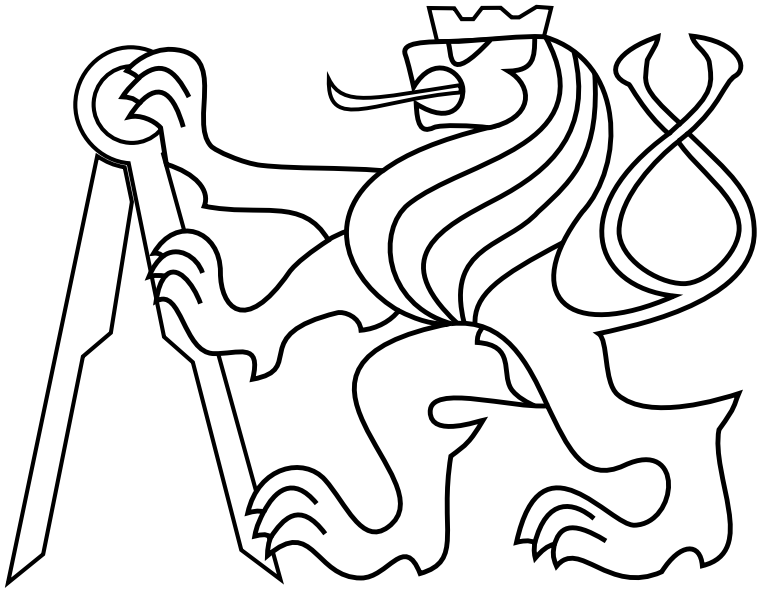
CZECH TECHNICAL UNIVERSITY IN PRAGUE

Faculty of Electrical Engineering

BACHELOR'S THESIS



Lidar and multi-camera calibration and fusion

Martin Fischer

Thesis supervisor: **Ing. Pavel Petráček**

Department of Cybernetics

May 2021

Author statement for undergraduate thesis

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

Prague, date ....

Signature

BACHELOR'S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Fischer Martin** Personal ID number: **483507**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Cybernetics**

Study program: **Cybernetics and Robotics**

II. Bachelor's thesis details

Bachelor's thesis title in English:

Lidar and Multi-Camera Calibration and Fusion

Bachelor's thesis title in Czech:

Kalibrace a fúze lidaru a vícekamerového senzoru

Guidelines:

The aim of this thesis is to implement a methodology to mutually calibrate lidar and multi-camera sensory setup [1, 2], and to develop and analyse a process to fuse the data onboard an uncrewed aerial vehicle. The following tasks will be solved:
1) Familiarize yourself with the Robot Operating System and with the Multi-Robot Systems group for stabilization and control of UAVs [3].
2) Describe, design, implement, and analyze the process of calibrating a three-dimensional lidar and multi-camera sensory setup.
3) Develop a method to fuse the point cloud data measured by a lidar with the RGB information from a set of cameras.
4) Implement a filtration of data fusion and analyse the influence on the performance of your methodology.
5) Analyse the performance of the developed algorithms on real-world datasets taken onboard an uncrewed aerial vehicle.

Bibliography / sources:

[1] Zoltán Pusztai, Iván Eichhardt and Levente Hajder "Accurate Calibration of Multi-LiDAR-Multi-Camera Systems," Sensors, 2018.
[2] X. Li, W. Guo, M. Li, C. Chen and L. Sun, "Generating colored point cloud under the calibration between TOF and RGB cameras," IEEE ICIA, 2013.
[3] T. Baca, et. al "The MRS UAV System: Pushing the Frontiers of Reproducible Research, Real-world Deployment, and Education with Autonomous Unmanned Aerial Vehicles", arXiv:2008.08050, 2020.

Name and workplace of bachelor's thesis supervisor:

Ing. Pavel Petráček, Department of Cybernetics, FEE

Name and workplace of second bachelor's thesis supervisor or consultant:

Date of bachelor's thesis assignment: **12.01.2021** Deadline for bachelor thesis submission: **21.05.2021**

Assignment valid until: **30.09.2022**

Ing. Pavel Petráček	prof. Ing. Tomáš Svoboda, Ph.D.	prof. Mgr. Petr Páta, Ph.D.
Supervisor's signature	Head of department's signature	Dean's signature

III. Assignment receipt

The student acknowledges that the bachelor's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the bachelor's thesis, the author must state the names of consultants and include a list of references.

.	
Date of assignment receipt	Student's signature

Acknowledgments

I would like to thank my supervisor Ing. Pavel Petráček for his patience, guidance, valuable advice, and improvements, which were very helpful for this thesis. I thank my family for their support throughout my studies. I am also grateful to all my classmates who helped me with study at the university, not only during a pandemic. Last but not least, I wish to thank my friend Tereza for spending her time correcting my mistakes in this work.

Abstract

This thesis deals with developing a method for a systematic fusion of the point cloud data measured by LiDAR with colour information from a set of cameras. Several filtration methods have been implemented to increase the robustness of the developed algorithm. The proposed solution is generally applicable for all robotic solutions. However, the main motivation is to equip all these sensors onboard an unmanned aerial vehicle and fuse the colour and the spatial information online mid-flight during an inspection mission. A process of calibration of the LiDAR and multi-camera sensors is also designed and implemented as one of the main challenges of the fusion process. With attention to finding the precise transformation, a calibration checkerboard pattern is used in this thesis. The developed methods are analysed and their performance is evaluated on data from simulation and real-world tests.

Keywords: unmanned aerial vehicle, LiDAR, camera, extrinsic calibration, sensor fusion, point cloud, ROS

Abstrakt

Tato práce se zabývá vývojem metody pro systematickou fúzi mračna bodů naměřeného LiDARem s barevnou informací ze sady kamer. Pro zvýšení robustnosti vyvinutého algoritmu bylo implementováno několik filtračních metod. Navržené řešení je obecně použitelné pro všechny robotické systémy. Hlavní motivací je ale použití těchto senzorů na palubě bezpilotní helikoptéry, kde budou data fúzována online během letu inspekční mise. Jako jedna z hlavních výzev fúze dat z rozdílných senzorů byla specifikována, navržena a implementována také kalibrace vnějších parametrů LiDARu a vícekamerového senzoru. Pro nalezení přesné transformace mezi senzory je v této práci použit šachovnicový vzor pro kalibraci. Vyvinuté metody jsou analyzovány a jejich chování je testováno na datech ze simulace i reálného světa.

Klíčová slova: bezpilotní helikoptéra, LiDAR, kamera, kalibrace vnějších parametrů, fúze senzorů, mračno bodů, ROS

Contents

List of Figures

List of Tables

Chapter 1: Introduction

Contents

In the field of autonomous robots multi-sensor systems are often used, containing radars, laser range-finders and cameras. In many applications like obstacle detection and localization systems, it is necessary to fuse data information supplied by each sensor to combine the advantages of every individual system in the final structure. The aim of this thesis is to develop a framework for a systematic fusion of XYZ point clouds from a Light Detection and Ranging (LiDAR) sensor and RGB images from a set of cameras. The proposed solution is generally applicable for all robotic solutions. However, our main motivation is to equip all these sensors onboard an unmanned aerial vehicle and capture data mid-flight during an inspection mission.

This thesis focuses on the methods for generating large coloured point clouds by fusion of the data from a LiDAR and a set of cameras. Due to the principle of modern LiDARs, which produce only spatial measurements, the colour information has to be supplied by other sensors. This is particularly important for 3D map colourization, but the fusion of the extra information can also improve navigation and localization of a mobile robot [1, 2]. The fusion also enables compensation of the individual limitations of each sensor type. For example, the colour information can be provided by cameras capturing the variable attenuation of light waves to produce colour information, but it is not possible to do spatial measurements with it.

For LiDAR and cameras data fusion purpose, the relative poses of the sensors have to be estimated. This thesis also aims to design and implement the process of calibration of the LiDAR and multi-camera sensors. With attention to finding the exact transformation, a calibration pattern is used. This thesis uses a planar checkerboard pattern to compute the calibration parameters. This approach can detect the inner corners of the checkerboard pattern from the camera with pixel accuracy. A planar model of the pattern can be estimated to the LiDAR point cloud corresponding to the plane with similar parameters as the real-world pattern board.

1.1 Motivation

In recent years, Unmanned Aerial Vehicles (UAVs) have been widely used in many fields, for example, the industrial [3, 4] and the heritage sector [5]. This technology can be used for inspection of the interior of warehouses [6, 7], factories [8] or power plants [9], and for inspection of artefacts within interiors of historical structures [10–12]. UAVs can carry various sensors, such as high-resolution cameras or laser scanners. The data from these sources are then used for creating static images, videos, or 3D maps.

