pmatrix} , \tag{5.11}$$

where $\vec{v}_k$ represents linear and $\vec{\omega}_k$ angular velocity at time step k.

Moreover, the objective of the localization task is to observe state of the UAV, which is generally defined as the pose of the UAV. As aforementioned in Chapter 3, due to critical safety requirements of the application, the total velocity of a UAV during the mission is limited to maximum of $1\,\mathrm{m\,s^{-1}}$. Based on this velocity limitations, the vehicle dynamics suppress steep angular deviations of roll and pitch angles, hence

$$\Phi_k \approx \Theta_k \approx 0, \quad \forall k. \tag{5.12}$$

Because of that, pose angles roll Φ and pitch Θ are neglected in the localization task. State of the robot at time step k is then defined as

$$\vec{x}_k = \begin{pmatrix} x_k \\ y_k \\ z_k \\ \Psi_k \end{pmatrix} . \tag{5.13}$$

Taking into account this assumption, Equation 5.7 is simplified to

$$\mathbf{R}_{fcu,k}(\Psi_k, \Theta_k, \Phi_k) \approx \mathbf{R}_z(\Psi_k)\,\mathbf{R}_y(0)\,\mathbf{R}_x(0) = \mathbf{R}_z(\Psi_k). \tag{5.14}$$

To find transformation introduced in Equation 5.6, coordinate systems of all the sensors utilized for the localization task have to be introduced. To maintain integrity, any dynamical influences of the UAV frame and sensors attachments during a flight are disregarded. Therefore, the sensors' coordinate systems are considered to be rigidly fixed to the *fcu* system. Under this assumption, the static transformations between sensors to *fcu* are associated with known UAV design prior a flight, as illustrated in Figure 5.1b.

The proposed localization approach utilizes fewer sensors than is available onboard. From the onboard sensors, described thoroughly in Section 2.2.1, vision-based and ultrasonic sensors are excluded. Therefore, solely up- and down-oriented laser rangefinders measuring a distance to the ground and ceiling, and a planar rotational laser scanner are utilized. The main reason for the utilization of a subset of sensors is to maintain initial simplicity of the system. Integration, fusion or refinement of state estimates using the excluded sensors remains a part of future work.

The following sections contain a description of the proposed system for solving the localization task, divided into two dependent subsystems. In Section 5.2, Equation 5.6 is determined in an enormous configuration space of a historical object by utilizing a global localization approach. In other words, essentially the kidnapped robot problem [57] is solved. An approach to locally refine the global approach estimate with a fast scan matching technique is introduced in Section 5.3. And finally, an approach to fuse these two localization estimates into a single final and reliable output is presented in Section 5.4.

5.2 Monte Carlo Localization

The configuration space of a robot inside a priori known map of a historical object is immense. That restricts straight registration of sensory data to the extensive map due to unknown initial conditions. To overcome that, a Monte Carlo Localization approach [58] is utilized to globally determine state of the robot and therefore solve the kidnapped robot problem.

Monte Carlo Localization, also known as particle filter localization, is a recursive and non-parametric Bayesian filter with linear time complexity. The non-parametricity specifies independence on probability distribution assumptions and provides an ability to approximate different types of probability distributions, including multi-modal distributions. Instead of describing the probability density function of robot states in a configuration space, it holds a set of randomly drawn samples from the probability density function itself. From this point forward, samples are addressed as particles or hypotheses, and sensor data as observations. MCL proceeds in two phases - prediction and correction. The current robot state is predicted according to the probabilistic motion of the robot. In the second phase, a set of hypotheses is updated according to sensor observations, followed by resampling of the probability density function.

To formally define the task, a posterior probability is required to be determined in every time step k as

$$p(\vec{x}_k|\vec{y}_{1:k}, \vec{u}_{1:k}), \tag{5.15}$$

where an unobservable state of the robot $\vec{x}_k$ is estimated at time step k given all the observations $\vec{y}_{1:k}$ and system control inputs $\vec{u}_{1:k}$ from the initial to the current time step k.

Computing this posterior probability on the state space subset in form of hypotheses yields an approximation of states probability distribution function.

A solution to the previous equation can be obtained by applying Bayes filter [59], which recursively computes the posterior probability as

$$p(\vec{x}_k|\vec{y}_k, \vec{u}_k) = \eta \, p(\vec{y}_k|\vec{x}_k) \int p(\vec{x}_k|\vec{x}_{k-1}, \vec{u}_k) \, p(\vec{x}_{k-1}|\vec{y}_{k-1}, \vec{u}_{k-1}) \, d\vec{x}_{k-1}, \tag{5.16}$$

where η is a normalization constant. The equation derivation holds under an initial condition $p(\vec{x}_0) = p(\vec{x}_0|\vec{y}_0, \vec{u}_0)$ and following Markov assumptions [60]:

- current state $\vec{x}_k$ is only dependent on the previous state $\vec{x}_{k-1}$ and a known control input $\vec{u}_k$

$$p(\vec{x}_k|\vec{y}_{1:k}, \vec{u}_{1:k}) = \eta \, p(\vec{y}_k|\vec{x}_k, \vec{y}_{1:k-1}, \vec{u}_{1:k}) \, \boldsymbol{p(\vec{x}_k|\vec{y}_{1:k-1}, \vec{u}_{1:k})}, \tag{5.17}$$

$$p(\vec{x}_k|\vec{y}_{1:k-1}, \vec{u}_{1:k}) = \int \boldsymbol{p(\vec{x}_k|\vec{x}_{k-1}, \vec{y}_{1:k-1}, \vec{u}_{1:k})} \, p(\vec{x}_{k-1}|\vec{y}_{1:k-1}, \vec{u}_{1:k}) \, dx_{k-1}, \tag{5.18}$$

$$p(\vec{x}_k|\vec{x}_{k-1}, \vec{y}_{1:k-1}, \vec{u}_{1:k}) = \boldsymbol{p(\vec{x}_k|\vec{x}_{k-1}, \vec{u}_k)}, \tag{5.19}$$

- and current observation $\vec{y}_k$ is conditionally independent of all previous measurements $\vec{y}_{1:k-1}$, on previous states $\vec{x}_{1:k-1}$ and control inputs $\vec{u}_{1:k}$

$$p(\vec{y}_k|\vec{x}_k, \vec{y}_{1:k-1}, \vec{u}_{1:k}) = \boldsymbol{p(\vec{y}_k|\vec{x}_k)}. \tag{5.20}$$

More detailed derivation of Bayes filter and its utilization for MCL can be found in [58, 61, 60].

Let us describe Equation 5.16 more thoroughly. Normalization constant

$$\eta = \frac{1}{\int p(\vec{y}_k|\vec{x}_k, \vec{y}_{1:k-1}, \vec{u}_{1:k}) \, p(\vec{x}_k|\vec{y}_{1:k-1}, \vec{u}_{1:k}) \, d\vec{x}_k} \tag{5.21}$$

makes the posterior density integrate to one. Equation 5.20 represents a conditional probability of an observation $\vec{y}_k$ given a robot state $\vec{x}_k$, commonly noted as an observation model, which is described further in Section 5.2.2. Next, Equation 5.19 represents a conditional probability of a state $\vec{x}_k$ given a previous state $\vec{x}_{k-1}$ and a control input $\vec{u}_k$, commonly noted as a motion model, which is described thoroughly in Section 5.2.1. Finally, the last element of the integral in Equation 5.16 is a recursive element of the same equation for the previous time step $k-1$. This element provides recursive incorporation of states $\vec{x}_{1:k}$, inputs $\vec{u}_{1:k}$ and observations $\vec{y}_{1:k}$.

An overview pseudocode of the MCL algorithm is presented in Algorithm 1, partially based on [58]. Sampling techniques employed on lines 1, 8 and 9 are described in Section 5.2.3. A motion model on lines 3 and 5 is described in Section 5.2.1 and an observation model on line 6 in Section 5.2.2.

5.2.1 Motion Model

Odometry based motion model, typically described in 2D [62], is employed. However, the motion model is extended to 3D space, similarly as in [63], to match the operational space.

Algorithm 1 Monte Carlo Localization

```
H = generate_M_random_hypotheses()
while true do
    u = get_input()
    for h in H do
        h.state = motion_model(h.state, u)                ▷ Prediction step
        h.weight = observation_model(h.state)             ▷ Correction step
    end for
    M = estimate_sufficient_M()                           ▷ Importance sampling
    H = resample_hypotheses(M, H)                         ▷ Resampling step
end while
```

Commonly, an odometry based motion model is based on inner odometry (i.e., wheel encoders in case of a ground robot). However, a reliable source of odometry information is not available in the proposed UAV system. Therefore, a concept of dead reckoning utilizing linear and angular velocities is imposed, where the velocities are provided by the flight controller unit in its own coordinate system. An inner IMU of the autopilot provides linear accelerations and angular velocities, where the linear accelerations are integrated to linear velocities. As aforementioned, the Pixhawk Cube autopilot contains 3 independent IMUs, specifically InvenSense MPU9250, ICM20948 and/or ICM20648 as first and second IMU, and ST Micro L3GD20+LSM303D or InvenSense ICM2076XX as a backup IMU. The autopilot fuses the three IMU data into one and integrates the accelerations to velocities and hence provides immediate access to synchronized linear and angular velocities. The dead reckoning principle then integrates provided velocities to estimate the pose of the robot. This pose estimate is used to relatively move each one of the hypotheses according to the UAV kinematic model, exhibited in Figure 5.2.

An odometry based model requires an input in the form of an odometry information. While the dead reckoning principle is commonly utilized with a velocity based motion model, the proposed solution employs it as a noisy odometry, because of a vast difference between velocity and MCL algorithm update rates. The common update rate for an IMU is approx. 120 Hz, while motion model is called approx. at 5 Hz. Hence, the dead reckoning integrates velocities in the background and provides odometry whenever outer systems require it. Therefore in MCL, relative pose change between two iterations given by the dead reckoning is considered as a piece of odometry information.

In comparison with [63], the motion model neglects variations in roll and pitch and therefore reduces kinematic degrees of freedom to 4. Additionally, as introduced Equation 5.30, a noise in the heading change $\Delta\Psi_k$ is accounted for. The 4 degrees of freedom kinematic model of the UAV is illustrated in Figure 5.2. Given an input odometry at time step k

$$\vec{u}_k = \vec{q}_k - \vec{q}_{k-1} = \begin{pmatrix} \Delta x_k & \Delta y_k & \Delta z_k & \Delta\Phi_k & \Delta\Theta_k & \Delta\Psi_k \end{pmatrix}^T, \tag{5.22}$$

other relations of the kinematic model are formulated as

$$\alpha_k = arctan\left(\frac{\Delta y_k}{\Delta x_k}\right), \tag{5.23}$$

$$\beta_k = arctan\left(\frac{\Delta z_k}{\sqrt{(\Delta x_k)^2 + (\Delta y_k)^2}}\right), \tag{5.24}$$

$$\Delta t_k = \sqrt{(\Delta x_k)^2 + (\Delta y_k)^2 + (\Delta z_k)^2}, \tag{5.25}$$

and the kinematic model itself

$$\vec{x}_k = \begin{pmatrix} x_{k-1} + \Delta t_k \cos(\beta_k)\cos(\alpha_k) \\ y_{k-1} + \Delta t_k \cos(\beta_k)\sin(\alpha_k) \\ z_{k-1} + \Delta t_k \sin(\beta_k) \\ \Psi_{k-1} + \Delta\Psi_k \end{pmatrix}. \tag{5.26}$$

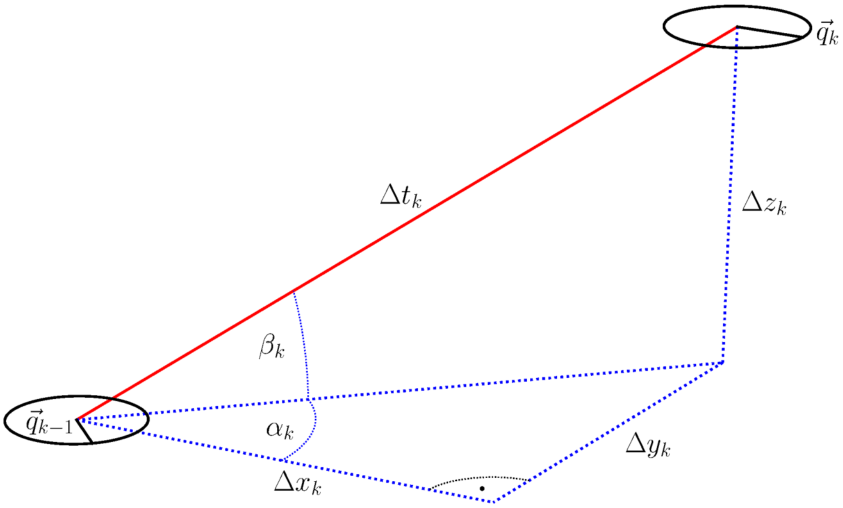


Figure 5.2: Odometry based motion model in 3D space based on [63]

However, the velocities used for dead reckoning, and the UAV movement and control are fundamentally inaccurate. Because of these inaccuracies, the presented kinematic model shatters and in practice does not hold. In order to account for these unknown noise errors, we introduce to the kinematic model artificial zero-mean Gaussian noises with standard deviations

$$\sigma_{\alpha_k} = \epsilon_0\alpha_k + \epsilon_1\Delta t_k, \tag{5.27}$$

$$\sigma_{\beta_k} = \epsilon_2\Delta z_k, \tag{5.28}$$

$$\sigma_{\Delta t_k} = \epsilon_3\Delta t_k + \epsilon_4\Delta\Psi_k, \tag{5.29}$$

$$\sigma_{\Delta\Psi_k} = \epsilon_5\Delta t_k + \epsilon_6\Delta\Psi_k, \tag{5.30}$$

where $\epsilon_{0:6}$ represent amount of noise particular parameters deliver to the system. Appropriate choice of these constants is essential in order for the motion model to distribute hypotheses according to the authentic motion of the robot. Its importance is described in [62], which is likewise used to empirically set these constants in our implementation.

With definition of a function *sample*(σ), which draws a random sample from Gaussian distribution $\mathcal{N}(0, \sigma)$, the motion model is finally defined by equation

$$\vec{x}_k = \begin{pmatrix} x_{k-1} + \hat{t}_k \cos(\hat{\beta}_k) \cos(\hat{\alpha}_k) \\ y_{k-1} + \hat{t}_k \cos(\hat{\beta}_k) \sin(\hat{\alpha}_k) \\ z_{k-1} + \hat{t}_k \sin(\hat{\beta}_k) \\ \Psi_{k-1} + \hat{\Psi}_k \end{pmatrix}, \tag{5.31}$$

where

$$\hat{\alpha}_k = \alpha_k + sample(\sigma_{\alpha_k}), \tag{5.32}$$

$$\hat{\beta}_k = \beta_k + sample(\sigma_{\beta k}), \tag{5.33}$$

$$\hat{t}_k = \Delta t_k + sample(\sigma_{t_k}), \tag{5.34}$$

$$\hat{\Psi}_k = \Delta \Psi_k + sample(\sigma_{\Psi_k}). \tag{5.35}$$

5.2.2 Observation Model

An observation model characterizes a process of observations generation in a real world. It shall cover individual characteristics of a modeled sensor, including probabilistic noise and uncertainty. An observation model defines a conditional probability

$$p(\vec{y}_k | \vec{x}_k, m_k), \tag{5.36}$$

where $\vec{y}_k$ is an observation (sensor measurement), $\vec{x}_k$ is a state and m_k is a map at time step k.

Individual beams of multi-beam sensors are correlated. Hence, an independence is assumed between particular laser beams and the Equation 5.36 accumulated from individual beam measurement likelihoods can be formulated as

$$p(\vec{y}_k | \vec{x}_k, m_k) = \prod_{l=1}^{L} p(y_k^l | \vec{x}_k, m_k), \tag{5.37}$$

where L denotes number of beam measurements and y_k^l measurement l at time step k. However, the independence assumption holds only in an ideal case. In real applications, beam occlusions, surface omnidirectional reflectance or sensor rotation-based errors occur. If enough measurements are provided, the independence assumption may be supported by taking into account each n-th beam measurement. In practice, about 50 samples is selected from total 1400 samples to increase their independence and likewise to speed up the multiplication process of Equation 5.37.

In the perception task, solely laser sensors are utilized, for whom a beam-based observation model is employed [62]. It supposes that a beam-based sensor follows a Gaussian probability distribution with the mean at the actual distance of the sensor from its target. Moreover, additional introduced probabilistic errors are accounted for in the model.

The observation model consists of four linearly combined components. Foremost, characteristics of an unbiased laser beam measurement y_k are represented by a probability

$$p_{hit}(y_k|\vec{x}_k, m_k) = \begin{cases} \eta \mathcal{N}(y_k, y_k^*, \sigma_{hit}) = \eta \frac{1}{\sigma_{hit}\sqrt{2\pi}}\, e^{\frac{(y_k-y_k^*)^2}{2\sigma_{hit}^2}} & \text{if } y_{min} < y_k \leq y_{max} \\ 0 & \text{otherwise,} \end{cases} \tag{5.38}$$

where $\mathcal{N}(y_k, y_k^*, \sigma_{hit})$ represents univariate normal distribution of a random variable y_k with mean y_k^* and standard deviation σ_{hit}, y_{min} defines minimal and y_{max} maximal sensing range of a particular sensor, and η is a normalization factor

$$\eta = \frac{1}{\int_{y_{min}}^{y_{max}} \mathcal{N}(y_k, y_k^*, \sigma_{hit}) dy_k}. \tag{5.39}$$

Variable y_k^* represents an expected measurement value obtained e.g., by ray casting from state $\vec{x}_k$ in a map m_k.

The other three components represent observation errors occurring due to sensor noise, failures or detection of dynamic obstacles. To model beams reflection before reaching the target by small undetectable or unknown dynamic obstacles, the following probability can be derived from the exponential distribution as

$$p_{short}(y_k|\vec{x}_k, m_k) = \begin{cases} \lambda_{short}\, \frac{e^{-\lambda_{short} y_k}}{1-e^{-\lambda_{short} y_k^*}} & \text{if } y_{min} < y_k \leq y_k^* \\ 0 & \text{otherwise,} \end{cases} \tag{5.40}$$

where λ_{short} is an intrinsic parameter of the distribution. Furthermore, to model sensor failures and invalid measurements, a point-mass discrete distribution centered at y_{max} is incorporated

$$p_{max}(y_k|\vec{x}_k, m_k) = \begin{cases} 1 & \text{if } y_k = y_{max} \\ 0 & \text{otherwise.} \end{cases} \tag{5.41}$$

To comprehend occasional random measurements coming either from failures or sensor crosstalks, a uniform distribution, spread over the full observation range, is included in form of a probability

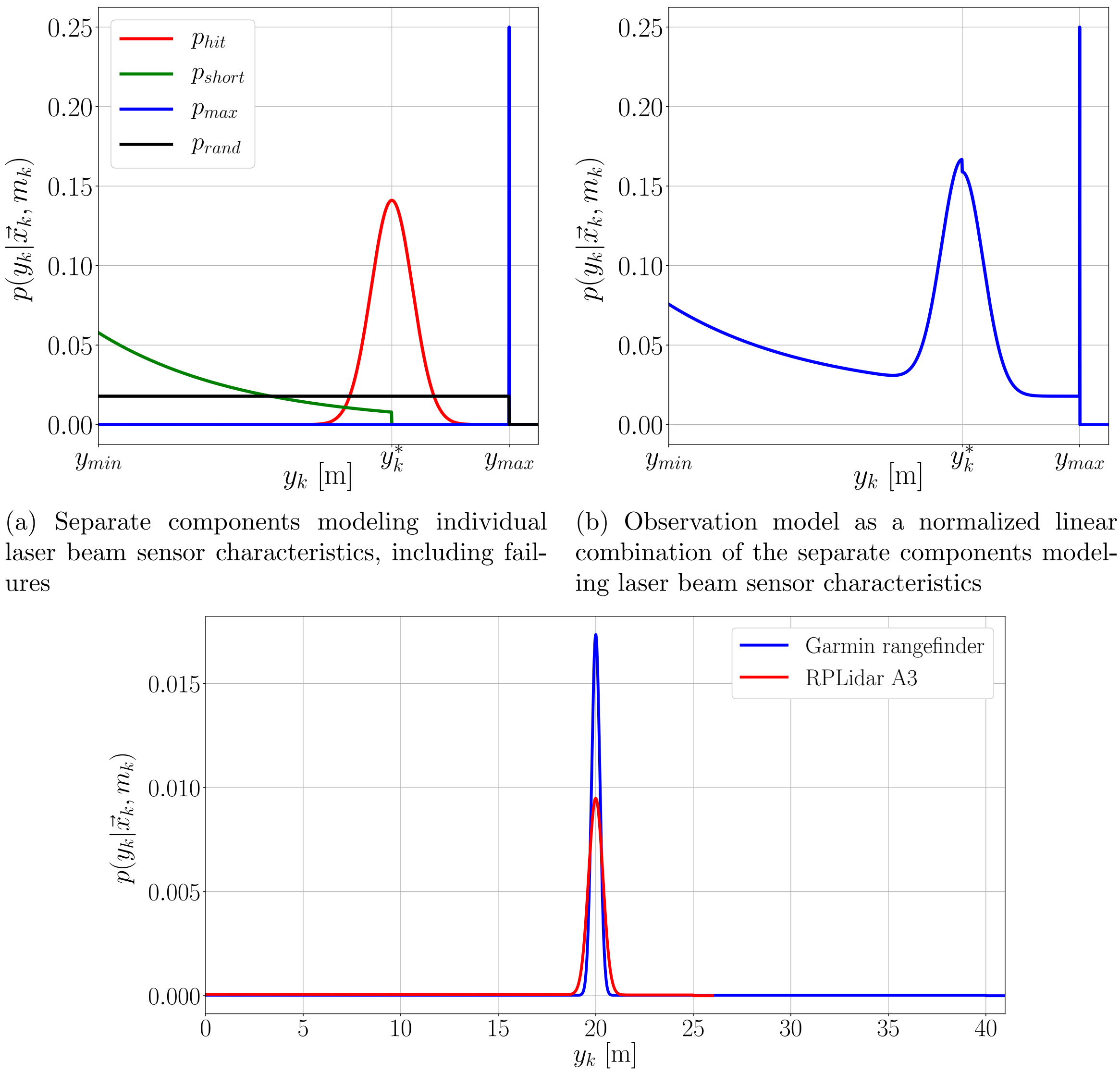
$$p_{rand}(y_k|\vec{x}_k, m_k) = \begin{cases} \frac{1}{y_{max}-y_{min}} & \text{if } y_{min} \leq y_k < y_{max} \\ 0 & \text{otherwise.} \end{cases} \tag{5.42}$$

The observation model, which was introduced above, is a result of a linear combination of the four components:

$$p(y_k|\vec{x}_k, m_k) = \begin{pmatrix} \nu_{hit} & \nu_{short} & \nu_{max} & \nu_{rand} \end{pmatrix} \begin{pmatrix} p_{hit}(y_k|\vec{x}_k, m_k) \\ p_{short}(y_k|\vec{x}_k, m_k) \\ p_{max}(y_k|\vec{x}_k, m_k) \\ p_{rand}(y_k|\vec{x}_k, m_k) \end{pmatrix}, \tag{5.43}$$

where constants ν represent components' importance weights for which holds

$$\nu_{hit} + \nu_{short} + \nu_{max} + \nu_{rand} = 1. \tag{5.44}$$



(a) Separate components modeling individual laser beam sensor characteristics, including failures

(b) Observation model as a normalized linear combination of the separate components modeling laser beam sensor characteristics

(c) Realistic observation models for each of the employed laser sensors with intrinsic parameters given by Table 5.1 and $y_k^* = 20\,\mathrm{m}$

Figure 5.3: Explicit visualization of an observation model, its specific components and a realistic observation model for the utilized laser sensors

Observation model of a single beam in Equation 5.43 represents single observation $p(y_k^l|\vec{x}_k, m_k)$ in Equation 5.37. A visual example for $\nu_{hit} = \nu_{short} = \nu_{max} = \nu_{rand} = 0.25$ is provided in Figure 5.3a and Figure 5.3b. In a multi-sensor situation, Equation 5.37 is extended to

$$p(\vec{y}_k|\vec{x}_k, m_k) = \prod_{s=1}^{S} \omega_s \left(\prod_{l=1}^{L_s} p(y_k^l|\vec{x}_k, m_k) \right), \tag{5.45}$$

where $s \in S$ denotes a sensor from set of sensors S, L_s is number of beam measurements provided by sensor s, $\omega_s \in \langle 0, 1 \rangle$ is importance weight (belief) of sensor s, and y_k^l is measurement l. Additionally to the assumption of independence between individual beams of a single sensor, mutual independence between individual sensors is likewise presumed.