This thesis is motivated by the existing industrial and cultural projects with the goal of autonomous inspection. The camera offers information about the colour of the surroundings, while LiDAR provides a spatial information. The fusion allows the construction of accurate and complete models of their environment. The fused information supplied by each sensor associating the advantages of every individual system brings benefits in many applications like navigation tasks. Depending on the application, coloured point clouds are also of great benefit for human or machine interpretation. Using the fusion of LiDAR and multiple cameras provides a more effective solution as the cameras may cover a wider field of view, which is typical for LiDARs opposite to using only a single camera, which is limited to its own field of view.

This thesis builds on the work of the MRS team of the Faculty of Electrical Engineering at Czech technical university. The main source of its data is the Dronument project [1], which is also the motivation for this thesis.

1.2 Related Work

To find the transformation between a camera and a 2D Laser Rangefinder, Zhang and Pless [13] introduced one of the first uses of a planar checkerboard. In their method, for several checkerboards poses, the checkerboard plane parameters are found relative to the camera. They then optimize the transformation by minimizing the euclidean distance error between the laser points and the checkerboard plane. A similar approach could be used to calibrate a 3D laser scanner and a camera. But most of these methods require some geometric constraints or some manual choice. Unnikrishnan and Martial [14] manually choose the 2D region of interest in the laser range image to find plane correspondences in both sensor frames. A two-stage optimization process is used, which involves estimating the rotation and translation independently and then jointly optimizing the two sets of parameters. In [15], the authors use a V-shaped calibration target formed by two triangular boards with a checkerboard on each triangle. As their target is non-planar, they are able to exploit the geometry of the setup to formulate a well-constrained cost function, minimizing point to plane distances. In [16] it is needed to define the bounds of the 3D experimental region relative to the LiDAR coordinate system and need to have a special stand, which does not hold the board with significant protruding elements close to the board boundaries or corners. This thesis tries to avoid most of these restrictions of [15] and [16]. It also avoids a manual choice of points from LiDAR or pixels from a camera like in [14].

[1] See: `https://dronument.cz/`

Fusion of LiDAR data with sensors of different properties is a problem often tackled in literature. One of the many examples is the fusion with monocular or stereo depth cameras. Monocular cameras estimate the depth of the scene by estimating the ego-motion of the camera first. In contrast, stereo cameras benefit from the static width of the baseline (the relative placement of the pair of two monocular cameras). This is addressed, for instance, in [17]. The dense stereo depth estimation is computationally complex due to matching corresponding points in the stereo images. A drawback of stereo based depth estimation is the limited range of depth sensing. Furthermore, dense depth estimation using stereo images is limited by a dynamic range of the image sensors, for example, the saturation of pixel values in bright areas [18].

Another option of the fusion is with a monocular camera, which is also the tackled problem in this thesis. In literature, this problem is challenged in various scenarios, such as the creation of 3D models of urban scenes for virtual reality [19] or for use in simultaneous localisation and mapping (SLAM) fusing either monocular [20, 21] or stereo [17] cameras. The colour data of each point is typically determined from a single observation, most often the closest frame or the first frame. In [22], the authors assume that the camera may not see an object measured with LiDAR. Therefore algorithm performs a visibility analysis first and uses only visible observations of a projected 3D point to compute the final colour. This thesis assumes that the distance between the camera and the LiDAR is low, therefore the pixel-based discrepancies of cameras are minimized. This assumption allows us to avoid the problem presented in [22]. Many systems [23, 24] capture high-resolution images instead of video. These systems typically use measured scene information to position each image within the LiDAR scans. In the proposed solution in this thesis, the data might be fused online during an inspection mission, since the static geometrical transformation among sensors is known from a calibration procedure.

1.3 Problem Definition

The key problem of creating a 3D model is an effective fusion of the data from multiple sensors. It is possible to use an RGB-depth camera like Kinect, which is a sensing device that captures both RGB image and depth image [25], however, the RGB-depth camera provides depth information up to a very limited range. In addition, the depth estimates obtained by the RGB-D camera are very noisy compared to a LiDAR. Therefore, the colour and texture information is usually collected by the camera, and the depth information is captured by the separate range sensor.

Using two different sensors brings other problems. The main challenge in fusing data from these two different sensor modalities is the requirement for precise calibration. It includes calibration of the camera's intrinsic parameters and the geometrical extrinsic parameters — precise transformation between the camera and the LiDAR. This calibration parameters are critical for correct fusion colours and points.

Obtaining the extrinsic parameters (the relative rotation and translation) between a camera and a LiDAR is a particularly discerning problem as the object features are obtained from different sensors with different modalities and noise patterns. Noise reduces the accuracy of the calibration. Furthermore, not all LiDARs and cameras have the same behaviour and measurement errors, which makes it difficult to generalize an approach. It is needed to address

these issues using features that are less susceptible to noise from sensor measurements and which are using a robust optimization strategy.

Apart from a precise intrinsic and extrinsic calibration, time synchronisation is also necessary to fuse camera data with a 3D points cloud. Both sensors are expected to be in motion, therefore data captured at different times may detect different objects.

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 letter	$\mathbf{R}$	a matrix or set
bold lowercase letter	$\mathbf{h}$	a column vector
upper index T	$\mathbf{R}^T$, $\mathbf{x}^T$	matrix and vector transpose
hat index T	$\hat{m}$, $\hat{P}$	a point in a homogeneous coordinate system

Table 1.1: Overview of the mathematical notation

1.5 Table of Symbols

Chapter	Symbol	Description
Mathematical background (Chapter 2)	f	Focal length
	m	Point on the image plane
	P	Point in space
	$\mathbf{K}$	Camera calibration matrix
	$\mathbf{I}$	Identity matrix
	$\mathbf{R}$	Rotation matrix
	$\mathbf{t}$	Translation vector
	k_1, k_2, k_2	Radial distortion parameters
	p_1, p_2	Tangential distortion parameters
Calibration (Chapter 4)	f	Focal length
	m	Point on the image plane
	P	Point in space
	$\mathbf{K}$	Camera calibration matrix
	$\mathbf{r}$	Rotation vector
	$\mathbf{t}$	Translation vector

Table 1.2: Summary of symbols utilized

Chapter 2: Mathematical background

Contents

2.1 Pinhole Camera Model

The pinhole camera model represents a simple camera with a single small aperture without a lens. The camera uses the central projection of three-dimensional points in space onto the two-dimensional image plane. It creates a centre-rotated image on the image plane. To simplify the mathematical description, the image plane between the focus and the scene is used. The image plane is located at the focal length f. This virtual image plane is parallel to the image plane behind the focal point and it has the same distance from the focal point as the image plane. The advantage of using the virtual image plane in front of the focus is that the projected image is not rotated (see Figure 2.1). In the following parts of the thesis, the term image plane is always used for the virtual image plane.

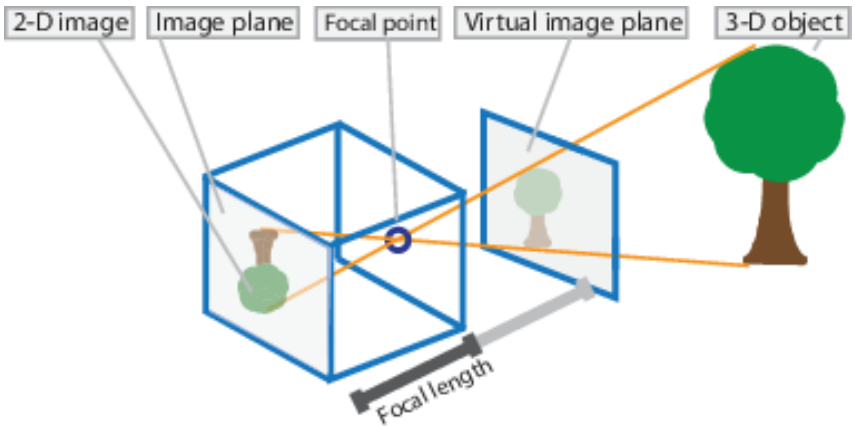


Figure 2.1: Central projection a pinhole camera model [26].