In the preceding paragraphs, a set of parameters being a subject to tuning was introduced. These parameters $(\sigma_{hit}, \lambda_{short}, \nu_{hit}, \nu_{short}, \nu_{max}, \nu_{rand})$, also called intrinsic model parameters, need to be determined for each particular sensor. A deterministic method to learn the intrinsic parameters from data is described in [62]. A set of experimental data was measured for each sensor within a range interval 1 m to 40 m, according to Figure 2.6. To determine the intrinsic model parameters from these data, Algorithm 2 based on [62] is adopted. While rangefinders naturally return a single beam measurement, the rotational planar scanner returns a set of measurements. Hence, only a particular beam measurement corresponding to the measured target was selected. A summary of parameters identification for each sensor is presented in Table 5.1.

The experimental data were measured in a static environment. Hence the parameters given by Table 5.1 are subject to change in dynamic environments. For example, a rangefinder measuring distance to the ground (altimeter) might detect dynamic obstacles such as people or furniture changes. Although outer dynamic obstacles are not expected in higher altitudes, other agents (UAVs) might be detected during a mission with an airborne formation of helicopters. Both of these example situations are a subject of λ_{short} parameter tweaking in order to account for potential dynamic obstacles detection.

Sensor	σ_{hit}	λ_{short}	w_{hit}	w_{short}	w_{max}	w_{rand}
Garmin Lidar Lite	0.045	0.218	0.920	0.002	0.000	0.076
RPLidar A3	0.134	0.046	0.867	0.042	0.000	0.090

Table 5.1: Intrinsic model parameters of utilized laser sensors learned from experimental data by Algorithm 2

5.2.3 Sampling

Computational and time complexity of MCL grows with a number of state dimensions. The state $\vec{x}$ of a UAV with an assumption of slow movements has four dimensions - three for the 3D position and one for the heading of the UAV. Besides, during an initialization phase, MCL has to randomly sample the whole state space, which is frequently broad. To be able to comprehend the task, a set of precautions and assumptions is proposed in order to reduce the complexities by scaling down the sampling space.

Without loss of generality, a feasible hypothesis $\mathbf{h}$ is defined as a weighted UAV state $\mathbf{h} = \langle w_h, \vec{x}_h \rangle$ located inside an indoor environment, while not being in collision with any part of the environment assuming bounded 3D dimensions. Each hypothesis represents a UAV state, hence its virtual dimensions correlate with dimensions of the UAV. For sake of computational simplicity, the physical body of the UAV is assumed to be a ball with a collision radius denoted r. Based on the proposed platform design described in Chapter 2, the collision radius is defined as $r = 0.4\,\text{m}$.

In order to evaluate the feasibility of a hypothesis, its map location and a collision status need to be determined. A hypothesis $\mathbf{h}$ with a position $\vec{p}_h$ is evaluated as collision-free,

Algorithm 2 Learn Intrinsic Parameters

```
Input: $\mathbf{X}$, $\mathbf{Y}$, $m$        ▷ Set of states, set of observations and a map
Output: $\sigma_{hit}, \lambda_{short}, \nu_{hit}, \nu_{short}, \nu_{max}, \nu_{rand}$        ▷ Intrinsic Parameters
while convergence criterion not satisfied do
    for $\vec{x}_k$ in $\mathbf{X}$ do
        $\mathbf{Y}_k = \mathbf{Y}(\vec{x}_k)$        ▷ Set of measurements at state $\vec{x}_k$
        for $y_k^l$ in $Y_k$ do
            $\eta = \left[p_{hit}(y_k^l|\vec{x}_k, m) + p_{short}(y_k^l|\vec{x}_k, m) + p_{max}(y_k^l|\vec{x}_k, m) + p_{rand}(y_k^l|\vec{x}_k, m)\right]^{-1}$
            $e_{hit}^{k,l} = \eta\, p_{hit}(y_k^l|\vec{x}_k, m)$
            $e_{short}^{k,l} = \eta\, p_{short}(y_k^l|\vec{x}_k, m)$
            $e_{max}^{k,l} = \eta\, p_{max}(y_k^l|\vec{x}_k, m)$
            $e_{rand}^{k,l} = \eta\, p_{rand}(y_k^l|\vec{x}_k, m)$
            $y_k^{l*} = y(\vec{x}_k, m)$        ▷ Correct measurement at state $\vec{x}_k$
        end for
    end for
    $|Y| = \sum_k \sum_l (p_{hit}(y_k^l|\vec{x}_k, m) + p_{short}(y_k^l|\vec{x}_k, m) + p_{max}(y_k^l|\vec{x}_k, m) + p_{rand}(y_k^l|\vec{x}_k, m))$
    $\sigma_{hit} = \sqrt{\frac{1}{\sum_k \sum_l e_{hit}^{k,l}} \sum_k \sum_l e_{hit}^{k,l} (y_k^l - y_k^{l*})}$
    $\lambda_{short} = \frac{\sum_k \sum_l e_{short}^{k,l}}{\sum_k \sum_l e_{short}^{k,l} y_k^l}$
    $\nu_{hit} = |Y|^{-1} \sum_k \sum_l e_{hit}^{k,l}$
    $\nu_{short} = |Y|^{-1} \sum_k \sum_l e_{short}^{k,l}$
    $\nu_{max} = |Y|^{-1} \sum_k \sum_l e_{max}^{k,l}$
    $\nu_{rand} = |Y|^{-1} \sum_k \sum_l e_{rand}^{k,l}$
end while
return $(\sigma_{hit}, \lambda_{short}, \nu_{hit}, \nu_{short}, \nu_{max}, \nu_{rand})$
```

if inequality

$$||\vec{p}_h - \vec{p}_{h,nn}||_2 > r \tag{5.46}$$

is satisfied. Position vector $\vec{p}_{h,nn}$ represents location of the nearest occupied map element and $||\vec{a}||_2$ represents $\mathcal{L}_2$ (Euclidean) norm of vector $\vec{a}$. Finding $\vec{p}_{h,nn}$ is an equivalent of finding the nearest neighbor, for which a KD-tree representation of the map is utilized. Because the map is not updated online during a mission, initialization of the KD-tree representation is performed once at mission initialization during a map preprocessing phase. A nearest neighbor search complexity of a KD-tree ranges from $O(log(n))$ for the best case scenario to $O(n)$ for the worst [64]. Evaluation of an indoor location of a hypothesis is performed by ray casting in the z-axis of the map from the position of the hypothesis. This technique was already introduced in Section 4.3 for patching holes in ground data. When generating solely feasible hypotheses, these conditions of feasibility significantly reduce the sampling space, as shown in Figure 5.4a.

The sampling space in z-coordinate of the map is further reduced by utilizing an estimated altitude of the UAV. Distance measurements from the down-oriented rangefinder and a previous estimate of the UAV position are fused together in order to obtain the altitude estimate. During a takeoff phase, which presumably occurs at a consistent ground location with no considerable ground deformations, only a range measurement $y_{s_{down}}$ of the downward looking rangefinder s_{down} is utilized. An altitude estimate exclusively from s_{down} is defined as

$$z_{lim}^{est} = y_{s_{down}} + \Delta z_{s_{down},fcu}, \tag{5.47}$$

where $\Delta z_{s_{down},fcu}$ represents z-coordinate difference between fcu and s_{down} coordinate systems in the coordinate system of a map.

The limits of z-coordinate sampling are then given as

$$z_{lim}^{min} = z_{lim}^{est} - z_{lim}^{\delta}, \tag{5.48}$$

$$z_{lim}^{max} = z_{lim}^{est} + z_{lim}^{\delta}, \tag{5.49}$$

where z_{lim}^{δ} represents an offset of z-coordinate sampling. The sampling offset z_{lim}^{δ} yields a sampling range to account for measurements error and recent vertical motion of a robot. During consequent mission phases, s_{down} measures distance to heterogeneous objects on the ground. To account for these measurements, the estimation limits at time step k are altered to

$$z_{lim}^{min} = z_{k-1} - z_{lim}^{\delta} - |z_{lim}^{est} - z_{k-1}|, \tag{5.50}$$

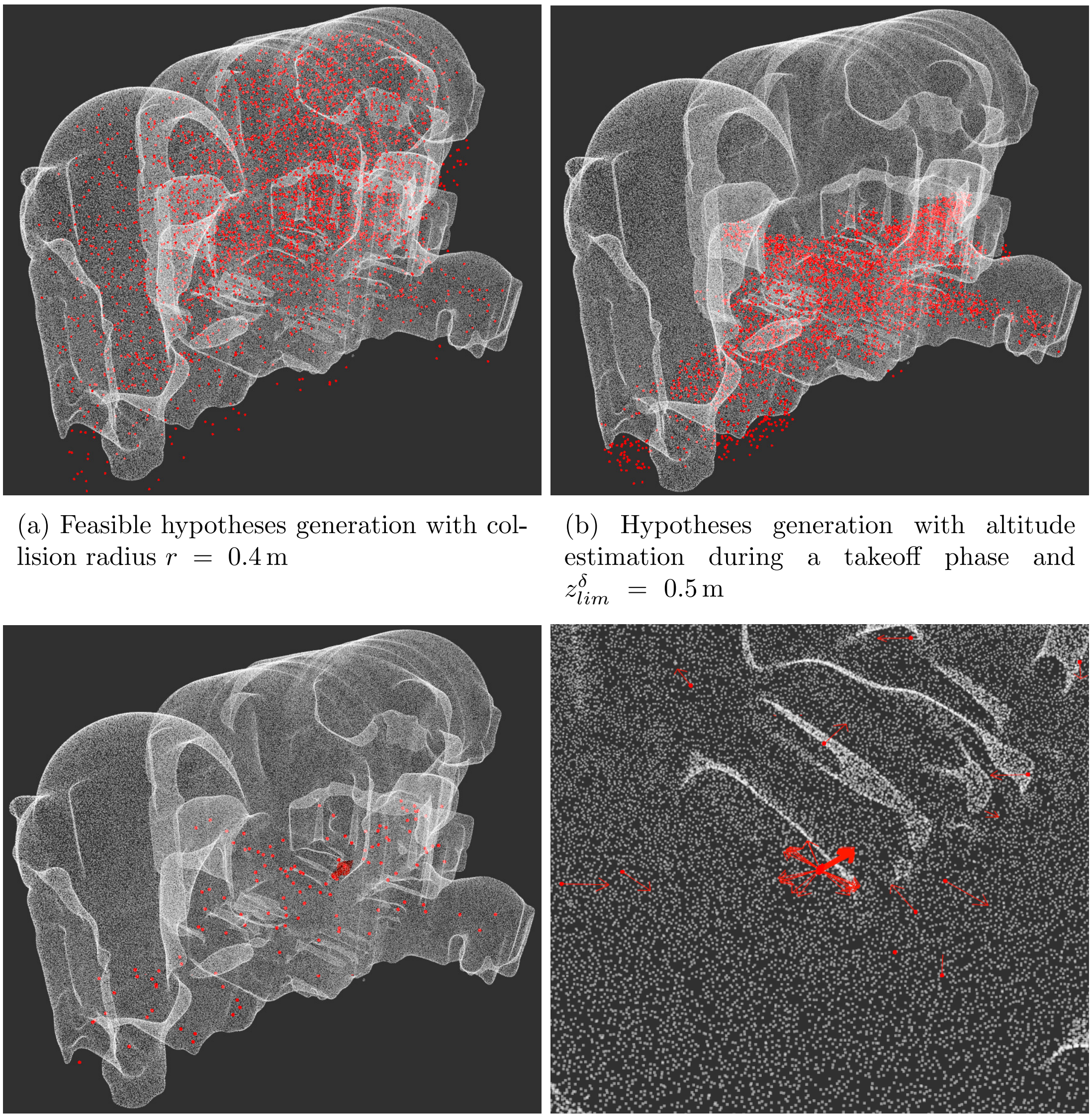
$$z_{lim}^{max} = z_{k-1} + z_{lim}^{\delta} + |z_{lim}^{est} - z_{k-1}|. \tag{5.51}$$

Sampling space limitation extended by z-coordinate filtering is shown in Figure 5.4b. Besides, a description of sampling control during various mission phases is described by a finite state machine later in this section.

Adaptive Sampling

To this point, reduction of hypotheses sampling space was discussed. Let us now introduce sampling techniques providing speed up and convergence guarantee of the MCL algorithm. Guarantee of convergence to the global minimum in finite time is not a characteristic of basic MCL algorithm, which may lead to incorrect convergence to a local minimum. Therefore, in each resampling step, a subset of hypotheses with the lowest weights is thrown away and replaced with a set of new randomly generated hypotheses over the whole sampling space. This feature introduces capabilities of MCL to find a global minimum even if it already converged into a local one, as shown in Figure 5.4c.

Many historical monuments share a characteristic of symmetricity, which likewise induces possibility to converge to a local minimum. To emphasize this characteristic, a static robot in the center of a square map can be considered. In this situation, MCL converges into one of the four local minimums - the center of the square with four different headings. Basic MCL algorithm is not able to cope with this situation without a distinct robot motion over the map. To speed up convergence during these situations, a subset of hypotheses is replaced by a set of new hypotheses matching the position of the latest state estimate of the algorithm with heading Ψ_{out}, however with a heading randomly selected from set $\{\Psi_{out} - \frac{\pi}{2}, \Psi_{out} + \pi, \Psi_{out} + \frac{\pi}{2}\}$. This mechanism to cope out with symmetricity of operational spaces is denoted on lines 10-15 in Algorithm 3 and visualized in Figure 5.4d.



(a) Feasible hypotheses generation with collision radius $r = 0.4\,\mathrm{m}$

(b) Hypotheses generation with altitude estimation during a takeoff phase and $z^{\delta}_{lim} = 0.5\,\mathrm{m}$

(c) Additional global feasible hypotheses generation after MCL convergence

(d) Local heading sampling of feasible hypotheses to cope with environmental symmetry

Figure 5.4: Sampling techniques of the MCL implementation to support correct and brisk convergence rate (hypotheses are marked red)

An efficient number of hypotheses M remains to be determined since it massively influences the performance of the algorithm. Sampling the space with few hypotheses might heavily extend the convergence process. On the other hand, sampling with large M might extremely extend the computational time, because the observation model is computed for each hypothesis separately. Besides the algorithm for sampling of the space with static M, two methods for adaptive sampling [62, 65] are adopted, which calculate online an effective number of hypotheses.

The first method, adopted from Augmented-MCL algorithm [62], controls the ratio of hypotheses added globally in order to prevent convergence to an erroneous local minimum. It compares the short-term with the long-term likelihood of observations used to specify a number of hypotheses to be globally added as

$$M_{global} = M\ max\left(0.0, 1.0 - \frac{w_{fast}}{w_{slow}}\right), \tag{5.52}$$

where

$$w_{slow} = w_{slow} + \alpha_{slow}(w_{avg} - w_{slow}), \tag{5.53}$$

$$w_{fast} = w_{fast} + \alpha_{fast}(w_{avg} - w_{fast}), \tag{5.54}$$

$$w_{avg} = \frac{1}{M}\sum_{m=1}^{M} w_m, \tag{5.55}$$

where w_m states for importance weight of a hypothesis with index m. Algorithm constants α_{slow} and α_{fast} define decay rate for the exponential filters that estimate the long-term and short-term averages. It is required to hold

$$0 \leq \alpha_{slow} \ll \alpha_{fast}. \tag{5.56}$$

The second method of adaptive sampling is KLD-sampling [65], which estimates a sufficient number of hypotheses M_{KLD}. Its key idea lies in bounding of the error introduced by the sample-based representation of the MCL. It provides such M_{KLD}, that the Kullback-Leibler divergence between the maximum likelihood estimate based on the hypotheses and the true posterior does not exceed a pre-specified threshold ϵ. The M_{KLD} estimate is based on drawing from a discrete distribution with k different bins and for

$$M_{KLD} = \frac{1}{2\epsilon}\chi^2_{k-1,1-\delta} \tag{5.57}$$

guarantees with probability $1-\delta$ that the K-L divergence between the maximum likelihood estimate (MLE) and the true distribution is less than ϵ. The previous equation can be further approximated by

$$M_{KLD} \doteq \frac{k-1}{2\epsilon}\left(1 + \frac{2}{9(k-1)} + \sqrt{\frac{2}{9(k-1)}} z_{1-\delta}\right)^3, \tag{5.58}$$

where $z_{1-\delta}$ is the upper $1-\delta$ quantile of the standard normal $\mathcal{N}(0,1)$ distribution.

The proposed implementation of the bins is based on octrees extended with fourth dimension - robot's heading. Similarly to octrees, the whole heading space $\langle 0, 2\pi)$ is divided to intervals of a specific size. These intervals, together with the 3D position, constitute the bins. Prior to each resampling step, a number of occupied bins k by current hypotheses is counted and M_{KLD} is estimated by Equation 5.58.

During computation of the motion model (line 5 in Algorithm 1), the set of hypotheses is moved according to Equation 5.31. At this stage, a subset of hypotheses might become non-feasible by colliding with the environment. This subset of hypotheses is replaced by a set of new random global hypotheses.

Algorithm 3 Resampling Step

```
Input: $M_{KLD}$, $M_{global}$, $M_{local}$                ▷ Sampling limits
       $\mathbf{H}$                                        ▷ Set of hypotheses
Output: $\mathbf{H}_{resampled}$                           ▷ Updated set of hypotheses
$\mathbf{H}_{resampled} = \emptyset$                       ▷ Empty set initialization
for i in $\langle 0, M_{KLD})$ do
    $\mathbf{h}$ = draw from $\mathbf{H} \propto w_{\mathbf{H}}$    ▷ Sample with respect to importance weights
    $\mathbf{H}_{resampled} = \mathbf{H}_{resampled} + \mathbf{h}$
end for
for i in $\langle 0, M_{global})$ do
    $\mathbf{h}$ = generate_random_hypothesis()            ▷ Sample over the whole sample space
    $\mathbf{H}_{resampled} = \mathbf{H}_{resampled} + \mathbf{h}$
end for
for i in $\langle 0, M_{local})$ do
    $\Psi$ = draw random from $\{\frac{\pi}{2}, -\frac{\pi}{2}, \pi\}$
    $\mathbf{h}$ = MCL estimate in previous step
    $\mathbf{h}$.state.heading = $\mathbf{h}$.state.heading + $\Psi$    ▷ Sample rotated hypotheses
    $\mathbf{H}_{resampled} = \mathbf{H}_{resampled} + \mathbf{h}$
end for
return $\mathbf{H}_{resampled}$
```

5.2.4 Enhancements

From MCL point of view, a mission is divided into two dependent phases - takeoff and the mission itself. During takeoff, the main objective is to determine an initial global estimate, while during the mission, the objective is to track a UAV movement. Therefore, sampling behavior modifications are proposed to alter the behavior of the MCL according to three states, given by a finite state machine outlined in Figure 5.5.

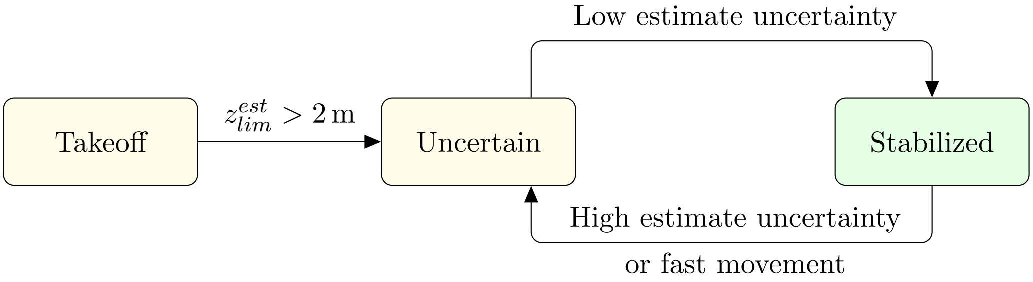


Figure 5.5: Monte Carlo Localization sampling state machine

At the beginning of the mission, a takeoff is initialized from an unknown ground location. At low altitudes, dynamic obstacles, mainly in the form of people, are likely to be present. As a consequence, the importance weights of the sensors in Equation 5.45 are during the takeoff phase set according to the estimated altitude of the UAV as

$$\omega_s = \begin{cases} 1.0 & \text{if } s \text{ is a vertical rangefinder} \\ \frac{1}{2} z_{lim}^{est} & \text{otherwise.} \end{cases} \tag{5.59}$$

Hence, up to limit of 2 m, the importance weight of the horizontal sensor is linearly scaled by data from the altimeter to cope with the presence of dynamical obstacles. In altitudes above 2 m, the sensors importance weights are then set to $\omega_s = 1$. A determinant of the covariance matrix of the state estimate is computed in order to evaluate the state estimate uncertainty. The determinant value then triggers transitions between the Stabilized and Uncertain states. Parameters of the sampling processes in each particular state are summarized in Table 5.2, where M_{min} and M_{max} define lower and upper saturation limits of the total number of hypotheses given as

$$M = M_{KLD} + M_{local} + M_{global}. \tag{5.60}$$

State	M_{min}	M_{max}	M_{global}	M_{local} (static ratio)
Takeoff	high	high	high (static ratio)	high
Uncertain	medium	medium	medium (static ratio)	medium
Stabilized	low	low	Augmented-MCL (dynamic ratio)	low

Table 5.2: Parameters of hypotheses sampling for states defined in Figure 5.5

Having a set of hypotheses of size M, state estimate output $\vec{x}_k$ at time step k is given as

$$\vec{x}_k = \frac{\sum_{m=1}^{M} \omega_m \vec{x}_m}{\sum_{m=1}^{M} \omega_m}, \tag{5.61}$$

where ω_m is importance weight and $\vec{x}_m$ is state of hypothesis m at time k. Hence, resulting state estimate is given by weighted average over set of hypotheses.

5.3 Scan Matching

The global state estimation presented in the previous section is not single-handedly suitable for real-time localization of a fast-moving UAV due to low update rate and sampling dimensions complexity. For these reasons, having an initial global state estimate yielded by the global estimation, fast local refinement procedure is initialized based on onboard data alignment. Iterative Closest Point algorithm [31] is utilized to perform the 3D alignment of onboard data and a map. Map possession provides a reliable alignment reference improving robustness and performance of the system. In other words, a map registration is performed in each localization time step by ICP.

By concept, the ICP algorithm tries to minimize an error between two point clouds **P** and **Q** according to some error metric. Let us define point cloud as a collection of multidimensional points (in our case 3D with XYZ dimensions). Given a source point cloud **P** with points $\vec{p} \in \mathbf{P}$ and a reference point cloud $\mathbf{Q}$ with points $\vec{q} \in \mathbf{Q}$, an initial problem is finding correspondence pairs $(\vec{p_i}, \vec{q_i})$, where $\vec{p_i} \in \mathbf{P}, \vec{q_i} \in \mathbf{Q}$. Given these correspondences, a transformation $\mathbf{T}_{icp}$ from $\mathbf{P}$ to $\mathbf{Q}$, such that when applied to $\mathbf{P}$, the transformation assigns all correspondence points $\vec{p_i}$ onto $\vec{q_i}$, is estimated. Due to the non-convexity of the optimization, an initial transformation guess is supplied to the ICP algorithm. In the proposed implementation, the transformation is estimated from an onboard IMU to supply speed and accuracy of the map registration process. According to [66], registration process can be modularized into four stages: selection, matching, rejection, and alignment. In the following sections, a description of utilized techniques in each of the stages is included.

As implementation of the registration process, a modular framework for aligning in 3D – Point Cloud Library [67, 68] – is adopted. Decoupling the registration process to two separate registration processes for lateral (XY and heading estimation) and vertical (altitude estimation) movement is introduced later in Section 5.4. Both, lateral and vertical, registration processes share their underlying principle, which is described herein Section 5.3.

5.3.1 Selection

During the selection phase, a source and reference point clouds are prepared. Primarily, only a subset of the input point clouds is registered in order to reduce point redundancy and significantly speed up the convergence, while still yielding sufficient results. As aforementioned in Section 4.3, uniform sampling is applied to a map in point cloud representation. Besides, uniform sampling is likewise applied to horizontal data taken onboard, although double the granularity of points with respect to the map is kept in order to maintain a higher level of features' details.

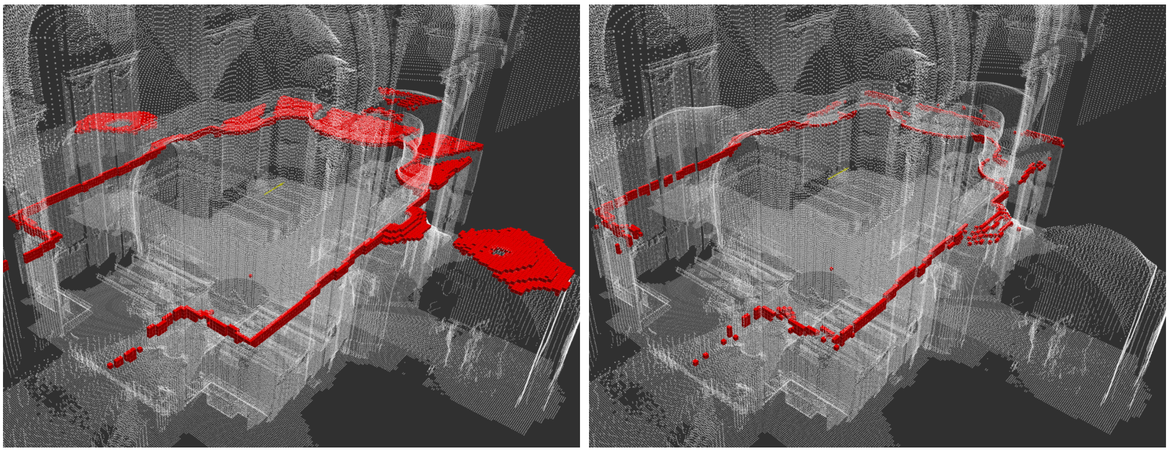
At time step k, a reference point cloud is obtained by selecting a subset of points $\vec{q_i} \in \mathbf{Q}$ from a map, which all satisfy

$$l(\vec{x}_{k-1}, \vec{\Omega}_{k-1}, d_1) < \vec{q_i} < l(\vec{x}_{k-1}, \vec{\Omega}_{k-1}, d_2), \tag{5.62}$$

where $l(\vec{x}_{k-1}, \vec{\Omega}_{k-1}, d_{1/2})$ define planes parallel to the XY plane of the robot with state $\vec{x}_{k-1}$ and orientation $\vec{\Omega}_{k-1}$. Distance $d_1 = -d_2$ specifies translation of these planes in z-axis of the robot. Orientation $\Omega_{\vec{k-1}}$ is provided by the onboard autopilot since the roll and pitch of the UAV are not specifically estimated. In other words, a subset of points is selected from point cloud $\mathbf{Q}$, where each point is located inside a 3D interval defined by two planes derived from the UAV pose.

Equation 5.62 preserves visually occluded points, making them incorrectly observable by the horizontal laser scanner from certain positions, as shown in Figure 5.6a. A point is evaluated observable if a path of a laser beam from a sensor position (rigidly defined by a robot position) to the particular point is collision-free. To determine a collision status of such path, a ray-casting algorithm, implemented over octree representation of a map, is employed. Therefore, all unobservable points are filtered out from the reference point cloud. Final selection of a reference point cloud from the map is shown in Figure 5.6b.

The introduced proposition of the map registration decreases robustness during the registration of a single measurement from the horizontal laser scanner onto a 3D stripe of map points. During a UAV movement, map features present in the reference cloud may not

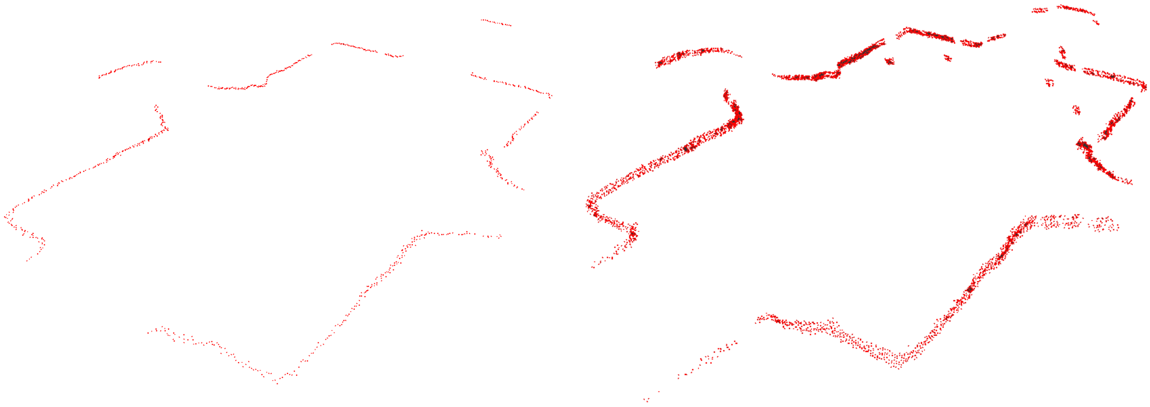


(a) Without occlusions

(b) With occlusions (final)

Figure 5.6: Visualization of a reference point cloud obtained during the selection phase of ICP

be present in the source cloud yet, and an initial transformation guess becomes even more decisive. Therefore, usage of a short history of measurements is proposed to increase the robustness of the map registration during a robot motion. Selection of a source point cloud from onboard data is visualized in Figure 5.7.



(a) Single measurement

(b) Aggregation of multiple past measurements during an upward movement

Figure 5.7: Visualization of a source point cloud obtained during the selection phase of ICP

5.3.2 Matching & Rejection

Objective of the matching phase is to determine correspondence pairs of points in source and reference data. A greedy approximation of finding ideal correspondences by pairing each $\vec{p}_i \in \mathbf{P}$ with its closest neighbor $\vec{q}_i \in \mathbf{Q}$ is adopted. The Point Cloud Library implementation utilizes FLANN library [69, 70] for fast nearest neighbor searches, which significantly speeds up the search in comparison with a naive brute-force pairing. Further evaluation of pairs feasibility is done during the rejection phase.

In the rejection phase, a filter stack is built, where each layer contains a correspondence pair evaluation filter called a correspondence rejector. The objective of a correspondence rejector is to remove a subset of pairs to increase robustness and accuracy of the alignment. Typical subjects to removal are outliers or pairs diverging from a statistical distribution across all the pairs. Proper pairs rejection significantly increases robustness and convergence rate to a global minimum of the alignment phase, since it filters out noise measurements, or source and reference divergences. In the following paragraphs, a brief description of employed correspondence rejectors is introduced. These rejectors are applied on correspondence pairs provided by the matching step in this particular order.

Median distance rejection procedure filters out pairs with a distance larger than the median of all the point-to-point pair distances. Compared to a fixed threshold value of distance filtering, the median rejection yields better results since it adapts to a distribution of the distances between the two sets.

Duplicate reference matches rejection procedure filters out correspondences, where a point $\vec{q}$ from a reference cloud is assigned to multiple points $\vec{p}_i$ from a source cloud. If multiple points $\vec{p}_i$ from the source cloud are assigned to a single point $\vec{q}$ from the reference cloud, only the pair with minimum Euclidean distance is kept and the rest is rejected. The duplicate reference matches rejector is the reason for keeping double the granularity of a source cloud to preserve the best correspondences, which could be possibly thrown away by the uniform sampling.

RANSAC-based rejection procedure filters out outlier pair correspondences using Random Sample Consensus algorithm. This method applies random transformations to subsets of given sets and rejects correspondences based on the Euclidean distance of pairs after the random transformation is applied to the source cloud. Due to the randomized nature of RANSAC algorithm, this approach significantly increases chances of the alignment phase to converge into a global optimum.

5.3.3 Alignment

Objective of the alignment phase is to find a transformation $\mathbf{T}_{icp}$, which minimizes an error function $J(\mathbf{T}_{icp})$ over N correspondence pairs. Hence, the objective is to minimize

$$J(\mathbf{T}_{icp}) = \sum_{i=1}^{N} ||\mathbf{T}_{icp}\vec{p}_i - \vec{q}_i||^2. \tag{5.63}$$

In literature, multiple methods for solving the previous equation, classified as an unweighted point-to-point error metric [71], can be found [71, 72, 73, 74]. Comparison of these methods can be found in [75]. From the various approaches, a closed-form solution using singular value decomposition [72] is utilized in the implementation of PCL library.

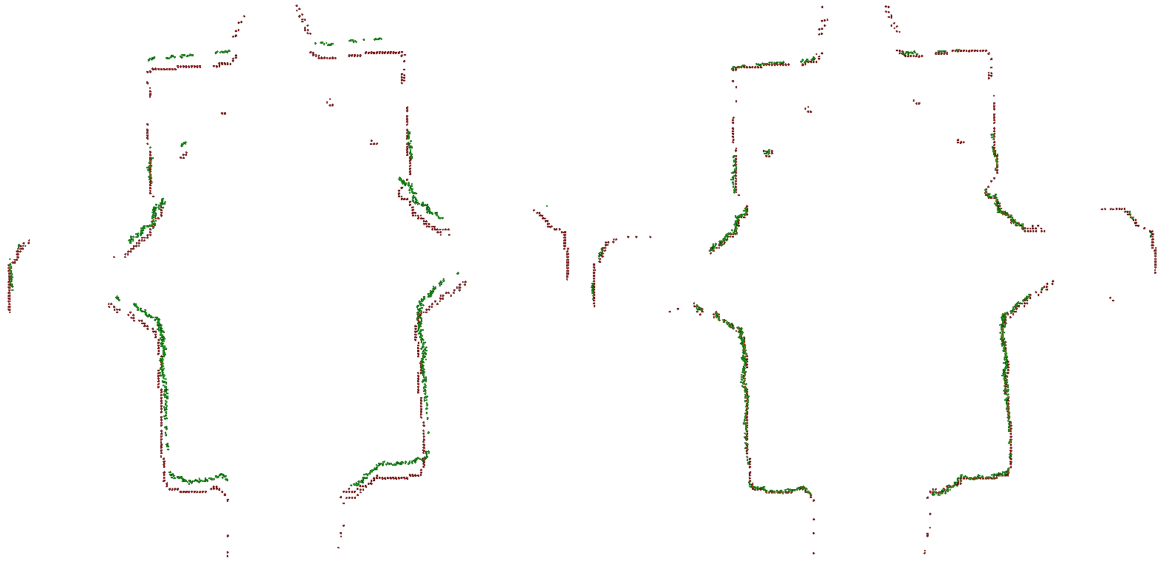
Convergence rate of the alignment phase depends on an initial guess of transformation $\mathbf{T}_{icp}$. To determine the initial transformation, the dead reckoning of IMU data is employed. Similarly to the utilization of the dead reckoning in Section 5.2.1, the internal autopilot velocities and accelerations are integrated to estimate the pose of the robot.

Since ICP is an iterative algorithm, termination criteria need to be determined. A maximum number of iterations and absolute mean square error value are employed to check the performance of the ICP. Both the parameters are empirically defined based on given map granularity, quality of an initial transformation guess, and desired registration accuracy.

At time step k, the alignment phase yields estimate of Equation 5.6 in matrix form as

$$\mathbf{T}^k_{fcu} = \left[\begin{array}{c|c} \mathbf{R}^k_{fcu} & \vec{t}^{\,k}_{fcu} \\ \hline 0 & 1 \end{array}\right] = \mathbf{T}^k_{icp}\mathbf{T}^{k-1}_{fcu}. \tag{5.64}$$

The final state estimate at time step k is then concatenated from the 3D position given by the translation vector $\vec{t}^{\,k}_{fcu}$ and heading determined from the orientation matrix $\mathbf{R}^k_{fcu}$. A visual example of the final alignment is shown on real data in Figure 5.8.



(a) Before alignment

(b) After alignment

Figure 5.8: Example of the ICP lateral alignment on real data. Red color depicts map (reference cloud) and green color depicts planar laser scanner data (source cloud).

5.4 Fusion

In the previous two sections, two approaches for map-based localization were presented. To get as accurate state estimate as possible, both estimates obtained by these methods are fused together in order to obtain a final state estimate output, as presented in Figure 5.9. The fusion process is thoroughly described in this section.

The proposed approach intents to take advantage of both localization algorithms. Monte Carlo Localization yields a global state estimate, however is relatively slow even with usage of the adaptive sampling techniques. Besides, its accuracy is heavily dependent on a map resolution, and motion and observation model parameters. Both of the models are specific for each particular object they describe. A procedure for obtaining an approximation of observation model parameters was described in Section 5.2.2. According to an available expected motion model behavior, its parameters were chosen empirically. Despite the concept, the motion model might not produce flawless results for dynamically complicated vehicles, like UAVs.

On the other hand, Iterative Closest Point yields a local transformation estimate, when an initial global estimate is provided in order to cope with an extensive state space. After an initial global estimate $\vec{x}_0^{mcl}$ is found, the ICP algorithm is initialized and starts tracking the system state history over time by concatenating local transformations provided by ICP. This refinement process yields significantly faster and precise state tracking results than MCL itself. ICP state estimation is decoupled to two parallel estimations - lateral (XY axes and heading) and vertical (altitude). There are two fundamental reasons for parallelization.

Primarily, a vast difference between data volume in horizontal and vertical planes is taken into consideration. The horizontal laser scanner provides 14 000 samples per second, which is a subject for reduction. On the other hand, in the vertical plane, only two observations are obtained – downward y_{down} and upward y_{up}. The vast difference in the data volumes requires decoupling, otherwise the horizontal estimation would heavily overweight the vertical one. As described in Section 5.3, a short history of concatenated horizontal scans is registered onto a reference cloud being a subset of map points during the ICP procedure. This registration process is performed in 3D and therefore likewise finds the transformation in the vertical plane. However, if the reference cloud does not contain salient features in the vertical plane, a global z-coordinate optimum might not be distinguishable. In such case, the output estimate is an aftereffect of the initial transformation guess.

5.4.1 Vertical Estimation

During the vertical estimation, invalid real or map-based measurements can be obtained. Real sensor yields invalid measurements with random probability exhibited by sudden maximal range or not-a-number value. A map-based measurement is invalid when the ray casting procedure does not hit an occupied cell. Besides, the downward looking rangefinder may detect dynamic obstacles, which are represented by an identifiable discrepancy between real and map-based observations. Let s_{down} be downward and s_{up} upward oriented rangefinders with their particular real $(y_{down},\ y_{up})$ and map-based $\left(y_{down}^{map},\ y_{up}^{map}\right)$ observations. If either y_{down} or y_{down}^{map} observation is invalid, or a dynamic obstacle is detected according to

$$|y_{down} - y_{down}^{map}| > \delta_{y_{down}}, \tag{5.65}$$

a z-coordinate estimate is obtained at time step k as

$$z_k = z_{k-1} + y_{up}^{map} - y_{up}. \tag{5.66}$$

The dynamic obstacle threshold $\delta_{y_{down}}$ specifies difference threshold for real and map-based measurements, whose exceeding classifies the difference as a proof of dynamic obstacle. On the other hand, if either y_{up} or y_{up}^{map} observation is invalid, the estimate is given by

$$z_k = z_{k-1} + y_{down} - y_{down}^{map}. \tag{5.67}$$

In case both sensors are producing invalid data, the estimation is given by the lateral ICP initialized according to the dead reckoning principle, described in Section 5.3.3. And finally, having valid observations from both sensors results to vertical ICP estimation with a source cloud formulated from y_{down} and y_{up}, and a reference cloud formulated from y_{down}^{map} and y_{up}^{map}.

As illustrated in Figure 5.9, both estimators share evaluation information between themselves. MCL shares its global state estimate $\vec{x}_0^{mcl}$ if its accuracy, given by the error covariance $\mathbf{\Sigma}_k^{mcl}$, is found sufficient. On the other hand, the ICP estimator evaluates the accuracy of the registration process by computing absolute mean square error ϵ_k^{icp} between reference point cloud and source point cloud transformed by the local transformation found during the registration process itself. The absolute mean square error metric is used to evaluate, whether the estimate is stuck in a local minimum. In this case, both estimators are reinitialized in order to find the correct state.

5.4.2 Kalman Filtering

Both state estimates are fused using a Kalman Filter. Without loss of generality, UAV is assumed to be a linear dynamic system for a short period of time during hovering and slow flights with small tilts. These flights characteristics are expected in confined environments of historical monuments. A Linear Kalman Filter (LKF) is utilized to estimate an internal state of the UAV from a series of noisy measurements. Linear Kalman Filter is optimal and recursive algorithm for estimating state of a stochastic system. It is recursive since it updates the current state using the previous state and current observations, rather than the entire history. The optimality is yielded by minimization of the mean-square error of the system state.