2.2 Camera Parameters

Let the centre of projection $\mathcal{F}_c$ be the origin of a Euclidean coordinate system and let the plane $\mathcal{Z}$, with equation $z = f$, be called the image plane, where f is the focal length and the plane $\mathcal{Z}$ is the plane in the direction of the optical axis. The point, where an optical axis intersects the image plane is called the principal point and it has coordinates (c_x, c_y). A point in space with coordinates $P = (X, Y, Z)$ is mapped onto the point $m = (u, v)$ on the image plane where a line joining the point P and the centre of projection $\mathcal{F}_c$ intersects the image plane. This is depicted on Figure 2.2. Using similar triangles, it can be shown that the point $(X, Y, Z)^T$ is mapped onto the point $\left(f\frac{X}{Z},\ f\frac{Y}{Z},\ f\right)^T$, which lies on the image plane and could by called image point [27].

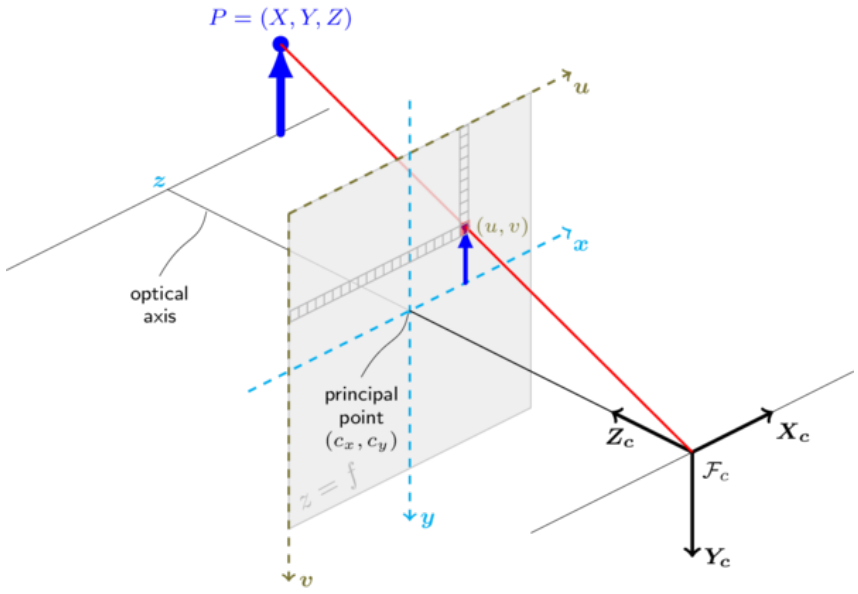


Figure 2.2: Projection of a point in space onto a point on the image plane [28].

If the points in space and on the image plane are rewritten in a matrix form, then the central projection is very simply expressed as a linear mapping between their homogeneous coordinates. It can be written in terms of matrix multiplication as

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} fX \\ fY \\ Z \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}. \tag{2.1}$$

In practice, it may happen that the origin of coordinates of the image plane is not at the principal point. As a result, the general formula for the mapping is

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} fX + Zc_x \\ fY + Zc_y \\ Z \end{bmatrix} = \begin{bmatrix} f & 0 & c_x & 0 \\ 0 & f & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}. \tag{2.2}$$

Let the matrix on the right-hand side of the equation Equation 2.2 be labeled as

$$\mathbf{K} = \begin{bmatrix} f & 0 & c_x & 0 \\ 0 & f & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{2.3}$$

Then Equation 2.2 can be rewritten in a concise form

$$\hat{m} = \mathbf{K} \ [\mathbf{I}|\mathbf{0}] \, \hat{P}, \tag{2.4}$$

where $\hat{m}$ is an image point vector and P is a world point vector in the homogeneous coordinate system. The matrix $\mathbf{K}$ is also called the camera calibration matrix.

At this point, the coordinates in the image coordinate system are known, however, the points in space are described in terms of a different Euclidean coordinate system, known as the world coordinate system. The two coordinate systems can be transformed between each other using a geometric transformation. It is also better not to make the camera centre explicit, but to represent the world to image transformation as $P_{cam} = \mathbf{R}P + \mathbf{t}$, where $\mathbf{R}$ is a 3x3 rotation matrix and $\mathbf{t}$ is a 3-dimensional translation vector. This equation is commonly described as

$$\mathbf{x'} = \left[\begin{array}{c|c} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{array}\right] \mathbf{x}, \tag{2.5}$$

where $\mathbf{x}$ is the homogeneous representation of the point P and $\mathbf{x'}$ represents the same point in the camera coordinate system.

This equation may be then written as

$$\begin{bmatrix} X_{cam} \\ Y_{cam} \\ Z_{cam} \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}. \tag{2.6}$$

Using Equation 2.5, the camera equation Equation 2.4 can be simply written as

$$\hat{m} = \mathbf{K} \left[\begin{array}{c|c} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{array}\right] \hat{P}, \tag{2.7}$$

where $\mathbf{R}$ is a rotation matrix and $\mathbf{t}$ represents a translation vector.

2.3 Camera Lens Distortion

The camera matrix does not account for the lens distortion because the pinhole camera model does not have a lens. To accurately represent a real camera, the camera model needs to include the lens distortion. There are two main distortion types: radial and tangential distortion.

2.3.1 Radial distortion

With real lenses, rays farther from the centre of the lens are bent more than those closer in (see Figure 2.3). This phenomenon is called radial distortion. This bulging effect is the source of the "barrel" or "pincushion" distortion, which are shown in Figure 2.4.

For radial distortions, the distortion is zero at the optical centre of the image and increases as it moves toward the edge. This distortion is so small that it can be characterized by the first few terms of a Taylor series expansion around a distortion radius r = 0. OpenCV library uses three such terms. The first term is conventionally called k_1 and the second one k_2 for most camera lenses. For highly distorted lenses there can be added a third radial distortion term k_3 [29].

To consider these distortions in our camera model we modify the pinhole camera model as follows:

$$\begin{aligned} r^2 &= u^2 + v^2, \\ u' &= u\ (1 + k_1 r^2 + k_2 r^4 + k_3 r^6), \\ v' &= v\ (1 + k_1 r^2 + k_2 r^4 + k_3 r^6), \end{aligned} \tag{2.8}$$

where u and v represent the original location of the image of the distorted point, u' and v' are the new location coordinates as a result of the distortion.

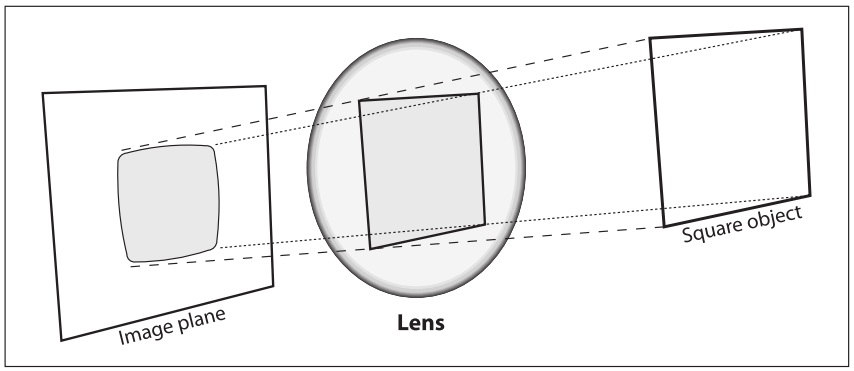


Figure 2.3: Radial distortion [29].

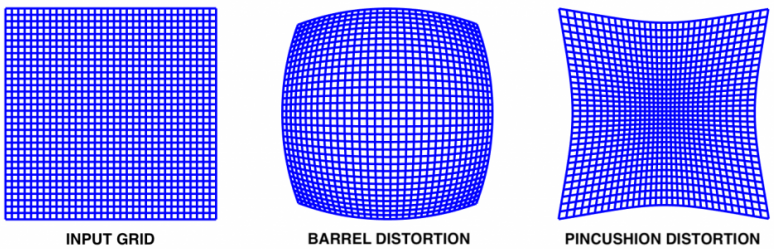


Figure 2.4: Example of the effect of barrel distortion and pincushion distortion on a square grid [30].

2.3.2 Tangential distortion

Tangential distortion is caused by the manufacturing imperfections of the camera construction. Lens and CCD sensor are misaligned from their mutual parallel position. This misalignment changes arrays from their ideal, perpendicular orientation to the optical axis and in effect moves the point where an array meets the sensor (see Figure 2.5). The points with the same distortion form an ellipse.