Both estimates, together with IMU data, represent uncertain information about the dynamical system. Using LKF, a hypothesis is obtained about the state of the continuously changing system. LKF implementation can be divided into two distinct phases – prediction and correction.

During the prediction phase at time step k, a state prediction$\widehat{\vec{x}}_k$ and an error covariance prediction $\widehat{\mathbf{\Sigma}}_k$ are determined as

$$\widehat{\vec{x}}_k = \mathbf{A}_k \vec{x}_{k-1} + \mathbf{B}_k \vec{u}_k, \tag{5.68}$$

$$\widehat{\mathbf{\Sigma}}_k = \mathbf{A}_k \mathbf{\Sigma}_{k-1} \mathbf{A}_k^T + \mathbf{Q}_k, \tag{5.69}$$

where $\mathbf{A}_k$ is the state-transition model, $\mathbf{B}_k$ is the control-input model, $\vec{u}_k$ is the system input, and $\mathbf{Q}_k$ is the covariance of the process noise. During the correction phase at time step k, corrections are performed according to sensor observations $\vec{y}_k$ to get the final state estimate $\vec{x}_k$ and error covariance $\mathbf{\Sigma}_k$ as

$$\mathbf{K}_k = \widehat{\mathbf{\Sigma}}_k \mathbf{P}_k^T (\mathbf{P}_k \widehat{\mathbf{\Sigma}}_k \mathbf{P}_k^T + \mathbf{S}_k)^{-1}, \tag{5.70}$$

$$\vec{x}_k = \widehat{\vec{x}}_k + \mathbf{K}_k (\vec{y}_k - \mathbf{P}_k \widehat{\vec{x}}_k), \tag{5.71}$$

$$\mathbf{\Sigma}_k = (\mathbf{I} - \mathbf{K}_k \mathbf{P}_k) \widehat{\mathbf{\Sigma}}_k, \tag{5.72}$$

where $\mathbf{K}_k$ is an optimal Kalman gain that minimizes the residual error, $\mathbf{P}_k$ is the observation model, $\mathbf{S}_k$ is the covariance of the observation noise, and $\mathbf{I}$ is an identity matrix.

The dynamical system is defined at time step k as

$$\vec{x}_k = \vec{x}_{k-1} + \Delta k \, \vec{u}_k, \tag{5.73}$$

where state $\vec{x}_k$ is given by Equation 5.13, Δk is the time difference between time steps k and $k-1$, and

$$\vec{u}_k = \begin{pmatrix} v_k^x & v_k^y & v_k^z & \omega_k^z \end{pmatrix}^T \tag{5.74}$$

is the input of the system in form of state velocities provided by the autopilot. Associating the dynamical model of a UAV from Equation 5.73 with Equation 5.69 yields

$$\mathbf{A}_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{B}_k = \begin{bmatrix} \Delta k & 0 & 0 & 0 \\ 0 & \Delta k & 0 & 0 \\ 0 & 0 & \Delta k & 0 \\ 0 & 0 & 0 & \Delta k \end{bmatrix}. \tag{5.75}$$

The process noise $\mathbf{Q}_k \in \mathbb{R}^{4\times4}$, the observation noise $\mathbf{S}_k \in \mathbb{R}^{8\times8}$ and the observation model $\mathbf{P}_k \in \mathbb{R}^{8\times4}$ matrices are given as

$$\mathbf{Q}_k = \operatorname{diag}(Q_x, Q_y, Q_z, Q_\Psi), \tag{5.76}$$

$$\mathbf{S}_k = \operatorname{diag}(S_x^{mcl}, S_y^{mcl}, S_z^{mcl}, S_\Psi^{mcl}, S_x^{icp}, S_y^{icp}, S_z^{icp}, S_\Psi^{icp}), \tag{5.77}$$

$$\mathbf{P}_k = \begin{bmatrix} \mathbf{I}_{4\times4} \\ \mathbf{I}_{4\times4} \end{bmatrix}. \tag{5.78}$$

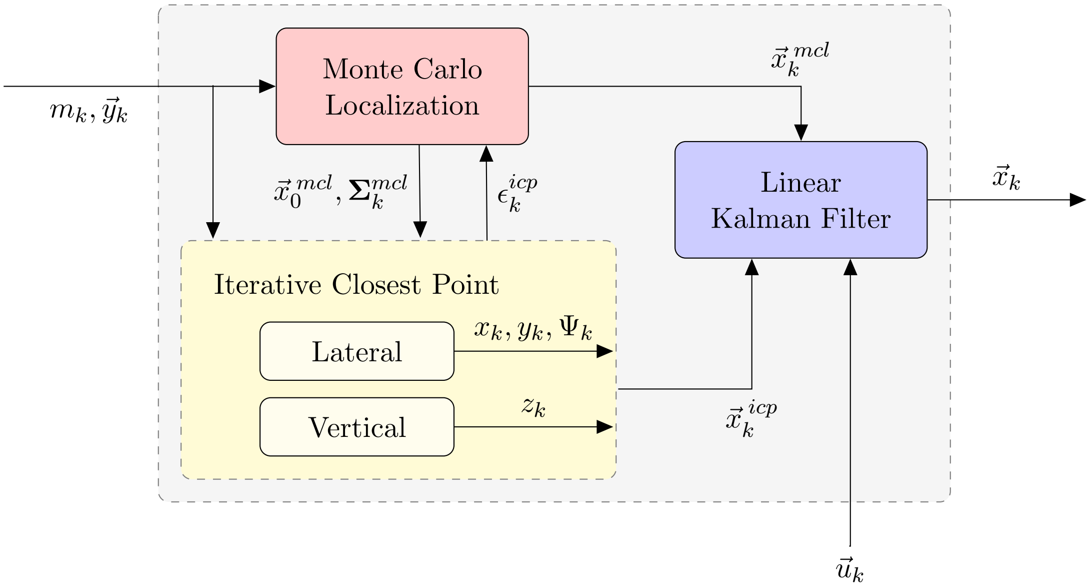


Figure 5.9: Workflow diagram of the state estimation process. Inputs of the fusion at a time step k are map m_k, sensor observations $\vec{y}_k$ and an input of the UAV dynamical model $\vec{u}_k$. A local state refinement and tracking by ICP algorithm is initialized after global state estimate $\vec{x}_0^{mcl}$ is provided by MCL. Reinitialization procedures are controlled with respect to MCL estimate covariance $\mathbf{\Sigma}_k^{mcl}$ and ICP absolute mean square error ϵ_k^{icp}.

Chapter 6: Experimental Verification

Contents

This chapter presents validation and verification of the developed system in simulation and on real data obtained during testing flights. The performance and reliability of the localization system is verified in a realistic simulation prior to its deployment to position control feedback loop of a UAV. The main intention of simulation is to estimate suitability of the developed system for deployment in safety-critical environments of historical buildings, to reduce probability of failures and to obtain a qualitative analysis of the system. Performance verification on real data evaluates suitability of the developed system in conditions matching reality. Having a precise outer reference system, a quantitative analysis is performed in order to evaluate the limits of the proposed localization system.

The following sections describe in detail evaluation metrics used for assessment of the localization system, simulation setup with analysis of simulation results, obtaining of real-world ground truth dataset during testing flights, and the quantitative analysis of the system on data acquired during the same testing flights with a ground truth reference. The parameters of the localization system evaluation in the simulation and on real data are summarized in Table 6.1. Multimedia materials complementing analysis of the system are available at `http://mrs.felk.cvut.cz/theses/petracek2019`.

6.1 Evaluation Metrics

To objectively evaluate experiments, an analysis tool for localization estimates is introduced. A quantitative analysis is performed with respect to a ground truth pose data obtained either instantly in a simulation or in real-world by an outer reference system, as described in Section 6.3. Common trajectory evaluation methods for visual and/or inertial odometry are employed, such as Root Mean Square Error (RMSE) and Absolute Trajectory Error (ATE) metrics. Likewise common Relative Pose Error (RPE) metric is omitted, since it measures local accuracy over a fixed time interval, hence measuring an odometry drift. Since a map provides a reference, the localization system suppresses any long-term odometry drift, as will be clear from evaluation figures in the following sections. RPE can be used to evaluate a

	Specification	Symbol	Value
Map	Resolution	-	10 cm
MCL	Motion model	$\epsilon_0, \epsilon_1, \epsilon_2, \epsilon_3, \epsilon_4, \epsilon_5, \epsilon_6$	0.01, 0.5, 0.6, 0.8, 0.7, 0.7, 0.4
	Observation model	$\sigma_{hit}, \lambda_{short}, \nu_{hit}, \nu_{short}, \nu_{max}, \nu_{rand}$	Summarized in Table 5.1
	Augmented-MCL	$\alpha_{slow},\ \alpha_{fast}$	0.1, 0.85
	KLD-sampling	$z_{1-\delta}, \epsilon$	2.327, 0.035
	Hypotheses count	M	dynamic
	Sampling limits	z^{δ}_{lim}	0.150 m
	- Takeoff	$M_{min}, M_{max}, M_{global}, M_{local}$	500, 5000, 0.4M, 0.05M
	- Uncertain		300, 3000, 0.2M, 0.05M
	- Certain		150, 1500, dynamic, 0.05M
ICP	Termination criteria		
	- Max iterations	-	5
	- MSE threshold	ϵ^{icp}	0.2 m
	Scan aggregation time	-	200 ms
	Selection plane offset	$d_1,\ d_2$	0.15 m, −0.15 m
LKF	Observation noise		
	- MCL	S_x, S_y, S_z, S_Ψ	1e^{-2} m, 1e^{-2} m, 1e^{-2} m, 1e^{-2} rad
	- ICP	S_x, S_y, S_z, S_Ψ	8e^{-4} m, 8e^{-4} m, 5e^{-3} m, 8e^{-4} rad
	Process noise	Q_x, Q_y, Q_z, Q_Ψ	5e^{-5} m, 5e^{-5} m, 5e^{-5} m, 8e^{-5} rad

Table 6.1: Parameters of the localization system for simulation and real experiments presented in Chapter 6 and Chapter 7

global error of a trajectory by averaging over all possible time intervals. Besides, ATE and RPE are correlated metrics and usage of both of them is redundant.

The simplest quality estimate of a localization history (trajectory) provides Root Mean Square Error defined as

$$RMSE(\vec{x}_{1:K}) = \sqrt{\frac{\sum_{k=1}^{K}(\vec{x}^*_k - \vec{x}_k)^2}{K}}, \tag{6.1}$$

where K represents total number of discrete time steps, $\vec{x}_k$ localization state estimate and $\vec{x}^*_k$ ground truth state at time step k. RMSE measures error between two samples at a given time. The Absolute Trajectory Error quantifies global consistency and like RMSE directly measures the error between two samples at a given time, however firstly aligns the true and estimated trajectories. The ATE alignment is achieved in a closed form by singular value decomposition, whose implementation is adopted from [76]. According to [77], the ATE of a trajectory at time step k can be computed as

$$ATE(\vec{x}_k) = (\vec{x}_k^*)^{-1} S \, \vec{x}_k, \tag{6.2}$$

where S is a rigid body transformation that maps estimated trajectory $\vec{x}_{1:K}$ onto the ground truth trajectory $\vec{x}^*_{1:K}$. To evaluate the metric, RMSE of ATE is then computed over all time indices as

$$RMSE(\vec{x}_{1:K}) = \sqrt{\frac{\sum_{k=1}^{K} ATE(\vec{x}_k)^2}{K}}. \tag{6.3}$$

6.2 Simulation

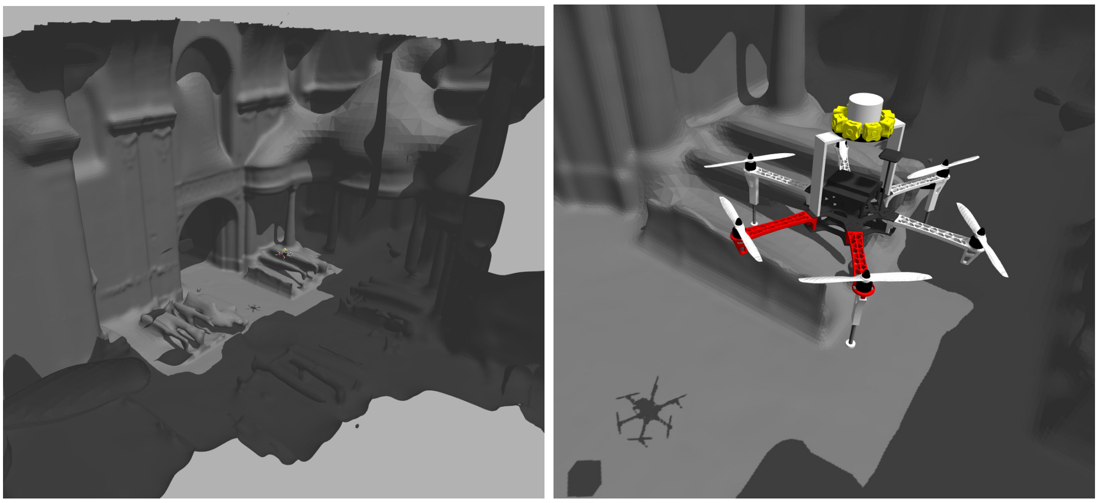
A realistic simulator Gazebo[1] [78] is employed due to its convenient integration with ROS[2] framework [79], which is used for the system implementation. In the simulator, identical UAV controllers and sensors are simulated, including control and sensor noise, as used for real robots. This utilization provides easy transfer from the simulation environment onto a real UAV without considerable effort. Furthermore, the simulation environment provides ground truth data used for evaluation of the system, which is hard to obtain in the real world, especially for challenging localization task in indoor environments. Gazebo simulator reflects the real world as close as possible, although it never truly expresses all factors of reality, such as influence of an environment (aerodynamic feedback), wind turbulence, propeller vibrations, etc.

The simulation validates feasibility of the localization system for deployment of a real UAV platform. It also assists with development of a reliable and robust system together with its integration into a position feedback of autonomous UAV control. Hence, to manifest the reality as close as possible, a point cloud captured in Church of St. Mary Magdalene, presented in Chapter 4, is converted into a mesh model and integrated into Gazebo simulator, as shown in Figure 6.1a. In the picture, roof and part of the outside wall of the model are visually hidden to provide a clear view on interiors of the simulation world. This model is used for the simulations throughout this section.

During development of the system, multiple simulation experiments with varying horizontal and vertical velocities of the UAV were performed. Several trajectories were generated and used for verification and validation of the localization system. Six experiments are presented to demonstrate the localization system performance. Linear and angular velocities of the UAV in most of the experiments are $0.5\,\mathrm{m\,s^{-1}}$, and $0.5\,\mathrm{rad\,s^{-1}}$ respectively, which is planned to be used as limits during real missions in historical buildings for safety reasons. The localization system is running online during the simulation experiments and no offline post-processing is performed in the same way as during a real deployment. A position control feedback from the state estimation is not established in this section, and the UAV is flying according to simulated GNSS. The experiments are summarized by the following list.

[1] Gazebo 9.0, `http://gazebosim.org/`

[2] ROS Melodic, `http://www.ros.org/`



(a) Mesh model of Church of St. Mary Magdalene in Chlumín

(b) Detail of a UAV platform used for simulations

Figure 6.1: A testing world and a UAV model in Gazebo simulator

- Figure 6.2: Localization system convergence during UAV takeoff. The figure demonstrates that the convergence time of MCL was approx. 9 s after takeoff. After that, the global estimate triggered local refinement by ICP, as visible at the z-axis figure.
- Figure 6.3: Oscillations of position reference in vertical plane by ±1 m demonstrating vertical movement tracking with absence of long-term lateral and heading drifts.
- Figure 6.4: Reference position changes in each direction with static heading demonstrating state tracking in 3D space. The presented trajectory visits several diagonal positions with respect to an initial pose of the UAV with 1 m distance in each axis.
- Figure 6.5: Heading rotation of 360° demonstrating performance during orientation changes.
- Figure 6.6: Circular trajectory in XY plane with center-oriented heading of the vehicle.
- Figure 6.7: Circular trajectory with $2\,\mathrm{m\,s^{-1}}$ horizontal velocity of the vehicle. The experiment demonstrates system performance for higher velocities, where roll and pitch angles, that are assumed to be close to zero during slow flights, can no longer be neglected. The influence of vehicle dynamics is particularly visible at z-axis figure.

Three of the presented experiments – diagonal references and both circular trajectories – are visualized in Figure 6.8. The figure visually compares the final LKF state estimate and ground truth position with respect to a map. Table 6.2 specifies MCL, ICP, and LKF state estimate accuracy during each of the experiments. The presented simulations show the performance of the system in scenarios with various complexity. In each experiment, the system is capable of estimating and following state changes with precision specified in Table 6.2. The experiments show, that the final state estimation update rate depends on data size, maximal number of iterations and desired accuracy of ICP algorithm. During the simulation, the system provides state estimate at 10 to 15 Hz. However, onboard a UAV, a faster update rate can be achieved, since no simulation computation is required.

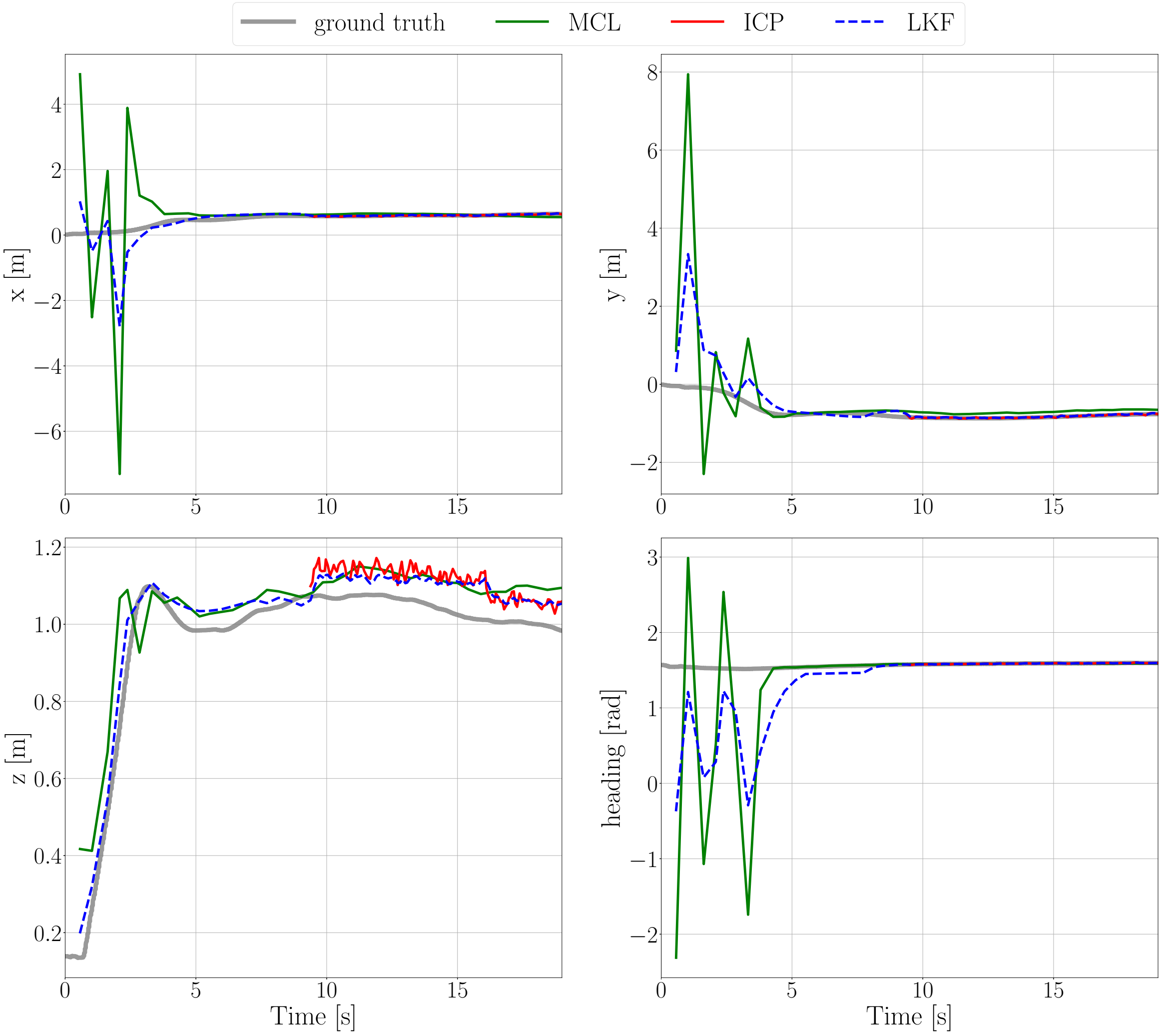
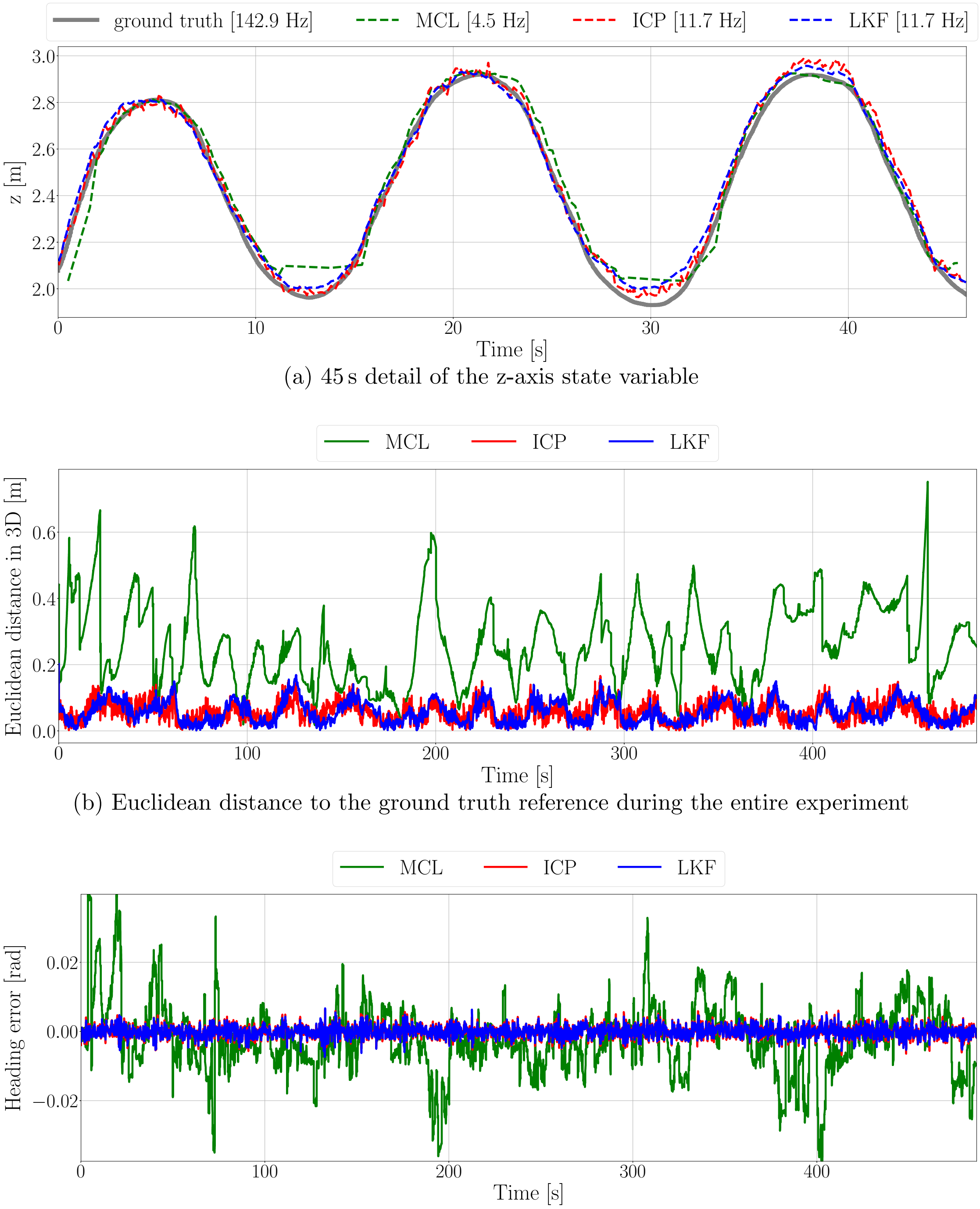


Figure 6.2: Separate state variables during takeoff phase of a simulated UAV flight as estimated by individual algorithms

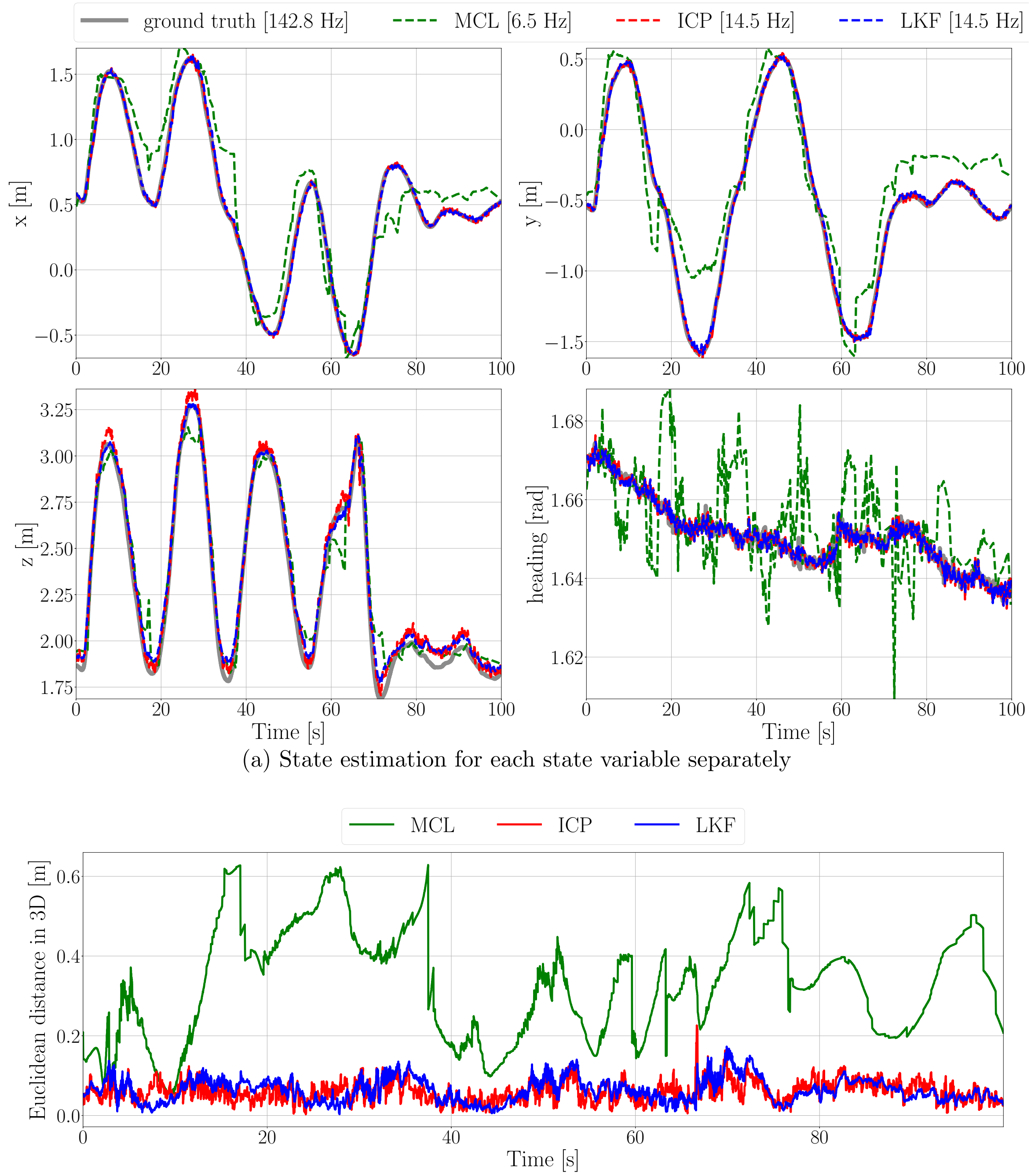


(a) 45 s detail of the z-axis state variable

(b) Euclidean distance to the ground truth reference during the entire experiment

(c) Heading error with respect to ground truth during the entire experiment

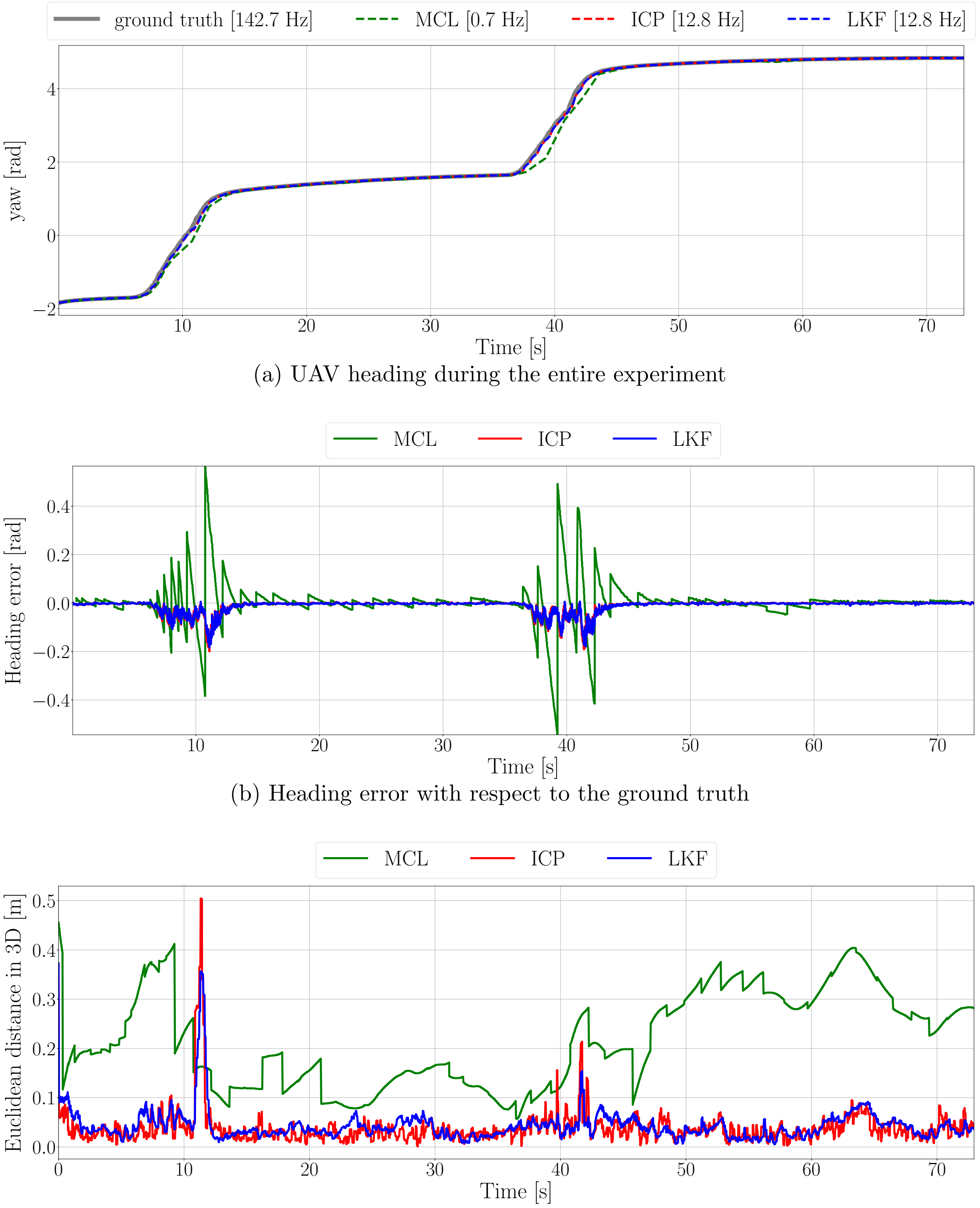
Figure 6.3: Simulation verification of the localization system during 8 min vertical oscillations to demonstrate altitude tracking and long-term drift suppression in position and heading of the UAV



(a) State estimation for each state variable separately

(b) Euclidean distance to the ground truth reference during the entire experiment

Figure 6.4: Simulation verification of the localization system for trajectory tracking with fixed heading and simultaneous movement in each translational axis to demonstrate state tracking in 3D space

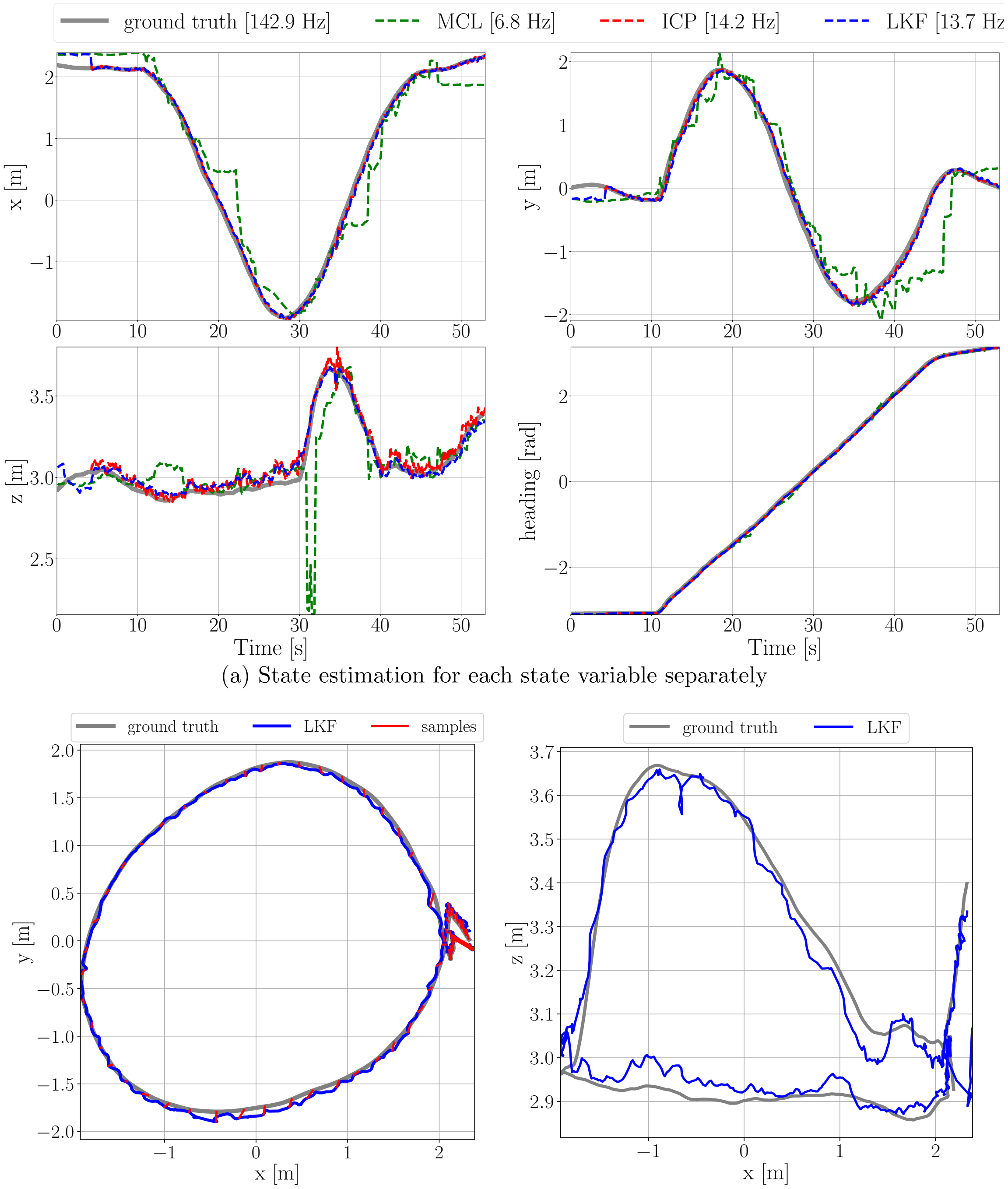


(a) UAV heading during the entire experiment

(b) Heading error with respect to the ground truth

(c) Euclidean distance to the ground truth reference during the entire experiment

Figure 6.5: Verification of the localization system in a simulation with a fixed position and 360° rotation of the UAV to demonstrate tracking of heading changes

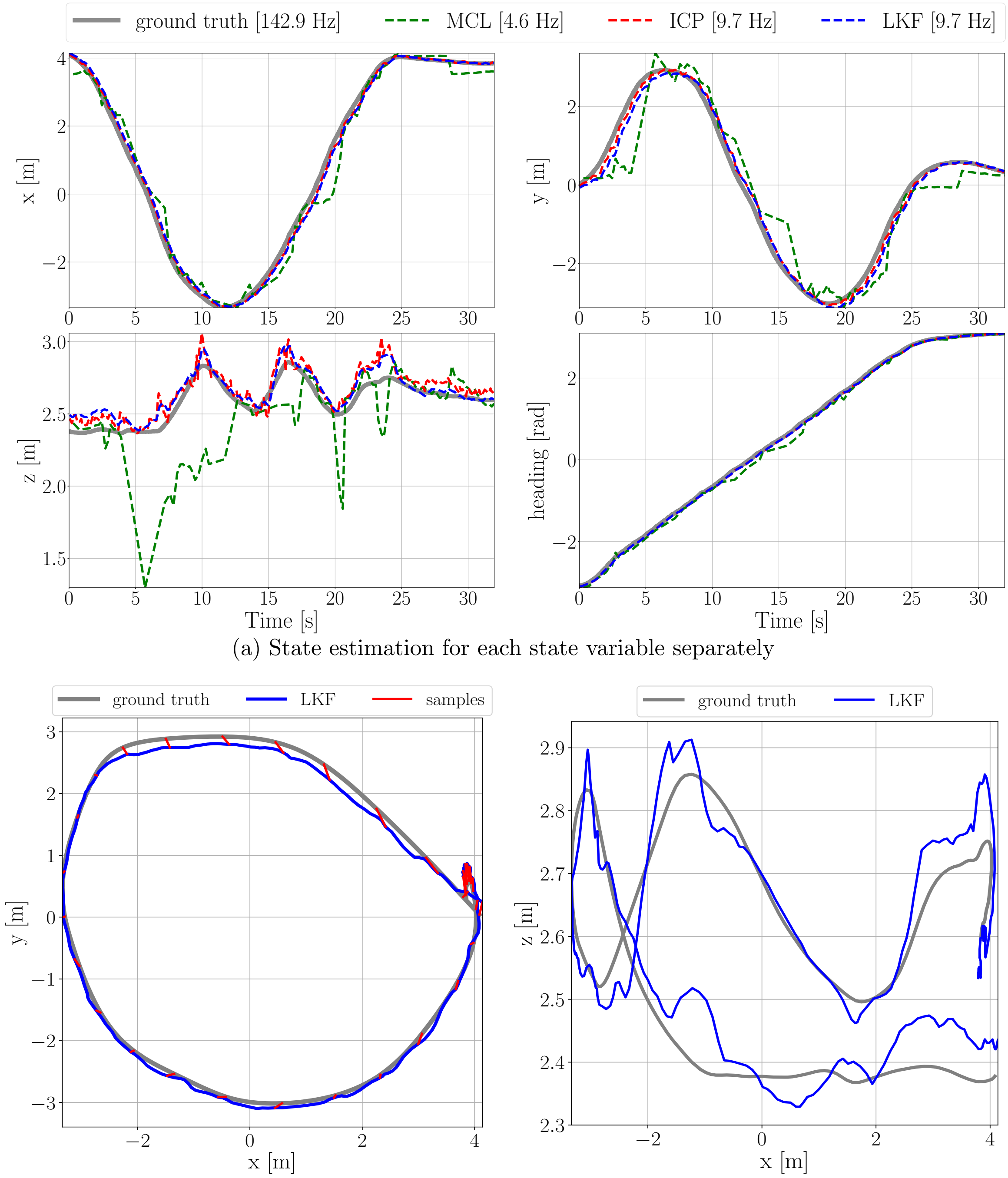


(a) State estimation for each state variable separately

(b) Trajectory top view with the final state estimate, ground truth and indices between time-closest samples. Every 10th closest sample pair is depicted by red color.

(c) Trajectory side view of the final estimate and ground truth

Figure 6.6: Verification of the localization system in a simulation using a circular trajectory with heading oriented into the center of the trajectory at 0.5 m s^{-1} horizontal velocity



(a) State estimation for each state variable separately

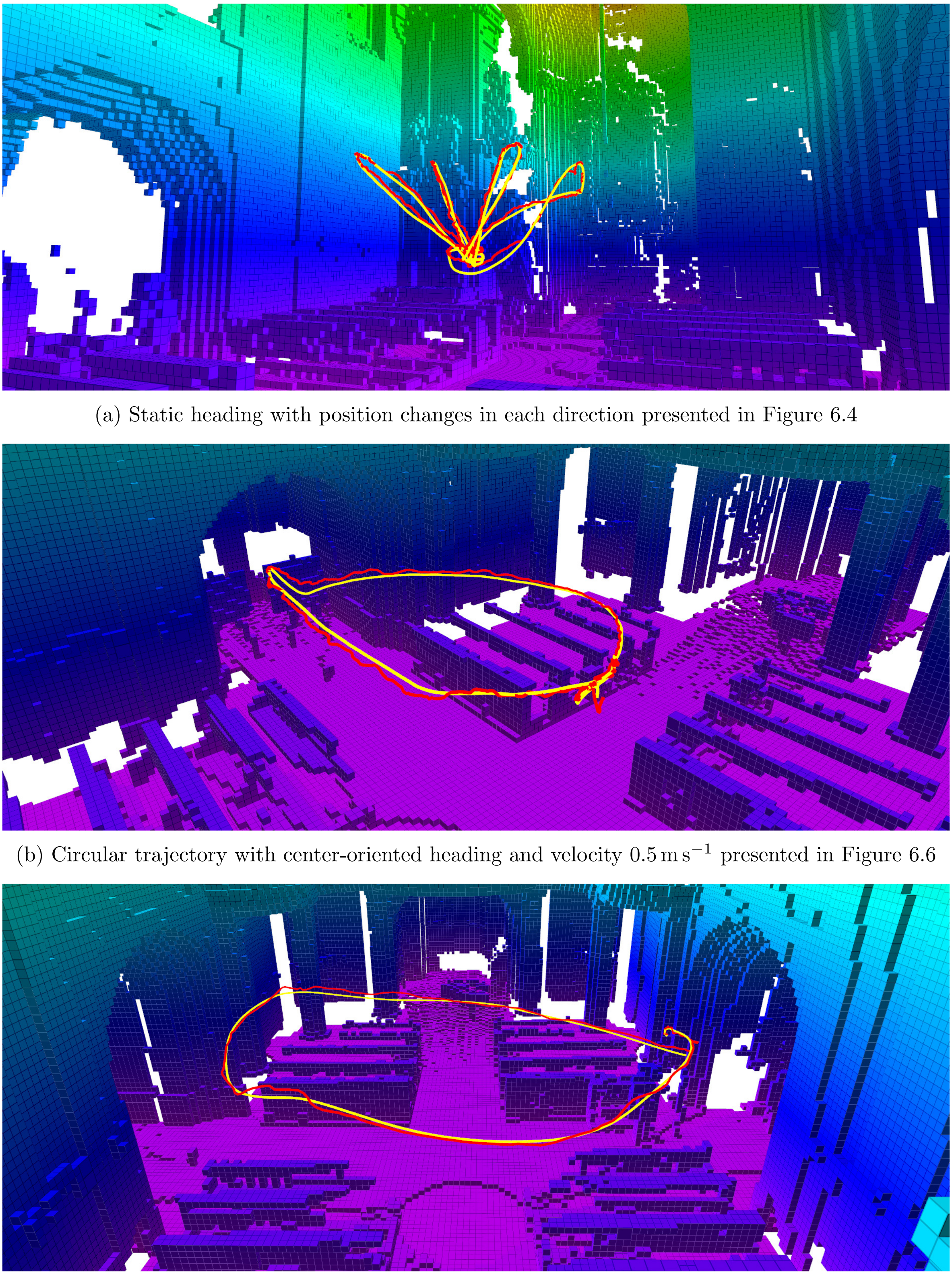
(b) Trajectory top view with the final state estimate, ground truth and indices between time-closest samples. Every 10th closest sample pair is depicted by red color.