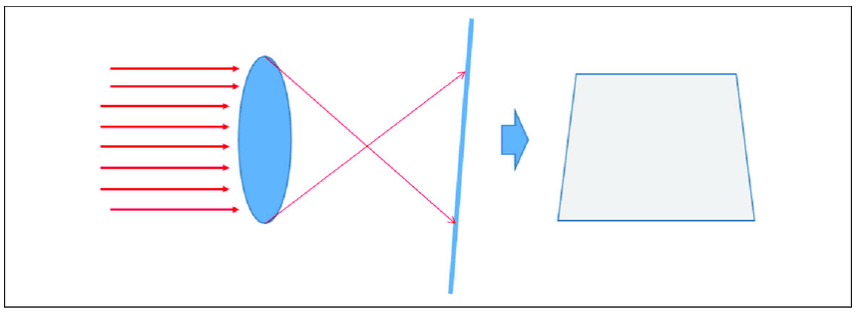


Figure 2.5: Tangential distortion [31].

OpenCV library uses two additional parameters p_1 and p_2 to characterize it. This results in

$$\begin{aligned} r^2 &= u^2 + v^2, \\ u' &= u + \ (2p_1uv + p_2(r^2 + 2u^2)), \\ v' &= v + \ (p_1(r^2 + 2v^2) + 2p_2uv), \end{aligned} \tag{2.9}$$

where u and v represent the original location on the image of the distorted point, and u' and v' represent the new location as a result of the distortion [29].

Chapter 3: Preliminaries

Contents

The aim of this thesis is to design a completely autonomous system for fusion of the data from LiDAR and camera. This system consists of both software and hardware. The implementation is performed with the Robot Operating System (ROS). This tool is hugely popular for its evolving functionality and there is also support for processing of the data from LiDAR and cameras attached to an unmanned aerial vehicle.

3.1 Robot Operating System

ROS (Robot Operating System) provides libraries and tools to help software developers create robot applications. It provides hardware abstraction, device drivers, libraries, visualizers, message-passing, package management, and more[1].

ROS Nodes are processes that perform computations in a modular way. A UAV control system usually includes many nodes. For example, one node handles sensory data, one node performs localization, and another node controls sensors. ROS Master is the principal part of a ROS computational process because it provides naming and registration for ROS nodes. Without the ROS Master, the nodes would not be able to find each other. Figure 3.1 illustrates the core communication between ROS master and nodes.

The concept of ROS allows every node to transfer messages to other nodes. A message is a data structure consisting of typed fields. Messages are transported via so-called topics or services. There are two types of topics. The first type is a subscriber, which listens to the topic with a specific name and acquires data. The second type is a publisher, which publishes this data. There can be multiple publishers and subscribers for a single topic.

ROS source code is organized in packages. Packages are atomic items in ROS, which means that it is the most granular part that can be built. Packages use runtime processes — nodes, ROS-dependent libraries, configuration files, or anything else that helps with organization.

[1]Source: `http://wiki.ros.org/`

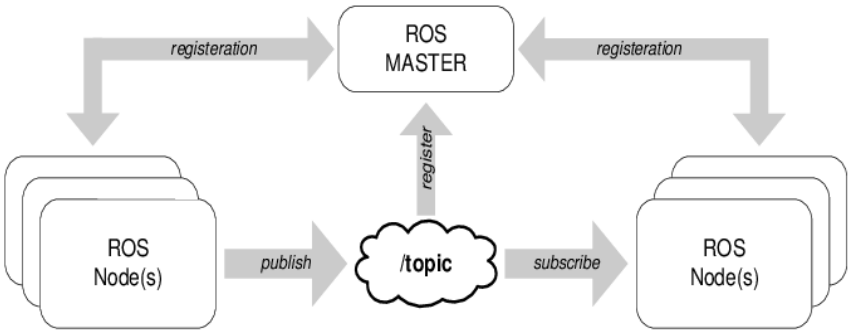


Figure 3.1: ROS environment communication between master and nodes [32].

3.2 Hardware

On the hardware side, this thesis works with two basic sensors. The quality of data from both sensors fundamentally affects the resulting calibration and the quality of the fused map. To better understand calibration and fusion problems, it is good to know the functionality principle of both sensors.

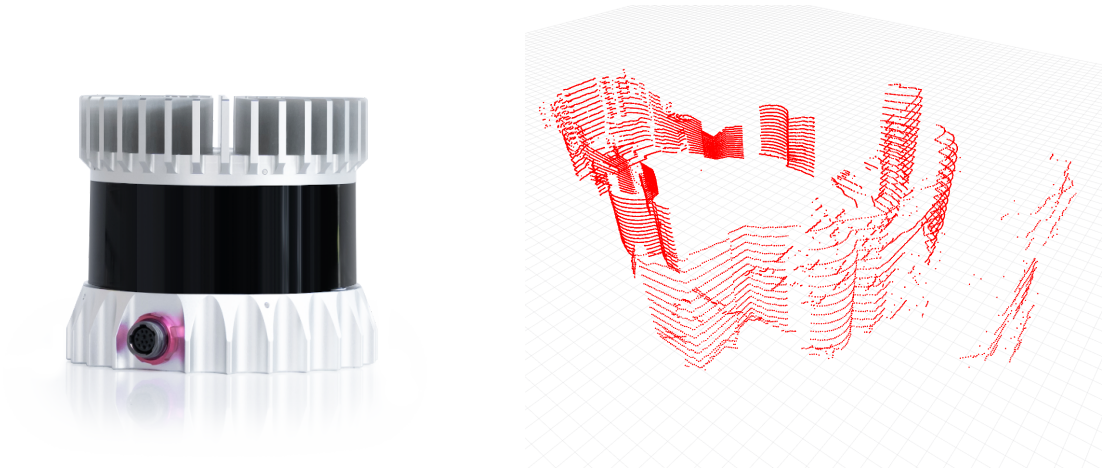
3.2.1 LiDAR

The Light Detection and Ranging (LiDAR) system is an onboard system for the localization of UAVs and the mapped environment surrounding the UAV. The system consists of a laser transmitter and a laser receiver. The measurement of distances is then based on the measurement of the time between sending the laser beam and receiving its reflection. The differences in the laser return time give the distance between the UAV and the object. The distance combined with a known angle can be used to make a 3D point that represents the environment. The scanning speed of the LiDAR also determines the maximum speed of movement of the drone, because the movement of the UAV reduces the accuracy of measurements. Modern systems cope with LiDAR movements during a continuous laser measurement using integrated IMU and motion de-compensation.

For this thesis, a rotation LiDAR from the MRS team laboratory is used. It is Ouster OS0-16 or OS1-128, with rotation rate of both 10 and 20 Hz. This model of the LiDAR is shown in Figure 3.2.

3.2.2 RGB camera

A camera is an electromechanical device using the process of image formation, namely the formation of the two-dimensional representation of the three-dimensional world. This thesis assumes that cameras can be modelled as a pinhole camera, although a real camera has slight differences compared to the pinhole model. For example, a real camera has a standard CMOS or CCD sensor with an RGB pixel array instead of an image plane of a pinhole model. The camera sensor is divided into individual pixels, each detecting red, green, and blue wavelengths. The resolution of the camera sensor, i.e. the number of pixels, also affects



(a) OUSTER OS0-128 LiDAR sensor [33]. (b) Data from LiDAR

Figure 3.2: Demonstration of LiDAR and produced data

the quality of the resulting fusion, especially at larger distances. There is also a lens on a real camera that adjusts the flight path of the rays and adds other distortion to the projection.

For this thesis USB 2.0 board-level camera - mvBlueFOX-MLC with Resolution 640x480 pixels is used and it is shown in Figure 3.3.

Figure 3.3: mvBlueFOX camera [34].

3.3 UAV platform

In real experiments and simulations, a drone developed by the MRS team of the Faculty of Electrical Engineering in CTU is used. This thesis uses the Tarot T650 model of a platform (see Figure 3.4). The platform is capable of completely autonomous flight and operation. Various sensory packages are used for different tasks, allowing autonomous operation in difficult environments with obstacles. The package with a camera, LiDAR sensor and a large data storage disk is used in this thesis. The platform is equipped with a PixHawk flight controller running the PX4 stack, which acts as a low-level controller. It also has a powerful

onboard computer, usually an Intel NUC. This computer runs all the control and estimation algorithms and takes in data from multiple sensors [35].

(a) Tarot T650 model in simulation. (b) Tarot T650 model in real-world.

Figure 3.4: Photo of the Tarot T650 UAV model used in this thesis.

Chapter 4: Calibration

Contents

This part of the thesis deals with extrinsic calibration, which aims to find the geometric transformation matrix between a camera and a LiDAR. The transformation matrix converts the point coordinates between the coordinate systems of the two sensors. Without precise calibration, it is not possible to fuse points from LiDAR and pixels from the camera. As the projection errors arise with the distance of the objects to the sensors, even slight calibration errors may produce infeasible results, which results in high requirements for precise retrieval of calibration. The Figure 4.1 describes the transformation between LiDAR and camera coordinate system.