(c) Trajectory side view of the final estimate and ground truth. Neglected roll and pitch angles influence the vertical estimation during faster movements.

Figure 6.7: Verification of the localization system in a simulation using a circular trajectory with heading oriented into the center of the trajectory at $2\,\mathrm{m\,s^{-1}}$ horizontal velocity

Trajectory	Vertical Oscillations	Diagonal Deviations	Yaw Rotation	Circular with Heading Change	
max linear velocity [m s^{-1}]	0.5	0.5	0.5	0.5	2.0
Figure	Figure 6.3	Figure 6.4	Figure 6.5	Figure 6.6	Figure 6.7
ATE translation [m]					
MCL	0.244	0.277	0.221	0.482	0.548
ICP	0.039	0.043	0.045	0.062	0.135
LKF	0.042	0.051	0.048	0.097	0.195
ATE heading [°]					
MCL	0.630	3.259	31.198	8.159	6.477
ICP	0.585	0.813	2.333	1.414	3.882
LKF	1.152	0.866	4.031	1.728	4.168
RMSE translation [m]					
MCL	0.284	0.346	0.233	0.566	0.730
ICP	0.061	0.063	0.050	0.094	0.241
LKF	0.167	0.067	0.050	0.127	0.320
RMSE heading [°]					
MCL	0.630	0.863	7.117	1.952	5.631
ICP	0.582	0.810	2.333	1.616	4.267
LKF	0.579	0.809	2.049	1.601	4.277
max translation error [m]					
MCL	0.649	0.623	0.395	1.512	2.075
ICP	0.154	0.226	0.504	0.232	0.446
LKF	0.167	0.156	0.356	0.336	0.686
max heading error [°]					
MCL	2.269	3.324	31.746	8.813	16.280
ICP	1.547	3.306	12.157	4.017	8.637
LKF	1.540	3.307	11.232	3.972	8.598

Table 6.2: Summary of the state estimate accuracy for MCL, ICP and LKF during simulations, where LKF yields the final system state estimate



(a) Static heading with position changes in each direction presented in Figure 6.4

(b) Circular trajectory with center-oriented heading and velocity $0.5\,\mathrm{m\,s^{-1}}$ presented in Figure 6.6

(c) Circular trajectory with center-oriented heading and velocity $2\,\mathrm{m\,s^{-1}}$ presented in Figure 6.7

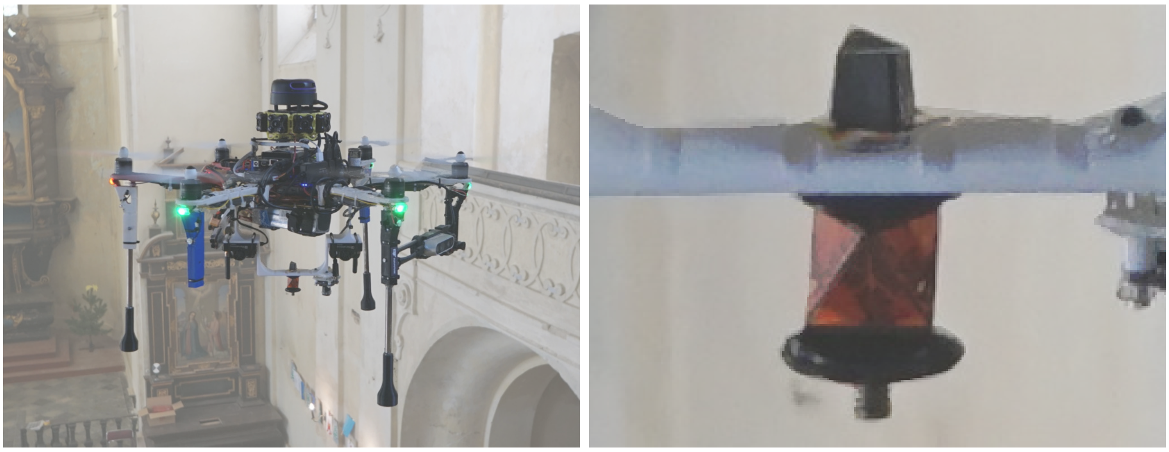
Figure 6.8: Simulation verification – 3D position trajectories visualization (yellow: ground truth, red: final state estimate)

6.3 Ground Truth Dataset

In Section 4.1, mapping of Church of St. Mary Magdalene in Chlumín with multistation Leica Nova MS60 was described. Here in Section 6.3, usage of total/multi stations Leica Nova MS60 and Leica Viva TS16 for tracking of a UAV movement during a UAV flight is introduced. Equipping the UAV with a reflector (glass prism with a special coating on the reflective surfaces [80]), a station is able to lock and track the reflector in 3D space. Particularly, the UAV is equipped with Leica GRZ101 360° Mini Prism reflector characterized by properties in Table 6.3. The onboard mounting is shown in Figure 6.9. Due to the lightweight and small dimensions of the reflector, the stations provide only the 3D position of the reflector relative to a coordinate system of the station. The tracking of a target is handled by Automatic Target Recognition (ATR) system, which locks and tracks the reflector target. During short occlusions between a station and the target, a predicted trajectory of the target is followed to be able to focus back once the occlusions disappear. A different UAV platform with identical sensory equipment was employed during the presented deployment, because the proposed hardware platform in Chapter 2 was not prepared for deployment at the time.

Height	Diameter	Weight	Point Accuracy	3D Translation	3D Rotation
30 mm	28 mm	<30 g	±1.5 mm	Yes	No

Table 6.3: Parameters of Leica GRZ101 360° Mini Prism reflector



(a) Airborne photo of the UAV

(b) Detail of the reflector

Figure 6.9: UAV mounted with onboard sensors and Leica GRZ101 Mini Prism during a manual flight, where a localization dataset with 3D translation was collected

In Figure 6.10, two snapshots of a manual flight with a UAV carrying the Leica GRZ101 360° Mini Prism tracked by both stations are shown. Requirements of the ATR system in the context of UAV tracking can be summarized by the following list:

- clear view from a station to the target,

- no extremely steep angles between UAV and the stations (constraint on maximal altitude based on a distance between UAV and the stations, which is limited in indoor environments),
- slow aerial movement to prevent loss of target track, and
- no large nonlinear movements during short occlusions.

During multiple manual flights in Church of St. Mary Magdalene, both stations were capable to some extent track the UAV and report its 3D position at a frequency of 5 Hz. Distance between the UAV and both stations was varying from 5 to 30 m with altitude of the UAV varying from 0 to 15 m. As the UAV was controlled manually by a human operator, some of the maneuvers lack smoothness in both, position and velocity. Also, the legs of the UAV repeatedly intercepted the visual trajectory between stations and the reflector resulting in short period occlusions. In these situations, target reference got frequently lost and no data were coming from any of the stations. Hence, the data further used as a ground truth reference contain short time period outages as the stations initialized re-locking procedure. Due to fewer tracking outages for multistation Leica Nova MS60, its data were adopted as the ground truth translation reference.

Coordinate system of the map exactly matched coordinate systems of the stations. However, time synchronization between the UAV and stations was unfortunately not provided and data obtained by the stations had to be manually synchronized with data acquired onboard the UAV. Since the ground truth position data rate is 5 Hz, the synchronization process introduces maximal time error of 200 ms, which for a UAV with velocity of 1 m s^{-1} results to an accuracy of 0.2 m. Additionally, to cope with lack of reference rotation, a rotation for each translation sample was determined by ICP algorithm described in Section 5.3. This step was performed offline with parameters of ICP preset to obtain as accurate rotation estimate as possible.

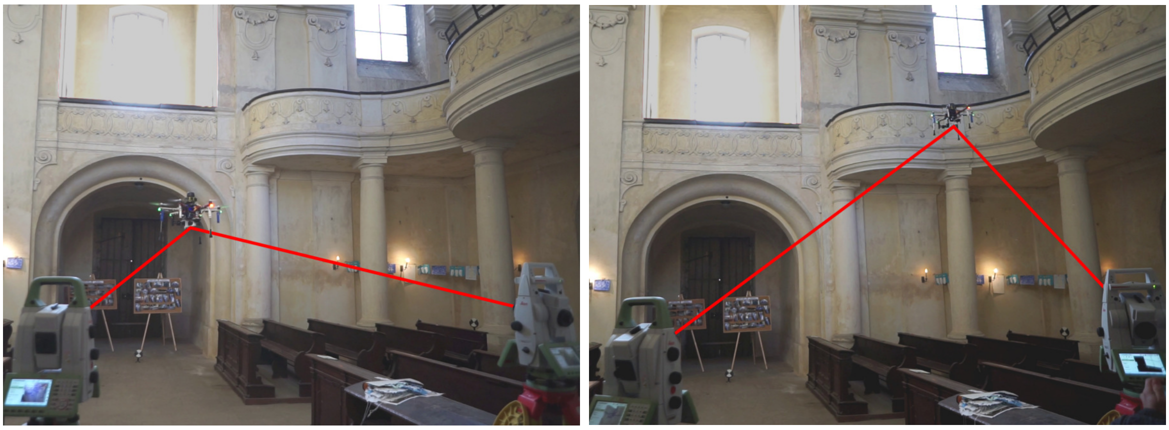


Figure 6.10: Demonstration of automatic tracking of a prism reflector mounted onboard a UAV by the Leica stations. The red lines are added to the picture to highlight the stations' laser beams measuring distance to the onboard reflector.

6.4 Verification on Real Data

In this section, a set of experiments is evaluated on real-world data described in Section 6.3. From multiple manual flights tracked by an outer reference system, three particular trajectories are presented. The helicopter, equipped with all the presented sensors as shown in Figure 6.9a, was operated manually. Due to the manual operation, the velocities of the helicopter during the experiments vary according to the header of Table 6.4. Each presented experiment was operated in parts of the map with enough reference data, as the proposed approach is based on map registration procedures.

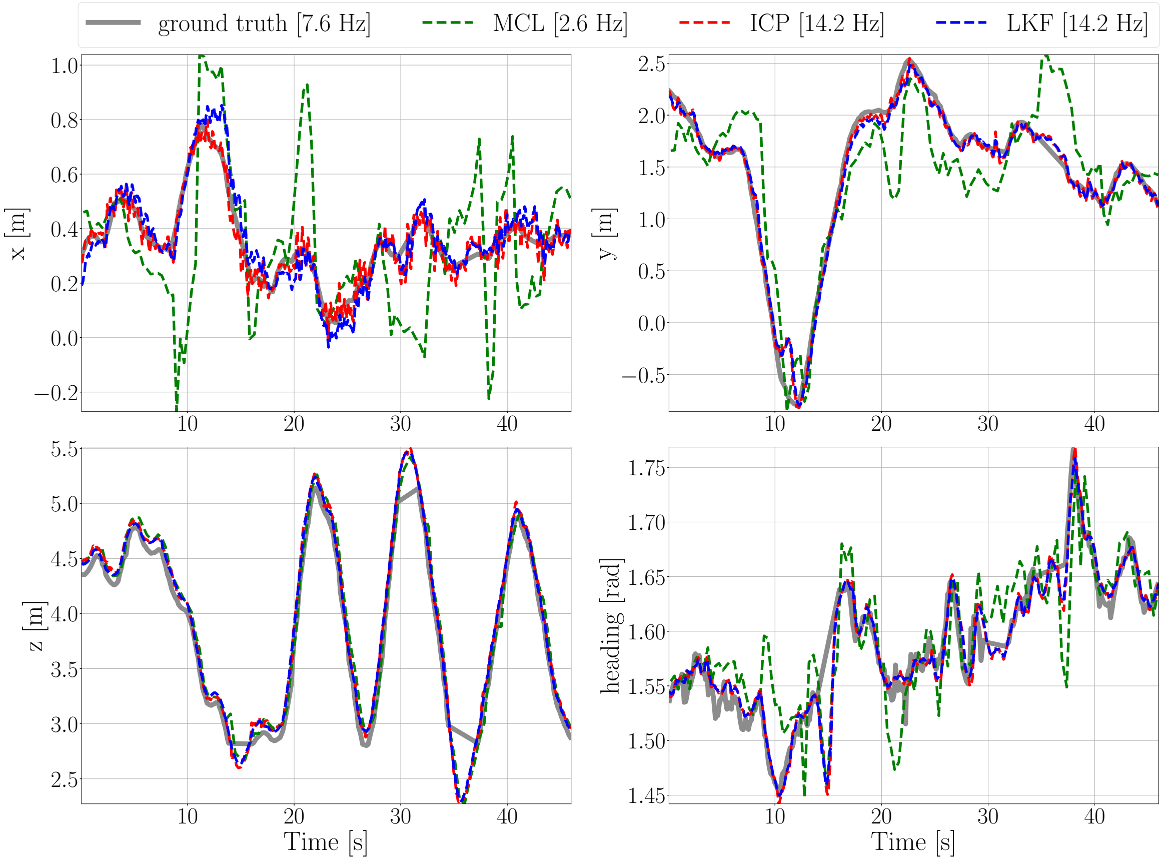
In Table 6.4, the quantitative analysis of the proposed localization system on real data is presented for MCL, ICP, and LKF state estimation separately. The presented experimental verification shows the performance of the system similar to the simulation verification in Section 6.2. In each experiment, the system is capable of state estimation and its following during the UAV movement. The three presented offline evaluations on experimental data are summarized by the following list. The resulting state estimation history during each evaluation on experimental data, together with position ground truth, is embedded within a map in Figure 6.14.

- Figure 6.11: First experimental flight contains oscillations of the UAV altitude and small heading changes. The visual trajectory between the total station and the onboard tracked reflector was occluded at times around 15 s, 30 s and 35 s, leading to lack of ground truth data at these particular time periods.

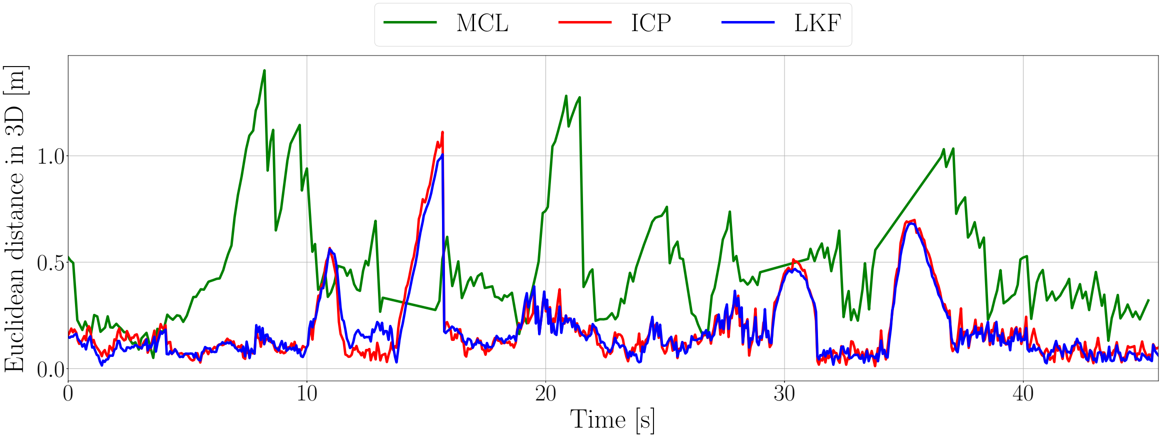
- Figure 6.12: Second experimental flight contains the takeoff phase of the flight. It shows the convergence of the MCL into the global optimum and initialization of the ICP procedure at altitude of approx. 2.3 m. The experiment also shows capabilities to track the lateral motion of a UAV.

- Figure 6.13: Third experimental flight shows drawbacks of the MCL global estimation. During the experiment, the MCL state estimation update rate is 2.2 Hz on average. Such a low rate leads to loss of the tracking capabilities during quick state changes. Particularly at the time around 22 s, motion of the vehicle accelerated in each axis, leading to a velocity peak and loss of tracking capabilities of MCL state estimation.

The quantitative analysis of the localization system, given in Table 6.4, shows estimation accuracy with translational RMSE less than 0.25 m during each experiment. The experiments also show minimal delay and smoothness of the final state tracking estimate. Both of these parameters are important for deployment onto a real UAV platform since both could destabilize control of the UAV. In conclusion, the proposed localization system proved to be a reliable and robust source with sufficient precision of the position estimate.

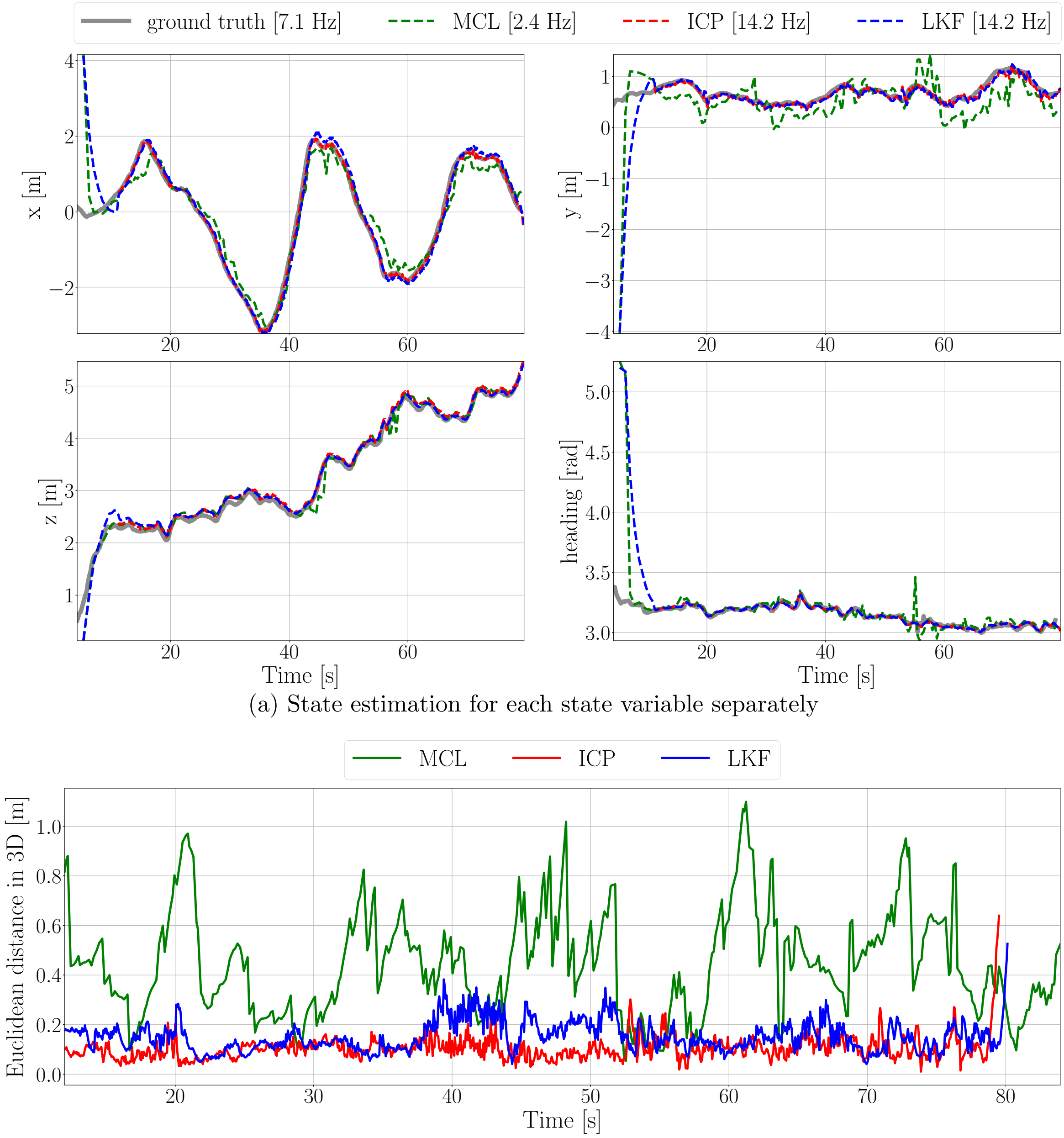


(a) State estimation for each state variable separately



(b) Euclidean distance to the ground truth reference during the entire experiment

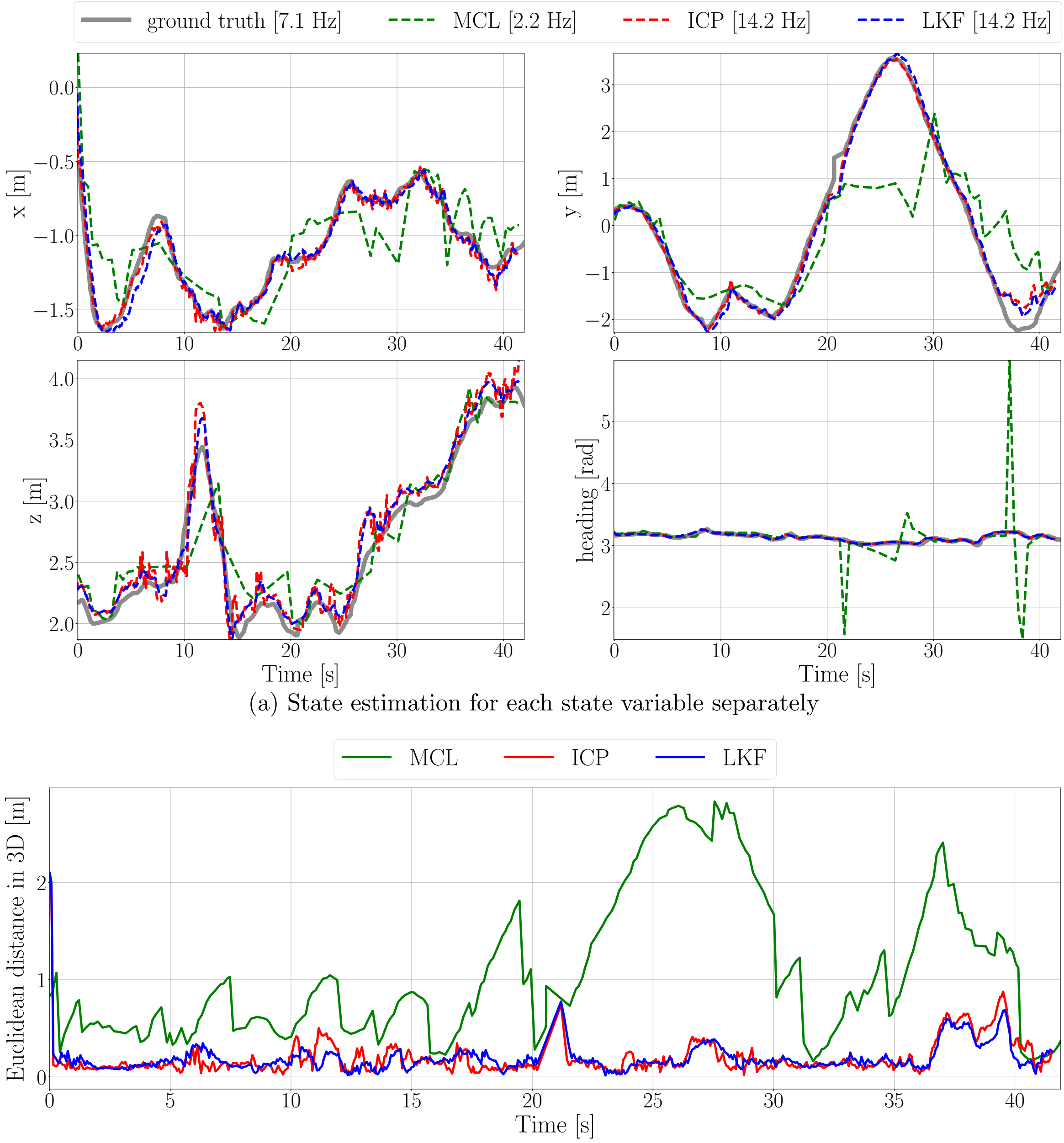
Figure 6.11: Experimental verification of the localization system during a manual flight. Ground truth reference is interrupted around 15 s, 30 s and 35 s due to visual occlusions between a total station and the tracked target.



(a) State estimation for each state variable separately

(b) Euclidean distance to the ground truth reference

Figure 6.12: Experimental verification of the localization system during second manual flight. The verification contains takeoff phase of the flight, where MCL estimation convergence and ICP procedure initialization is visible at altitude of approx. 2.3 m.



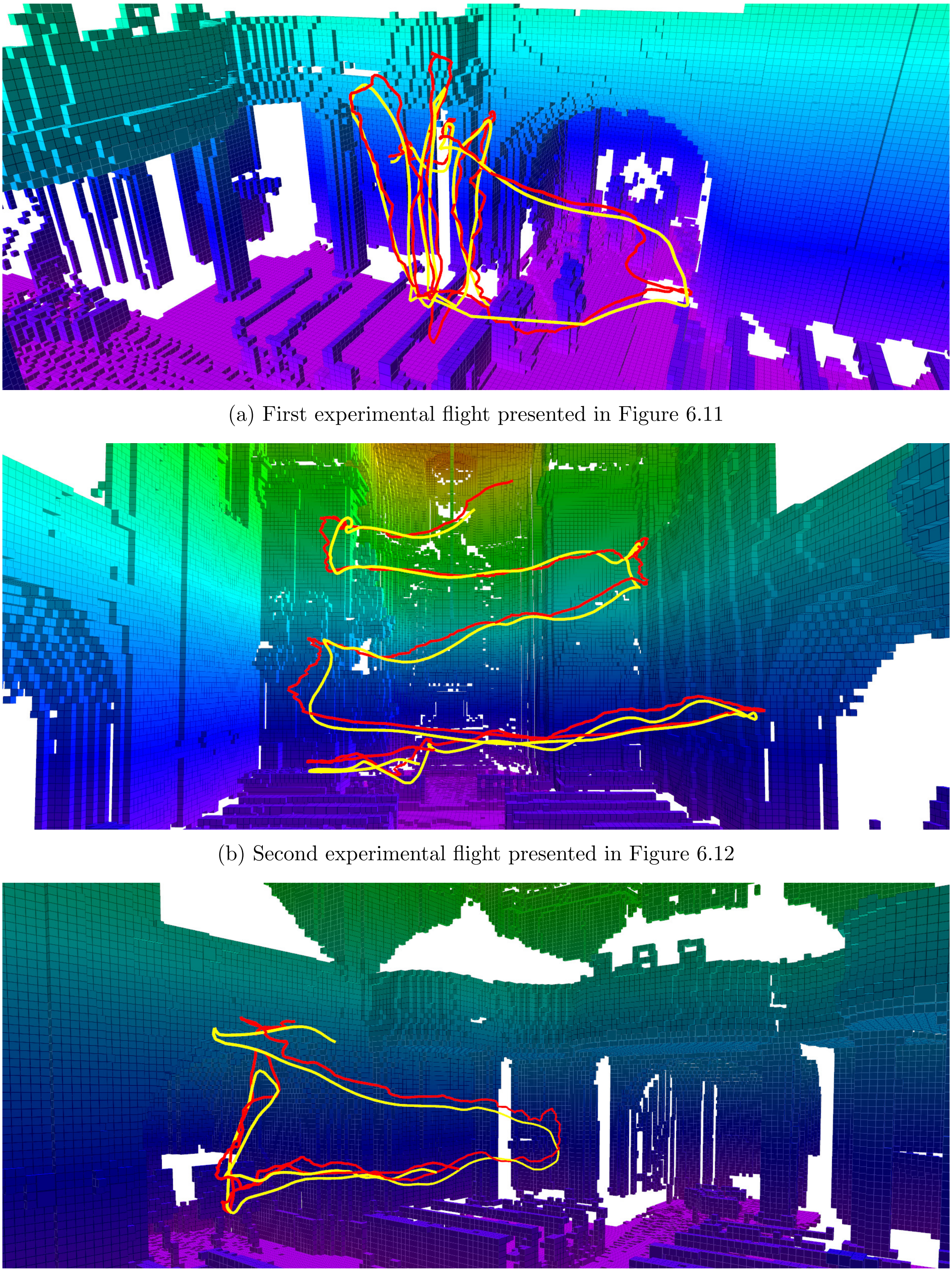
(a) State estimation for each state variable separately

(b) Euclidean distance to the ground truth reference

Figure 6.13: Experimental verification of the localization system during third manual flight. The experiment contains losses of the global MCL estimation due to low update rate of the algorithm and fast motion of the UAV.

Trajectory	Experiment 1	Experiment 2	Experiment 3
length [m]	45.812	24.055	21.163
avg linear velocity [$\mathrm{m\,s^{-1}}$]	0.485	0.361	0.505
max linear velocity [$\mathrm{m\,s^{-1}}$]	2.294	1.586	1.734
Figure	Figure 6.11	Figure 6.12	Figure 6.13
ATE translation [m]			
MCL	0.476	0.407	0.946
ICP	0.108	0.091	0.214
LKF	0.108	0.168	0.214
ATE heading [°]			
MCL	0.069	3.595	21.605
ICP	2.494	2.388	2.770
LKF	3.353	2.395	2.750
RMSE translation [m]			
MCL	0.419	0.440	0.946
ICP	0.143	0.117	0.234
LKF	0.140	0.179	0.230
RMSE heading [°]			
MCL	3.411	3.193	21.159
ICP	2.483	6.791	2.739
LKF	2.460	2.381	2.747
max translation error [m]			
MCL	1.248	0.920	2.881
ICP	0.573	0.271	0.765
LKF	0.522	0.385	0.594
max heading error [°]			
MCL	8.184	9.105	87.135
ICP	6.962	6.791	11.331
LKF	6.928	6.807	11.302

Table 6.4: Summary of the state estimate accuracy for MCL, ICP and LKF during experimental verification during manual flights in Church of St. Mary Magdalene in Chlumín. LKF estimate yields the final system state estimate. Due to the ground truth reference interrupts, specified in Section 6.3, maximal errors are calculated from data with available ground truth reference only.



(a) First experimental flight presented in Figure 6.11

(b) Second experimental flight presented in Figure 6.12

(c) Third experimental flight presented in Figure 6.13

Figure 6.14: Verification on real data – 3D position trajectories visualization (yellow: ground truth, red: final state estimate)

Chapter 7: Position Control

Suitability of the presented localization system for deployment into a UAV position control feedback is herein discussed. This chapter is also complemented by multimedia materials available at `http://mrs.felk.cvut.cz/theses/petracek2019`. The localization system is integrated into the UAV control pipeline developed in the Multi-Robot Systems group[1]. The control pipeline is depicted in Figure 7.1, which extends the system architecture from Figure 3.3 by detailed overview of the Act component. A mission planner provides a reference setpoint for the model predictive controller (MPC) in the MPC tracker [36]. The MPC tracker outputs position, velocity and acceleration commands at 100 Hz handled by the non-linear SO(3) controller. The SO(3) controller outputs optimal angular velocities and thrust commands for an embedded attitude controller responsible for maintaining the desired attitude. Detailed description of the control system can be found in [35, 36].

The position feedback integration is verified exclusively in the simulation environment, which was introduced in Section 6.2. Control feedback from the state estimation module to the MPC tracker and non-linear SO(3) state feedback controller is established. In order to obtain a 6 degrees-of-freedom pose estimate, the state observer module concatenates the 3D position and the heading from the state estimate presented in Chapter 5, and tilt angles roll and pitch from the attitude controller. The proposed UAV platform is equipped with the PX4 stack running on Pixhawk attitude controller [81], although the UAV controllers are independent on type of the utilized flight controller. Supplying exclusively the position estimation to the SO(3) controller yielded slow and oscillatory response of the system. Hence, the UAV velocities are provided back from the autopilot to the SO(3) controller to improve the trajectory tracking system performance.

To verify the functionality of the position control loop, three experiments are presented. First in Figure 7.2a, multiple lateral reference setpoints are given to the MPC tracker. Second in Figure 7.2b, the trajectory follows the same lateral reference setpoints, although it oscillates in altitude and heading of the UAV. In both simulations, the references are successfully followed by the UAV showing the capabilities of the system to supply precise and reliable feedback information utilized for position control of the UAV. Third in Figure 7.2c, the desired reference is a circular trajectory over a set of obstacles (church benches) with the heading of the UAV oriented into the center of the trajectory. Likewise, the experiment proves the position control feedback functionality.

The constraints of precise and slow UAV dynamics slow down the response of the control system, as specifically evident in Figure 7.2c. Appropriate parameters (position and velocity gains) of the control system with the proposed system in the feedback loop can be tuned for faster response of the trajectory tracking.

[1] Czech Technical University in Prague, Faculty of Electrical Engineering

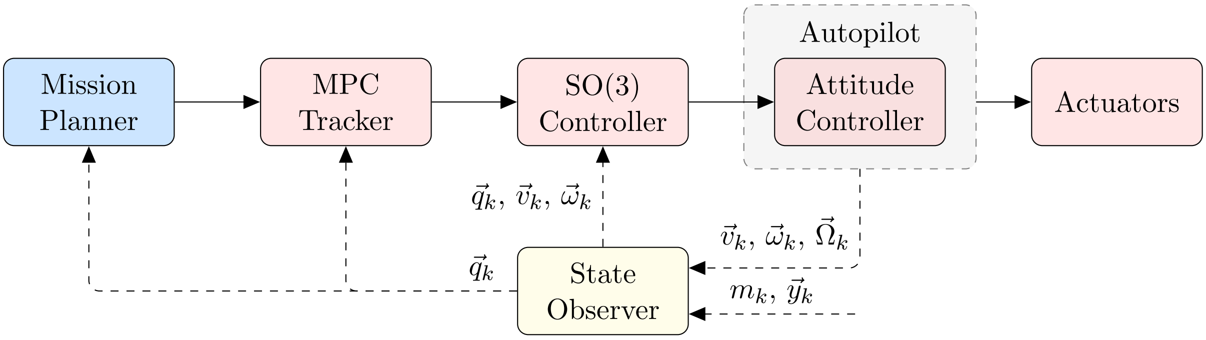
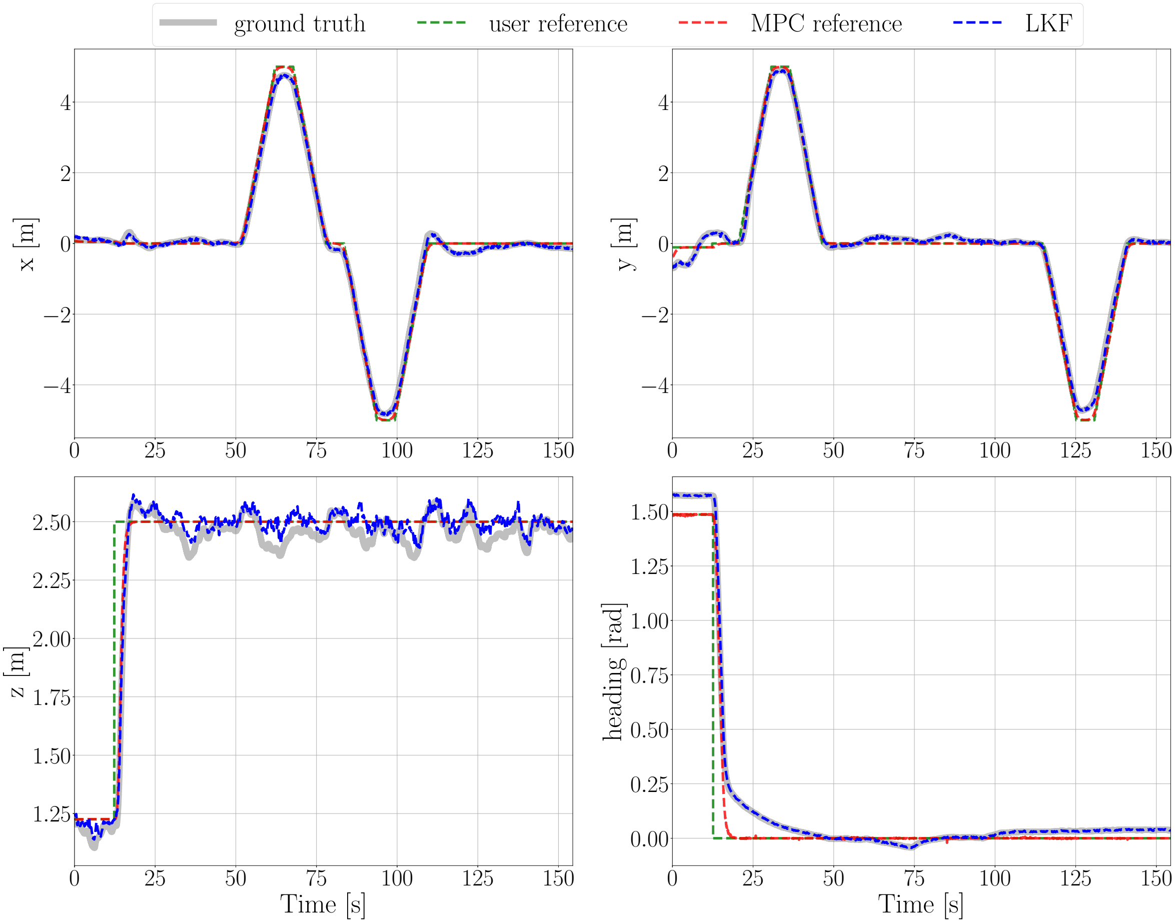
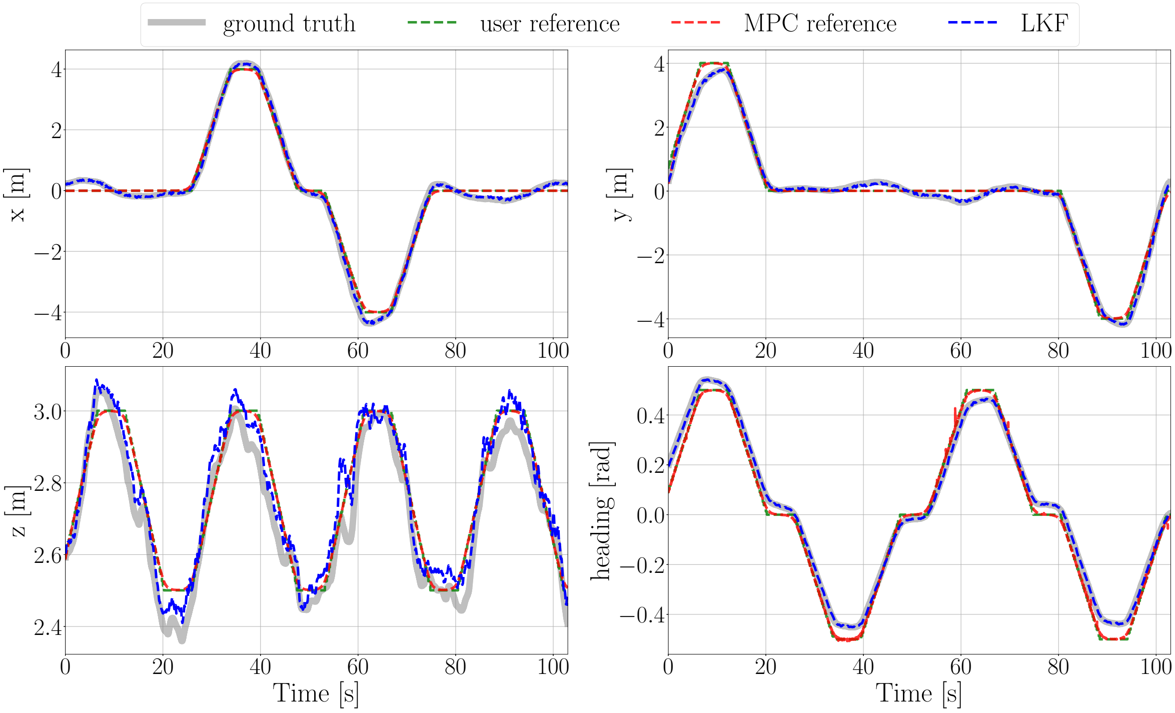


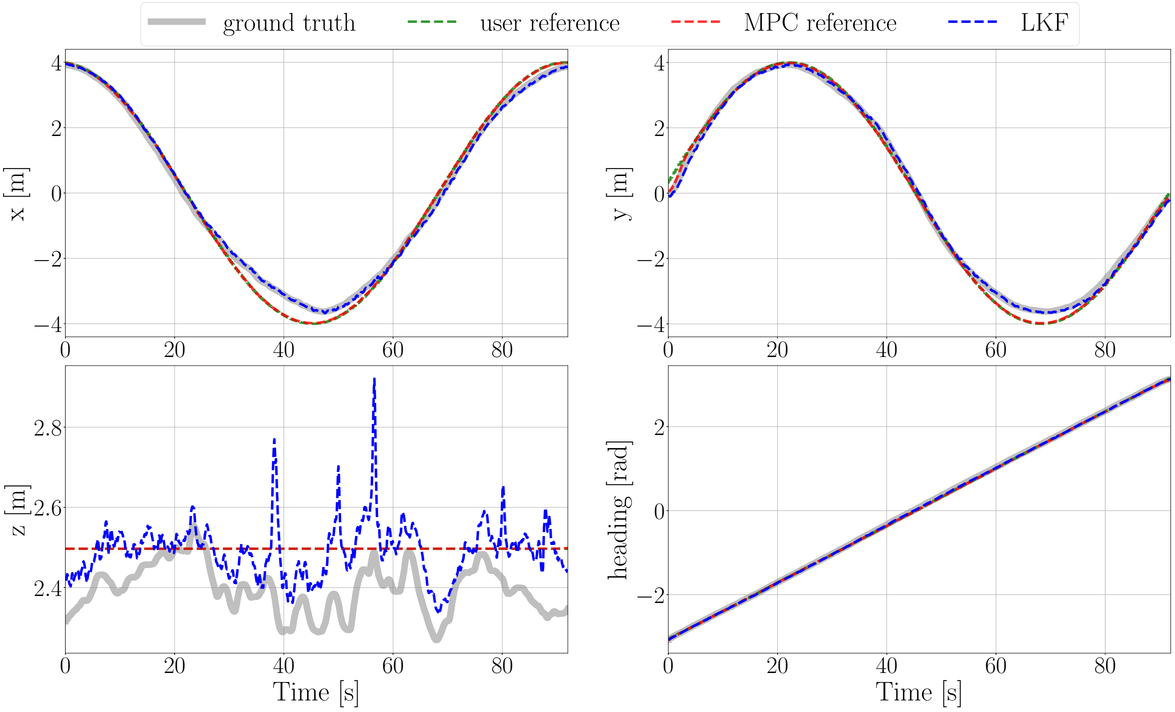
Figure 7.1: Diagram of the control pipeline of a UAV with detailed description of the Act component from Figure 3.3. At time step k, inputs of the state observer are a map m_k, sensors observations $\vec{y}_k$, linear $\vec{v}_k$ and angular $\vec{\omega}_k$ velocities of the UAV, and attitude $\vec{\Omega}_k$ of the UAV. The output is the pose $\vec{q}_k$ of the UAV, concatenated from a state estimate $\vec{x}_k$ and tilt angles roll and pitch given by $\vec{\Omega}_k$. Apart from the pose of the UAV, the velocities estimated by the autopilot are provided to the SO(3) controller to support the position control of the UAV.