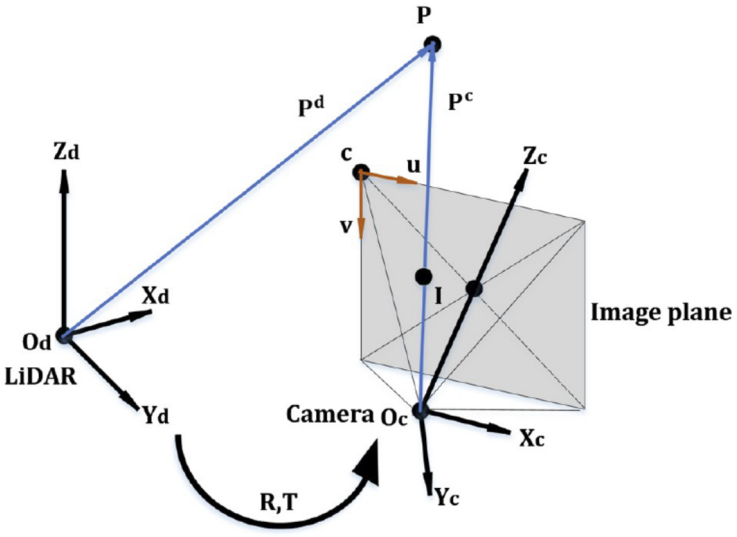


Figure 4.1: Projection model of the camera in a LiDAR-camera system. P represents the 3D point, O_d and O_c are LiDAR and camera coordinate systems [36].

This thesis uses a checkerboard pattern as a reference to obtain points of interest in the image and point cloud. It uses the same checkerboard pattern as for the intrinsic camera calibration (see Figure 4.3). The calibration method could be divided into three parts: extraction

of the checkerboard corners from the camera, extraction of the desk corners from the LiDAR data, and finding the transformation between the points and the pixels.

These parameters are required for the extrinsic calibration.

- The number of internal corners in each row and column on the calibration checkerboard pattern (see Figure 4.2).
- Side length of the squares of the checkerboard pattern (see Figure 4.2).
- Length between each edge of the desk and the checkerboard pattern (see Figure 4.2).
- Camera intrinsic parameters including the focal length, principal point, distortion coefficients and real size of pixels on the camera sensor (i.e., the sensor size).

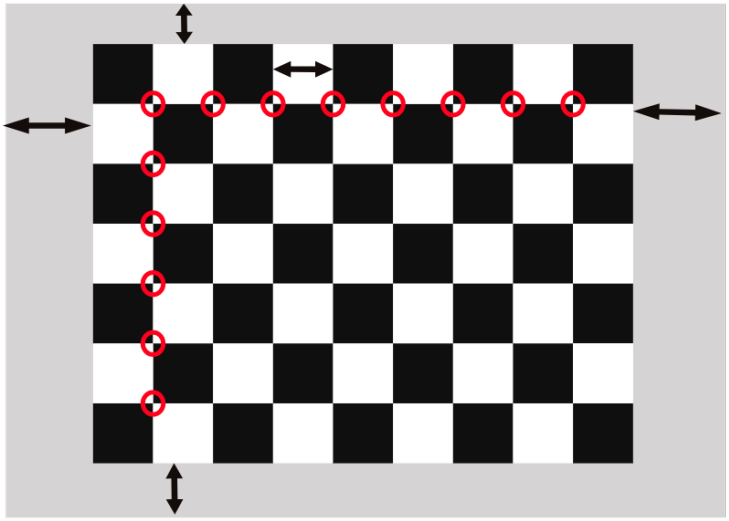


Figure 4.2: Parameters of checkerboard pattern desk – red circles show internal corners in a row and a column, and black arrows show the side length of the squares and length between each edge and checkerboard pattern.

4.1 Checkerboard extraction

Checkerboard detection consists of two steps. The first is to detect the checkerboard as a whole and distinguish it from other content on the image. This determines whether the image can be used in the calibration or not, particularly in difficult conditions. The second is finding the accurate corner locations. This determines the accuracy and precision of the calibration. Checkerboard corners are well suited because they have strong gradients in all the directions. The Harris operator is the most well-known corner detector often used in camera calibration [37].

In this thesis, the checkerboard pattern in the image from the camera is detected using the OpenCV function *cv::findChessboardCorners*[1]. OpenCV library uses adaptive threshold-

[1]See: `https://docs.opencv.org/3.2.0/d9/d0c/group__calib3d.html`

ing and erosion to binarize the image and separate the checkerboard squares into quadrilaterals by contour following. Finally, the checkerboard is detected as a 2D grid of connected quadrilaterals [37].

After detection of internal checkerboard corners, it is required to calculate the real corners of the board. Thus, the known real size of the squares is used together with the lengths between each edge from the pattern board. Using the similarity of the triangles, the distance of the real corners from the corners detected on the checkerboard in both dimensions of the image is calculated. The Figure 4.3 shows a checkerboard with detected internal and external corners from real data.

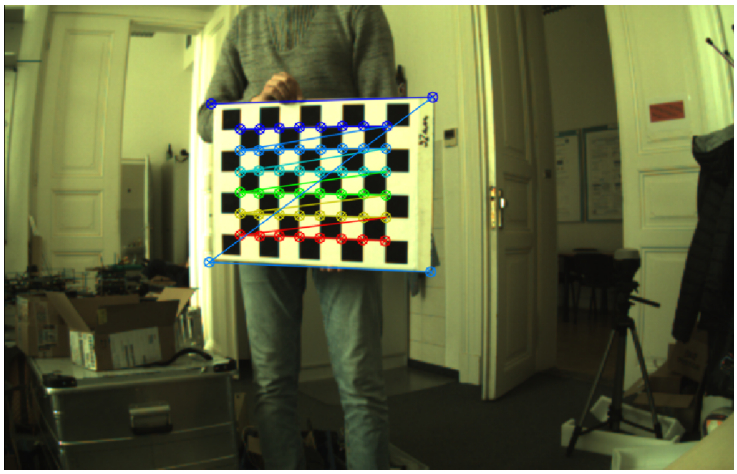


Figure 4.3: Checkerboard with detected internal and real corners.

From this data, the distance of the board from the camera can also be calculated. This distance is then used to detect the board from LiDAR data. The following equation is used to calculate the distance to the object $d(mm)$

$$d = \frac{f o_{hr} i_h}{o_h s_h}, \tag{4.1}$$

where f is the camera focal length in millimetres, o_{hr} is the real object height in millimetres, i_h is the image height in millimetres, o_{hp} is the object height in pixels and s_h is the sensor height in millimetres. Equation 4.1 can be rewritten as

$$d = \frac{f d_{hr}}{d_{hp} p_s}, \tag{4.2}$$

where d_{hr} is the real height of desk in millimetres, d_{hp} is the desk height in pixels and p_s is the pixel size in millimetres.

4.2 Desk extraction

The estimated distance of the board, which is calculated from the cameras data, allows the separation of the experimental region (including the board) from the environment point cloud. Due to the inaccuracy in distance calculation, a small deviation in both directions needs

to be added. The obtained experimental region consists of all objects at the same distances around the sensor, including the board. However, this step will significantly speed up and simplify the search for the board in the next step because it radically reduces the region of interest. It also removes all large planes, like walls, ceiling or ground, or at least a significant part of them.

Further, the experimental region is segmented into planes. In this thesis, the RANSAC algorithm is used to get a robust estimate of the board plane because it is suited for applications where interpretation is based on the data provided by error-prone feature detectors [38]. The algorithm selects the smallest number of data samples required to define a model uniquely. In the case of a plane, it selects three points. The RANSAC extracts shapes from the point cloud and constructs corresponding primitive shapes based on the three randomly selected points. The resulting candidate shapes are tested against all the points in the data to determine which points are well approximated by the primitive. After a given number of iterations, the shape which is largest and approximates the most points is extracted [39]. The Figure 4.4 shows RANSAC algorithm to estimate a line in 2D. This shape is cleansed from outlier points using a filter. The filter uses a simple principle; it filters points in a cloud based on the number of neighbours they have. It retrieves the number of neighbours within a certain radius for each point. The point is considered as an outlier if it has too few neighbours and the filer removes it.