(a) Multiple reference setpoints tracking



(b) Multiple reference setpoints tracking with oscillations in altitude and heading



(c) Circular trajectory tracking with center-oriented heading

Figure 7.2: Separate state variables during simulations with the proposed localization system integrated into the UAV position feedback of the Multi-Robot Systems group control pipeline

Chapter 8: Conclusion

Contents

In this thesis, a hardware and software solution of a specialized UAV platform was developed for documentation of historical monuments without access to a global navigation system. The application-tailored platform was designed, manufactured and already deployed in documentation task of Church of St. Mary Magdalene in Chlumín. Furthermore, a self-localization system in a priori generated map was developed. The system pipeline consists of generating and processing of a map, processing of onboard sensory data, and fusion of IMU, global Monte Carlo Localization and local Iterative Closest Point state estimation methods. The proposed localization solution was successfully integrated into the UAV control system of the Multi-Robot Systems group at FEE CTU. The accuracy of the system was verified in the simulation environment and on real-flight data recorded during flights with a precise position ground truth. The results show the capabilities to provide real-time state estimation with position RMSE less than 25 cm. The state estimate was integrated into the position feedback control loop of a UAV, which was tested in the simulation environment. Many experiments have been conducted in realistic scenarios of a historical building demonstrating capabilities of the system to precisely estimate state of the UAV in 3D space. The thesis is complemented by multimedia materials available at `http://mrs.felk.cvut.cz/theses/petracek2019`.

The entire assignment of this thesis has been fulfilled successfully. According to the assignment, the following tasks have been completed.

- Design, production and testing of a specialized UAV platform respecting requirements for deployment in historical monuments was presented in Chapter 2.
- A method for stabilization and localization in a map without access to an external localization service was developed and implemented in Chapter 5. Methods for generating of such map were described in Chapter 4.
- The proposed self-localization system was verified and validated in a realistic simulation environment in Section 6.2 and on real-flight datasets labeled by a precise position ground truth in Section 6.4.
- The proposed self-localization system was integrated into the closed-loop control pipeline of the Multi-Robot Systems group and tested in a realistic simulation environment in Chapter 7.

8.1 Future Work

Foremost, the UAV platform shall be complemented by a set of supporting vehicles capable of self-localization and conveying of light with tilting capabilities.

During development of the localization system, several ideas improving its functionality emerged. First, the generated maps by a terrestrial laser scanner contain large holes in the resulting output data due to visual occlusions between scanning locations and a scanned object. The system shall be complemented with map refinement mechanisms to fill these openings from aerial data. Second, the state estimation mechanism shall be extended with sequential scan matching to provide accurate velocity estimation. Third, dynamic obstacles shall be introduced into the localization system to consider an influence of other agents during a formation flight.

Regarding sources of state estimation, multiple other techniques shall be integrated into the fusion of various localization sources. That includes mainly down- and front-oriented optic-flow estimations useful in environments with appropriate lighting conditions and sparse obstacle density.

Bibliography

[1] P. Merriaux, Y. Dupuis, R. Boutteau, P. Vasseur, and X. Savatier, “A Study of Vicon System Positioning Performance,” *Sensors*, vol. 17, no. 7, 2017.

[2] M. Windolf, N. Götzen, and M. Morlock, “Systematic accuracy and precision analysis of video motion capturing systems – Exemplified on the Vicon-460 system,” *Journal of biomechanics*, vol. 41, pp. 2776–80, 02 2008.

[3] D. Kang and Y.-J. Cha, “Autonomous UAVs for Structural Health Monitoring Using Deep Learning and an Ultrasonic Beacon System with Geo-Tagging,” *Computer-Aided Civil and Infrastructure Engineering*, 05 2018.

[4] M. Nitsche, T. Krajník, P. Čížek, M. Mejail, and T. Duckett, “WhyCon: An Efficent, Marker-based Localization System,” 01 2015.

[5] P. Lightbody, T. Krajník, and M. Hanheide, “An Efficient Visual Fiducial Localisation System,” *SIGAPP Appl. Comput. Rev.*, vol. 17, no. 3, pp. 28–37, Nov. 2017.

[6] T. Krajník, M. Nitsche, J. Faigl, P. Vaněk, M. Saska, L. Přeučil, T. Duckett, and M. Mejail, “A Practical Multirobot Localization System,” *Journal of Intelligent & Robotic Systems*, vol. 76, no. 3, pp. 539–562, Dec 2014.

[7] V. Walter, M. Saska, and A. Franchi, “Fast mutual relative localization of uavs using ultraviolet led markers,” in *2018 International Conference of Unmanned Aircraft System (ICUAS 2018)*, 2018.

[8] V. Walter, N.Staub, M. Saska, and A. Franchi, “Mutual Localization of UAVs based on Blinking Ultraviolet Markers and 3D Time-Position Hough Transform,” in *14th IEEE International Conference on Automation Science and Engineering (CASE 2018).*

[9] V. Walter, N. Staub, A. Franchi, and M. Saska, “UVDAR System for Visual Relative Localization With Application to Leader–Follower Formations of Multirotor UAVs,” *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2637–2644, July 2019.

[10] Dronument. (2019) Dronument. Accessed on May 18, 2019. [Online]. Available: https://dronument.cz/

[11] M. Saska, V. Kratky, V. Spurny, and T. Baca, “Documentation of dark areas of large historical buildings by a formation of unmanned aerial vehicles using model predictive control,” in *IEEE ETFA*, 2017.

[12] M. Sauerbiera and H. Eisenbeissb, “UAVS for the documentation of archaeological excavations,” *Part 5 Commission V Symposium*, vol. 38, pp. 526–531, 05 2019.

[13] F. Remondino, A. Gruen, J. von Schwerin, H. Eisenbeiss, A. Rizzi, S. Girardi, M. Sauerbier, and H. Richards-Rissetto, “Multi-sensor 3D documentation of the Maya site of Copan,” 10 2009.

[14] D. C. Cowley, C. Moriarty, G. Geddes, G. L. Brown, T. Wade, and C. J. Nichol, “UAVs in Context: Archaeological Airborne Recording in a National Body of Survey and Record,” *Drones*, vol. 2, no. 1, 2018.

[15] D. Dominici, M. Alicandro, E. Rosciano, and V. Massimi, “Multiscale documentation and monitoring of L’Aquila historical centre using UAV photogrammetry,” *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLII-5/W1, pp. 365–371, 05 2017.

[16] I. Ma’arof, S. Z. Bahari, Z. A. Latif, N. A. Sulaiman, and A. M. Samad, “Image based modeling and documentation of Malaysian historical monuments using Digital Close-Range Photogrammetry (DCRP),” in *2013 IEEE International Conference on Control System, Computing and Engineering*, Nov 2013, pp. 424–429.

[17] Z. Xu, L. Wu, Y. Shen, F. Li, Q. Wang, and R. Wang, “Tridimensional Reconstruction Applied to Cultural Heritage with the Use of Camera-Equipped UAV and Terrestrial Laser Scanner,” *Remote Sensing*, vol. 6, no. 11, pp. 10 413–10 434, 2014.

[18] M. Nieuwenhuisen, J. Quenzel, M. Beul, D. Droeschel, S. Houben, and S. Behnke, “ChimneySpector: Autonomous MAV-based indoor chimney inspection employing 3D laser localization and textured surface reconstruction,” in *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*, June 2017, pp. 278–285.

[19] “A UAV for bridge inspection: Visual servoing control law with orientation limits,” *Automation in Construction*, vol. 17, no. 1, pp. 3 – 10, 2007.

[20] L. Dowling, T. Poblete, I. Hook, H. Tang, Y. Tan, W. Glenn, and R. R Unnithan, “Accurate indoor mapping using an autonomous unmanned aerial vehicle (UAV),” 07 2018.

[21] J. Tiemann, A. Ramsey, and C. Wietfeld, “Enhanced UAV Indoor Navigation through SLAM-Augmented UWB Localization,” in *2018 IEEE International Conference on Communications Workshops (ICC Workshops)*, May 2018, pp. 1–6.

[22] J.-C. Trujillo, R. Munguia, E. Guerra, and A. Grau, “Visual-Based SLAM Configurations for Cooperative Multi-UAV Systems with a Lead Agent: An Observability-Based Approach,” *Sensors*, vol. 18, no. 12, 2018.

[23] ——, “Cooperative Monocular-Based SLAM for Multi-UAV Systems in GPS-Denied Environments,” *Sensors*, vol. 18, no. 5, 2018.

[24] A. L. Majdik, L. Tizedes, M. Bartus, and T. Szirányi, “Photogrammetric 3D reconstruction of the old slaughterhouse in Budapest,” in *2016 International Workshop on Computational Intelligence for Multimedia Understanding (IWCIM)*, Oct 2016, pp. 1–5.

[25] K. Schmid, P. Lutz, T. Tomić, E. Mair, and H. Hirschmüller, "Autonomous Vision-based Micro Air Vehicle for Indoor and Outdoor Navigation," *Journal of Field Robotics*, vol. 31, no. 4, pp. 537–570.

[26] S. Grzonka, G. Grisetti, and W. Burgard, "A Fully Autonomous Indoor Quadrotor," *IEEE Transactions on Robotics*, vol. 28, pp. 90–100, 02 2012.

[27] R. Fanti, G. Gigli, L. Lombardi, D. Tapete, and P. Canuti, "Terrestrial laser scanning for rockfall stability analysis in the cultural heritage site of Pitigliano (Italy)," *Landslides*, vol. 10, no. 4, pp. 409–420, Aug 2013.

[28] W. Sheng, A. Okamoto, and S. Tanaka, "Visual Point-Based Analysis of Laser-Scanned Historical Structures," in *2015 International Conference on Culture and Computing (Culture Computing)*, Oct 2015, pp. 47–53.

[29] F. J. Perez-Grau, R. Ragel, F. Caballero, A. Viguria, and A. Ollero, "An architecture for robust UAV navigation in GPS-denied areas," *J. Field Robotics*, vol. 35, pp. 121–145, 2018.

[30] Z. J. Chong, B. Qin, T. Bandyopadhyay, M. H. Ang, E. Frazzoli, and D. Rus, "Synthetic 2D LIDAR for precise vehicle localization in 3D urban environment," in *2013 IEEE International Conference on Robotics and Automation*, May 2013, pp. 1554–1559.

[31] P. J. Besl and N. D. McKay, "A method for registration of 3-D shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, Feb 1992.

[32] G. A. Kumar, A. K. Patil, R. Patil, S. S. Park, and Y. H. Chai, "A LiDAR and IMU Integrated Indoor Navigation System for UAVs and Its Application in Real-Time Pipeline Classification," *Sensors*, vol. 17, no. 6, 2017.

[33] F. Wang, K. Wang, S. Lai, S. K. Phang, B. Chen, and T. H. Lee, "An efficient UAV navigation solution for confined but partially known indoor environments," 06 2014, pp. 1351–1356.

[34] D. Holz, M. Nieuwenhuisen, D. Droeschel, M. Schreiber, and S. Behnke, "Towards Multimodal Omnidirectional Obstacle Detection for Autonomous Unmanned Aerial Vehicles," *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, no. 2, pp. 201–206, Aug. 2013.

[35] T. Baca, G. Loianno, and M. Saska, "Embedded Model Predictive Control of Unmanned Micro Aerial Vehicles," in *2016 IEEE International Conference on Methods and Models in Automation and Robotics (MMAR)*, Miedzyzdroje, Poland, 2016.

[36] T. Baca, D. Hert, G. Loianno, M. Saska, and V. Kumar, "Model Predictive Trajectory Tracking and Collision Avoidance for Reliable Outdoor Deployment of Unmanned Aerial Vehicles," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1–8.

[37] V. Spurny, T. Baca, M. Saska, R. Penicka, T. Krajnik, J. Thomas, D. Thakur, G. Loianno, and V. Kumar, "Cooperative Autonomous Search, Grasping and Delivering in a Treasure Hunt Scenario by a Team of UAVs," *Published in Journal of Field Robotics*, 2018.

[38] G. Loianno, V. Spurny, T. Baca, J. Thomas, D. Thakur, T. Krajnik, A. Zhou, A. Cho, M. Saska, and V. Kumar, "Localization, Grasping, and Transportation of Magnetic Objects by a team of MAVs in Challenging Desert like Environments," *IEEE Robotics and Automation Letters*, 2018.

[39] T. Baca, P. Stepan, and M. Saska, "Autonomous Landing On A Moving Car With Unmanned Aerial Vehicle," in *The European Conference on Mobile Robotics (ECMR)*, 2017.

[40] V. Walter, T. Novák, and M. Saska, "Self-localization of Unmanned Aerial Vehicles Based on Optical Flow in Onboard Camera Images," in *Modelling and Simulation for Autonomous Systems*. Cham: Springer International Publishing, 2018, pp. 106–132.

[41] J. Xu, "Design Perspectives on Delivery Drones," in *Modelling and Simulation for Autonomous Systems*. CA: RAND Corporation, 2017.

[42] M. Beul, N. Krombach, Y. Zhong, D. Droeschel, M. Nieuwenhuisen, and S. Behnke, "A high-performance MAV for autonomous navigation in complex 3D environments," in *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, June 2015, pp. 1241–1250.

[43] B. Theys, G. Dimitriadis, P. Hendrick, and J. De Schutter, "Influence of propeller configuration on propulsion system efficiency of multi-rotor Unmanned Aerial Vehicles," in *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, June 2016, pp. 195–201.

[44] A. D. Team. (2019) Octorotor X8 configuration. Accessed on March 21, 2019. [Online]. Available: http://ardupilot.org/copter/docs/connect-escs-and-motors.html#connect-escs-and-motors3

[45] S. for All Markus Mueller. (2019) eCalc online tool (free non-commercial license). Accessed on March 21, 2019. [Online]. Available: https://www.ecalc.ch/

[46] (2019) MAVLink protocol. Accessed on March 26, 2019. [Online]. Available: https://mavlink.io/en/

[47] L. Coles, C. A. Rosen, B. Raphael, T. Garvey, and R. O. Duda, "Application of Intelligent Automata to Reconnaissance," p. 159, 11 1969.

[48] (2019) Leica Nova MS60 MultiStation. Accessed on Apr 08, 2019. [Online]. Available: https://leica-geosystems.com/products/total-stations/multistation/leica-nova-ms60

[49] (2019) Leica BLK360 Imaging Laser Scanner. Accessed on Apr 08, 2019. [Online]. Available: https://leica-geosystems.com/en-SG/products/laser-scanners/scanners/blk360

[50] O. Kořínek, "Church of St. Wenceslas exterior, Smíchov, Prague," Online, 2012, Licensed under CC3.0, Accessed on May 21, 2019. [Online]. Available: https://cs.wikipedia.org/wiki/Soubor:Kostel_svat%C3%A9ho_V%C3%A1clava_na_Sm%C3%ADchov%C4%9B_(10).jpg

[51] J. Groh, “Church of St. Wenceslas interior, Smíchov, Prague,” Online, 2010, Licensed under CC3.0, Accessed on May 21, 2019. [Online]. Available: https://cs.wikipedia.org/wiki/Soubor:Sm%C3%ADchov,_kostel_svat%C3%A9ho_V%C3%A1clava,_interi%C3%A9r.jpg

[52] D. Meagher, “Octree Encoding: A New Technique for the Representation, Manipulation and Display of Arbitrary 3-D Objects by Computer,” in *Technical Report IPL-TR-80-111*.

[53] J. Revelles, C. Ureña, and M. Lastra, “An Efficient Parametric Algorithm for Octree Traversal,” in *Journal of WSCG*, 2000, pp. 212–219.

[54] Wikipedia.org, “Schematic drawing of an octree, a data structure of computer science.” Online, 2010, Licensed under CC3.0, Accessed on May 21, 2019. [Online]. Available: https://commons.wikimedia.org/wiki/File:Octree2.svg

[55] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, “OctoMap: An Efficient Probabilistic 3D mapping framework based on octrees,” *Autonomous Robots*, 2013, software available at http://octomap.github.com.

[56] A. Knoll, I. Wald, S. Parker, and C. Hansen, “Interactive isosurface ray tracing of large octree volumes,” in *2006 IEEE Symposium on Interactive Ray Tracing*, 2006, pp. 115–124.

[57] D. Fox, W. Burgard, and S. Thrun, “Markov Localization for Mobile Robots in Dynamic Environments,” *J. Artif. Int. Res.*, vol. 11, no. 1, pp. 391–427, Jul. 1999.

[58] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, “Monte Carlo localization for mobile robots,” in *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*, vol. 2, May 1999, pp. 1322–1328 vol.2.

[59] A. H. Jazwinski, *Stochastic Processes and Filtering Theory.* Academic Press, 1970.

[60] Z. Chen, “Bayesian Filtering: From Kalman Filters to Particle Filters, and Beyond,” *Statistics*, vol. 182, 01 2003.

[61] S. Thrun, “Particle Filters in Robotics,” in *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence*, ser. UAI’02, 2002, pp. 511–518.

[62] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents).* The MIT Press, 2005.

[63] A. L. Ballardini and et. al., “An effective 6DoF motion model for 3D-6DoF Monte Carlo Localization,” in *4th Workshop on Planning, Perception and Navigation for Intelligent Vehicles*, ser. IROS, 2012.

[64] J. H. Friedman, J. L. Bentley, and R. A. Finkel, “An Algorithm for Finding Best Matches in Logarithmic Expected Time,” *ACM Trans. Math. Softw.*, vol. 3, no. 3, pp. 209–226, Sep. 1977.

[65] D. Fox, “KLD-Sampling: Adaptive Particle Filters and Mobile Robot Localization,” 10 2001.

[66] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm," in *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*, May 2001, pp. 145–152.

[67] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 9-13 2011.

[68] D. Holz, A. E. Ichim, F. Tombari, R. B. Rusu, and S. Behnke, "Registration with the Point Cloud Library: A Modular Framework for Aligning in 3-D," *IEEE Robotics Automation Magazine*, vol. 22, no. 4, pp. 110–124, Dec 2015.

[69] M. Muja and D. G. Lowe, "Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration," in *International Conference on Computer Vision Theory and Application VISSAPP'09), publisher = INSTICC Press, year = 2009, pages = 331-340.*

[70] ——, "Scalable Nearest Neighbor Algorithms for High Dimensional Data," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 36, 2014.

[71] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-Squares Fitting of Two 3-D Point Sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, no. 5, pp. 698–700, Sep. 1987.

[72] B. K. P. Horn, "Closed-form solution of absolute orientation using unit quaternions," *J. Opt. Soc. Am. A*, vol. 4, no. 4, pp. 629–642, Apr 1987.

[73] B. K. P. Horn, H. M. Hilden, and S. Negahdaripour, "Closed-form solution of absolute orientation using orthonormal matrices," *J. Opt. Soc. Am. A*, vol. 5, no. 7, pp. 1127–1135, Jul 1988.

[74] M. W. Walker, L. Shao, and R. A. Volz, "Estimating 3-D location parameters using dual number quaternions," *CVGIP: Image Understanding*, vol. 54, pp. 358–367, 1991.

[75] D. Eggert, A. Lorusso, and R. Fisher, "Estimating 3-D rigid body transformations: a comparison of four major algorithms," *Machine Vision and Applications*, vol. 9, no. 5, pp. 272–290, Mar 1997.

[76] Z. Zhang and D. Scaramuzza, "A Tutorial on Quantitative Trajectory Evaluation for Visual(-Inertial) Odometry," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, 2018.

[77] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," 10 2012, pp. 573–580.

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

[79] "ROS: an open-source Robot Operating System," in *ICRA Workshop on Open Source Software*, 2009.

[80] (2009) Surveying reflectors - white paper, Characteristics and influences. Accessed on May 09, 2019. [Online]. Available: https://accessories.leica-geosystems.com/downloads123/zz/accessory/accessories/white-tech-paper/White%20Paper%20Surveying%20Reflectors_en.pdf

[81] L. Meier, D. Honegger, and M. Pollefeys, "PX4: A node-based multithreaded open source robotics framework for deeply embedded platforms," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 6235–6240.

Appendices

CD Content

Table 1 lists arrangement of all directories on the attached CD.

Directory name	**Description**
thesis	the thesis in pdf format
sources/thesis	latex source codes
sources/indoor_localization	software source codes
media	multimedia materials

Table 1: CD Content

List of abbreviations

Table 2 lists abbreviations used in this thesis.

Abbreviation	Meaning
UAV	Unmanned Aerial Vehicle
FEE CTU	Faculty of Electrical Engineering, Czech Technical University in Prague
MRS	Multi-Robot Systems group at FEE CTU
LiDAR	Light Detection and Ranging
ROS	Robot Operating System
MBZIRC	Mohamed Bin Zayed International Robotics Challenge
MPC	Model Predictive Control
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
ESC	Electronic Speed Controller
LiPo	Lithium-Polymer accumulator
PWM	Pulse Width Modulation
FPV	First Person View
FCU	Flight Control Unit
IMU	Inertial Measurement Unit
MLE	Maximum Likelihood Estimate
ICP	Iterative Closest Point
LKF	Linear Kalman Filter
ODE	Open Dynamics Engine
ATE	Absolute Trajectory Error
RPE	Relative Pose Error
ATR	Automatic Target Recognition
MSE	Mean Squared Error
RMSE	Root Mean Squared Error

Table 2: Lists of abbreviations