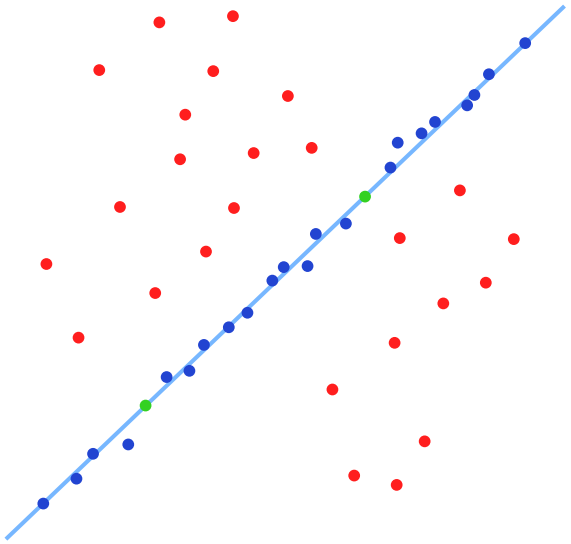


Figure 4.4: RANSAC algorithm to estimate a line. Green points are randomly selected points, blue ones are inliers, and reds are outliers [40].

The next step is the comparison of the plane to the checkerboard board. Parameters like height, width or number of points are tested. If parameters are almost the same, the plane is identified as the board. If they are not, points from the plane are extracted and the algorithm continues with the remaining data until the right plane is not found, the point cloud is too small, or the number of iterations is exceeded.

However, the obtained point cloud of the board may not include the whole board. The

problem could be with edges and corners. This problem may be caused by noise from LiDAR but also by the use of a filter of outlier points. Therefore a new point cloud of the board is created such that it physically corresponds to the real dimensions of the board. This model is aligned with the board from the previous part using an iterative closest point (ICP) algorithm. ICP starts with two meshes and an initial guess for their relative rigid-body transform. ICP then iteratively refines the transform by repeatedly generating corresponding points on the meshes and minimizing an error metric [41]. The Figure 4.5 shows idea behind the iterative closest point algorithm. After the end of the ICP algorithm, the corners from the aligned board are saved for the last part of the calibration proposed in Section 4.3.

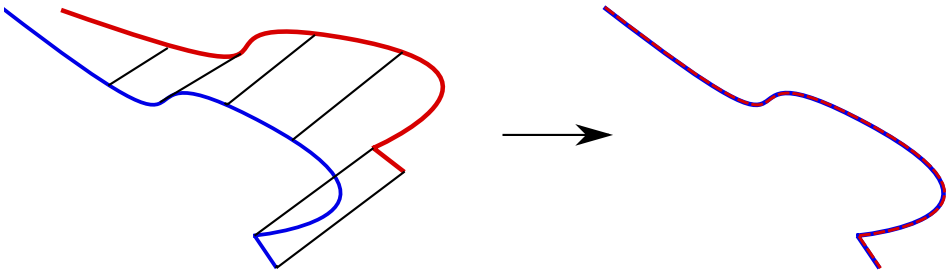


Figure 4.5: Idea of iterative closest point algorithm [42].

4.3 Search transformation

Given the 2D and 3D coordinates of the checkerboard pattern corners from the camera and the LiDAR, the extrinsic calibration can be initiated. The optimization problem can be defined as the Perspective-n-Point (PnP) problem, which determines the position and orientation given a set of n pairs between 3D points and their corresponding 2D projections in the image. Many methods can be applied to solve the 3-to-n points PnP optimization, ranging from iterative and non-iterative techniques, to methods with and without apriori information about the camera parameters. The further described optimization process is reliant on the number of correspondences, hence supplying more correspondence pairs improves the robustness of the camera-to-lidar pose estimation.

The solution proposed in this thesis uses the Efficient PnP (EPnP) algorithm, introduced by F.Moreno et al.s in 2008 [43]. The EPnP solves the camera pose by expressing the coordinates as a weighted sum of 4 non-coplanar virtual control points (see Figure 4.6). The coordinates of the control points become the unknowns of the problem. It is from these control points that the final pose of the camera is solved for. The EPnP method assumes that the camera parameters are known.

The EPnP problem is formulated as

$$\mathbf{p}_i^w = \sum_{j=1}^{4} \alpha_{ij} \mathbf{c}_j^w, \tag{4.3}$$

where $\mathbf{p}_i^w$ is a reference point in the world coordinates system, α_{ij} are the homogeneous barycentric coordinates, and $\mathbf{c}_j^w$ is a control point in the world coordinates system, which is

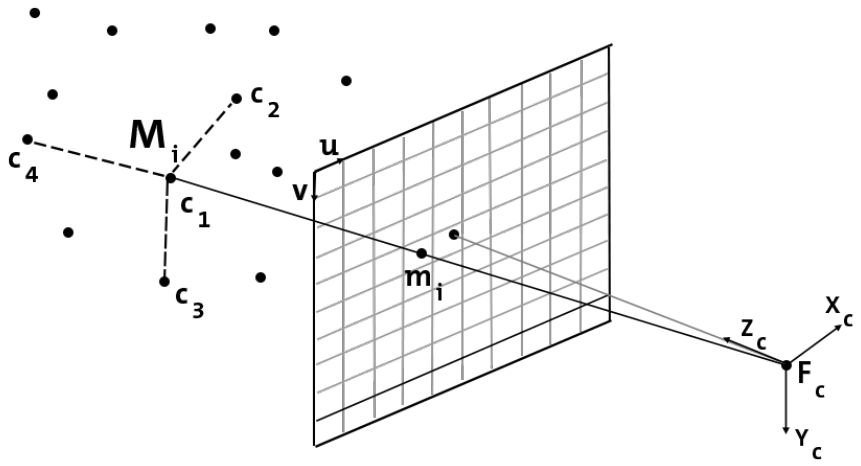


Figure 4.6: Given a set of 3D points M_i and their 2D projections m_i onto the image, the points $c_{1..4}$ form the base, representing all the set by a linear combination.

the unknown of the problem. The same relation holds in the camera coordinate system

$$\mathbf{p}_i^c = \sum_{j=1}^{4} \alpha_{ij} \mathbf{c}_j^c, \tag{4.4}$$

where $\mathbf{p}_i^c$ is a reference point and $\mathbf{c}_j^c = \begin{bmatrix} X_j^c & Y_j^c & Z_j^c \end{bmatrix}^T$ is a control point in the camera coordinates system.

The derivation of the matrix M, in whose kernel the solution must lie given that the 2D projections of the reference points are known, is as follows

$$\forall i, w_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \mathbf{K}\mathbf{p}_i^c = \mathbf{K} \sum_{j=1}^{4} \alpha_{ij} \mathbf{c}_j^c, \tag{4.5}$$

where the w_i are scalar projective parameters, u_i and v_i are the 2D coordinates of a projected p_i point and $\mathbf{K}$ is the camera parameters matrix.

Using the Equation 2.3 and rearranging it, the following two linear equations for each reference point could be written as

$$\sum_{j=1}^{4} \alpha_{ij} f x_j^c + \alpha_{ij}(c_x - u_i) z_j^c = 0, \tag{4.6}$$

$$\sum_{j=1}^{4} \alpha_{ij} f y_j^c + \alpha_{ij}(c_y - v_i) z_j^c = 0. \tag{4.7}$$

By concatenating them for all n reference points, we generate a linear system of the form

$$\mathbf{M}\mathbf{x} = \mathbf{0}, \tag{4.8}$$

where $\mathbf{M}$ is a $2n \times 12$ matrix, generated by arranging the known coefficients of Equation 4.6 and Equation 4.7, and $\mathbf{x} = \begin{bmatrix}\mathbf{c}_1^c & \mathbf{c}_2^c & \mathbf{c}_3^c & \mathbf{c}_4^c\end{bmatrix}^T$ is a vector made of the unknowns.

The solution to this system lies on the null space, or kernel, of M, expressed as

$$\mathbf{x} = \sum_{j=1}^{N} \beta_i \mathbf{v}_i, \tag{4.9}$$

where the set $\mathbf{v}_i$ are the columns of the right singular vectors of $\mathbf{M}$ corresponding to the $N \in \{1, 2, 3, 4\}$ null singular values of $\mathbf{M}$. After calculating the initial coefficients beta, the Gauss-Newton algorithm is used to refine them. The rotation matrix $\mathbf{R}$ and translation vector $\mathbf{t}$ minimizing the reprojection error of the world reference points $\mathbf{p}_i^w$ and their corresponding image points $\mathbf{p}_i^c$, are then calculated using methods presented in [43, 44]. In this thesis the OpenCV function *cv::solvePnP*[2] is used to estimate the extrinsic transformation in the form of orientation $\mathbf{R}$ and translation $\mathbf{t}$.

The optimal convergence of the EPnP method is not guaranteed, therefore the reprojection error of the optimization error is manually computed and used to filter out incorrect results. This improves the robustness of calibration, particularly during real-time calibration with incrementing number of frame correspondences over time.

The reprojection error is given by the following equation

$$\epsilon = \frac{\sum_{i=1}^{n}(u_{oi} - u_{pi})^2 + (v_{oi} - v_{pi})^2}{n}, \tag{4.10}$$

where u_o, v_o are the original coordinates on the image, and u_p, v_p are the projected coordinates using the estimated transformation and rotation vector from the EPnP.

If the reprojection error is smaller than from the previous iteration, a new rotation and a translation vector are stored. If the optimization converged to similar values, the average of old and new transformation is used and stored, making the calibration more robust to outliers. The output of the calibration, which continuously combines pairs of LiDAR and camera data in real-time until terminated, returns the transformation with the smallest reprojection error.

[2] See: `https://docs.opencv.org/3.2.0/d9/d0c/group__calib3d.html`

Chapter 5: Fusion method

Contents

This part of the thesis solves the problem of the fusion of data from a colour camera and a LiDAR. The result should be a coloured point cloud based on the data from the sensors. The fundamental process to achieve colourised point clouds is to project the 3D points onto 2D points in images, such that the appropriate colour is assigned to each 3D point. Because sensors could catch the same points several times it is necessary to add filters. Also, filters help to make the process more robust with the respect to potential errors. The fusion method in this thesis is separated into three parts: data association and point cloud transformation, a projection of points from the LiDAR onto the camera, and post-processed filtration.

5.1 Data association

When trying to fuse data from different sensors, it is necessary to know how to convert one sensor frame from its coordinate system to the coordinate system of the other sensor in order to map them on top of each other. Because even small calibration errors may produce infeasible results, this assumes that exact extrinsic calibration is known, as described in Chapter 4. This calibration is used as one of the input parameters of the program and without it the program cannot work properly.

Data synchronization is especially important when working with data from multiple sources. It means that the data coming from different sensors at a specific moment should have the same timestamp. When the data is processed, the point cloud from LiDAR and RGB image from the camera with equal timestamps are fused, providing a correct representation of that moment in time. In the opposite situation, when the timestamps aren't the same, the data from LiDAR and camera are discarded. It is because the UAV could move and the data that are taken at different times could represent a different point in space.

The time synchronization between sensors is achieved by using the TimeSynchronizer filter from the ROS message filters library. *The Synchronizer filter synchronizes incoming channels by the timestamps contained in their headers, and outputs them in the form of a single callback that takes the same number of channels. The Synchronizer filter is templated*

on a policy that determines how to synchronize the channels[1]. ApproximateTime policy was implemented to use an adaptive algorithm[2] to match messages based on their timestamps.

Using the rotation matrix and the translation vector, a 3D point is transformed from the LiDAR coordinate system into the coordinate system of the camera. Then the 3D point can be projected by the camera projection matrix to the appropriate pixel in the image plane in the camera. RGB color information is selected from the camera data and it is assigned to the original 3D point in the LiDAR coordinate system. It is impossible to find the correct colour value for invalid 3D points, so they should be filtered out before colouring the 3D point cloud. Obviously, points behind the camera are invalid. In addition, it is useful to reduce the number of points, which are close to each other. This results in a more continuous and uniform distribution of the points in space. This is accomplished by doing a convolution between a kernel and an image.

For the reduction of the number of points, a voxel grid filter was used. This filter falls into the class of down-sampling filters as it reduces the number of points in a cloud. It computes the centroid, a single point which then represents the given group of points – a voxel or a cluster. That means that the set of points which lie within the bounds of a voxel are assigned to that voxel and will be combined into one output point. The formula to compute the centroid is

$$\bar{p}\left(\bar{x}\bar{y}\bar{z}\right) = \frac{1}{n}\sum_{i=1}^{n}\left(x_i, y_i, z_i\right), \tag{5.1}$$

where n is the number of points inside the voxel, $\bar{x}$,$\bar{y}$,$\bar{z}$ of $\bar{p}$ are coordinates of the centroid, and x_i, y_i, z_i are the coordinates of each point p_i within the voxel [45].

Clearly, this option is more accurate since it considers the point distribution inside the voxel and then takes the geometrical centre of the voxel. Figure 5.1 is a comparison of voxel grid centroid and geometric centre in two dimensions.

5.2 3D-to-2D projection

Using the rotation matrix, the translation vector and the camera projection matrix a 3D point from LiDAR is projected onto a pixel in the image plane in the camera. The Equation 2.7 may be written as

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & c_x & 0 \\ 0 & f & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \tag{5.2}$$

where u and v represent the undistorted image point as projected by an ideal pinhole camera.

However, as it was described above, the camera has distortion, so it is necessary to use equal distortion on the image point. Combining the formula for tangential and radial

[1] Source: `http://wiki.ros.org/message_filters`

[2] Adaptive algorithm, see `http://wiki.ros.org/message_filters/ApproximateTime`

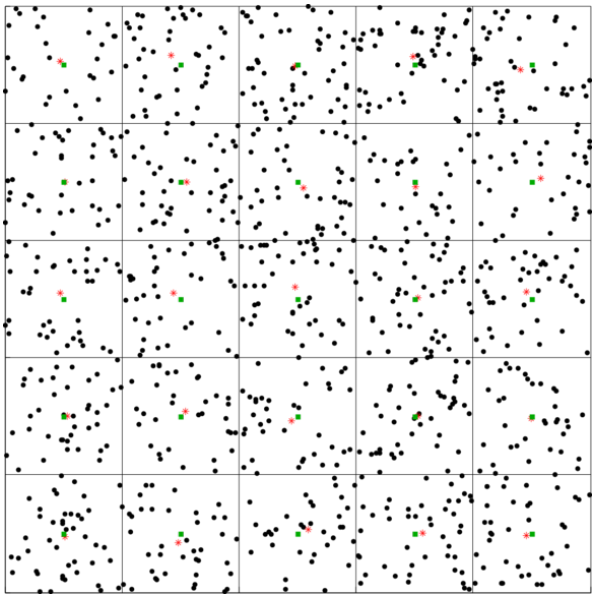


Figure 5.1: A 2D Voxel Grid with red stars representing the voxel centroids and green stars the geometric centres [46]

distortion results in the equation

$$\begin{bmatrix} u' \\ v' \end{bmatrix} = (1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \begin{bmatrix} u \\ v \end{bmatrix} \begin{bmatrix} 2p_1 uv + p_2(r^2 + 2u^2) \\ 2p_1 \left(r^2 + 2v^2\right) + 2p_2 uv \end{bmatrix}, \tag{5.3}$$

where u and v are the coordinates from ideal pinhole camera, $r^2 = u + v^2$, k_1, k_2 and k_3 are the terms for the radial distortion and p_1 and p_2 are the terms for the tangential distortion.

The next step is to remove the points which are out of bounds of the camera sensor. Given that the image plane is infinite and the area of the camera sensor is limited, the projection matrix can generate a 3D point outside of the camera sensor area. Out-of-view points need to be removed because no colour can be detected for these points. Therefore, every point that is projected out of sensor size is removed.

Finally, to get RGB colour information, a kernel is used around the projected pixel. Using a convolution between a kernel matrix and an image also allows the use of surrounding pixels and makes the algorithm more robust in case of inaccurate projection. This also helps with the increase in the uncertainty of the pixel location with increasing camera distance. The obtained colour information is assigned to the correct 3D point from the LiDAR data. This process is repeated for every point obtained from LiDAR.

5.3 Post processing

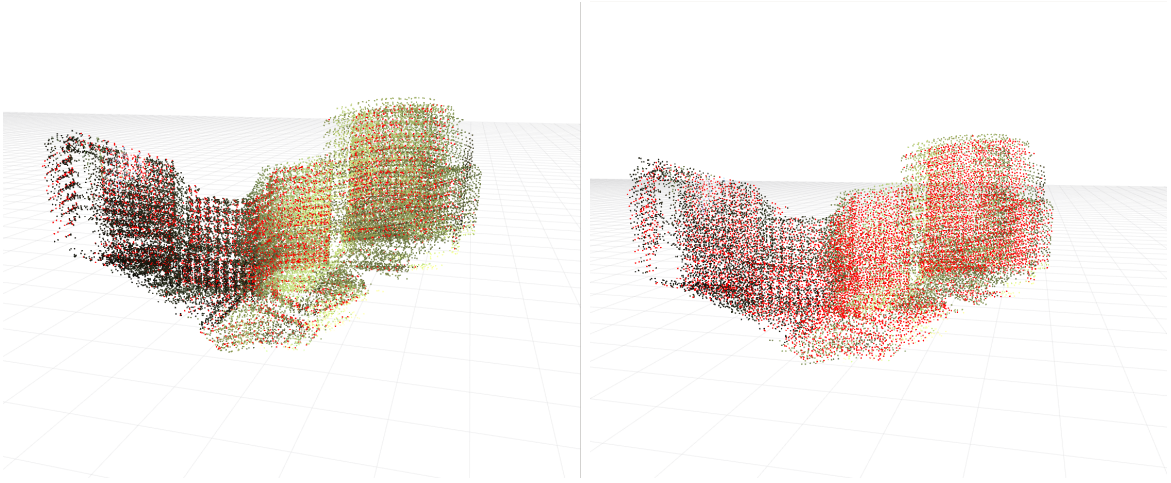
After getting the colour of the 3D point from the LiDAR, another filtering is needed. This is caused by the fact that the unmanned aerial vehicle moves and leans in all directions, so it is likely that LiDAR and camera catch the same 3D point several times. Because the lighting conditions of the same point from various angles may be different, using an average of more colour information leads to a better result.

This filter is also a member of the class of down-sampling filters. The first step of the filtering is to compute the Euclidean distance between each pair of points. The formula to compute a Euclidean distance is

$$d\left(P_1, P_2\right) = \sqrt{\left(x_1 - x_2\right)^2 + \left(y_1 - y_2\right)^2 + \left(z_1 - z_2\right)^2}, \tag{5.4}$$

where x_1, y_1, z_1 are the coordinates of the point P_1 and x_2, y_2, z_2 the coordinates of the point P_2.

If the computed distance is less than the specified value, it merges the two points and computes their average colour. Using this filter in conjunction with the voxel grid filter from Section 5.1 ensures that the new points from each iteration are merged with points from the previous iterations. This results in maintaining an even distribution of points in space. Figure 5.2 is a comparison of two different values for distance in the filter.



(a) Filter distance d = 0.2 (b) Filter distance d = 0.5

Figure 5.2: Comparison of two different values for distance in the second filter. Red points are affected by the distance-based filtration.

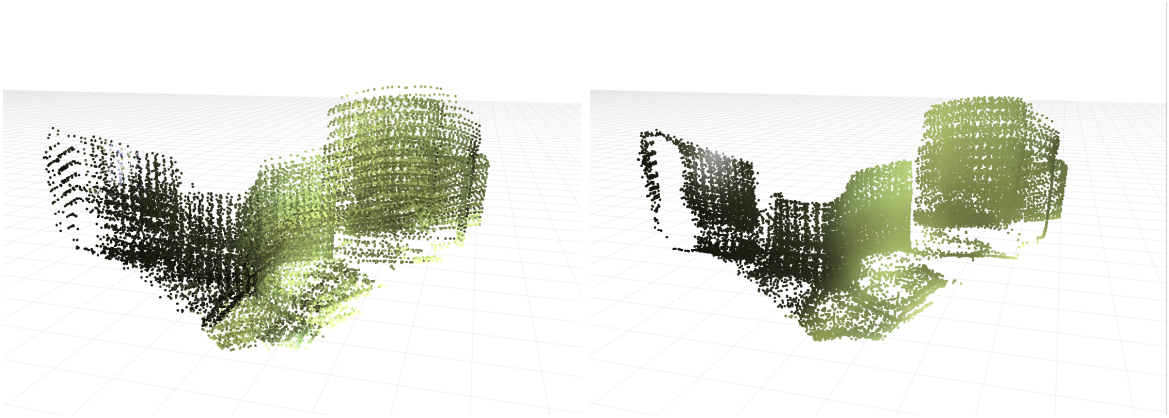
The next step is the transformation of the resulting point cloud into a stable frame. Until now, we considered all the points relative to the camera or the LiDAR frame. However, the LiDAR and the camera move in space, so the measured points move with them. Therefore, the points must be transformed for the last time into a frame that has an absolute and unchanging position. Without this, it would not be possible to create a 3D map. This stable frame might be e.g. the localization frame of the robot.

The final step, before publishing the resulting point cloud, is smoothing by convolution. Convolution filters are often used for adaptive smoothing or feature extraction. It also helps to remove the noise created by moving the sensors. This is accomplished by doing a convolution between a kernel and a point cloud. A kernel is a 3D matrix, and the convolution process can be expressed like

$$g(x, y, z) = \mathbf{w} f(x, y, z) = \sum_{\mathrm{d}x=-a}^{a} \sum_{\mathrm{d}y=-b}^{b} \sum_{\mathrm{d}z=-c}^{c} \mathbf{w}(\mathrm{d}x, \mathrm{d}y, \mathrm{d}z) f(x + \mathrm{d}x, y + \mathrm{d}y, z + \mathrm{d}z), \tag{5.5}$$

where $g(x, y, z)$ is the output filtered point cloud, $f(x, y, z)$ is the original point cloud, $\mathbf{w}$ is the filter kernel, and a, b, c are the kernel size.

This thesis uses a convolution filter with a Gaussian kernel. The kernel calculates the new point's 3D position and colour based on the Gaussian distribution and the specified distance that forms a sphere in 3D space. The result of the filter is smooth colour and position of points (see Figure 5.3).



(a) Original point cloud (b) Point cloud with gaussian filter

Figure 5.3: Comparison of the same point cloud before and after Gaussian filter.

Chapter 6: Results

Contents

This chapter presents the results of the designed system for calibration of the camera-to-lidar extrinsic parameters (see Section 6.1) and the data fusion from the LiDAR and cameras sensors (see Section 6.2). To qualify and quantify the performance, the calibration process was performed on simulated and real-world data. The performance of the proposed filters on the colour fusion is then analysed and compared in simulation and on real-world data qualitatively.

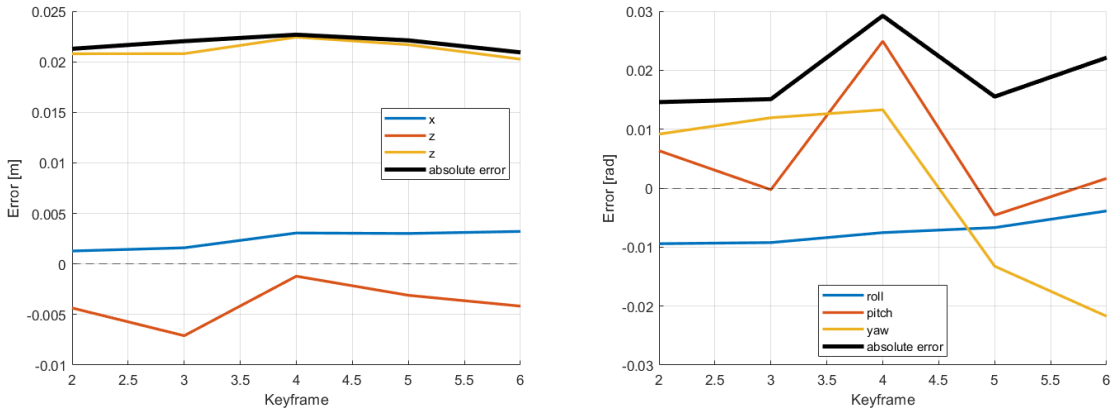
6.1 Calibration

To verify and validate the functionality of the proposed methods as well as to quantify the performance, the accuracy of the calibration process was performed on precise simulated data. Having a known extrinsic calibration, the obtained results can be seamlessly compared and the performance can be quantified. Throughout the validation, the same checkerboard pattern is used in simulated environment as well as in real-world experiments. This checkerboard pattern board has 8 interior columns and 6 interior rows with the square size of 0.052 meters, and distance between edge and checkerboard pattern 0.014, 0.018, 0.028, 0.055 meters (see Figure 4.2).

In the simulated environment, a robot equipped with a camera and a LiDAR was moving around a static calibration pattern while performing the calibration. In the real-world conditions, it is often more practical to move the calibration pattern in proximity to static sensors, but the calibration process is independent on the movement of both components. Despite the ideal simulation environment, many keyframes could not be captured. The problem was with the *cv::findChessboardCorners* library function, which could not find a checkerboard pattern on the image data. Nevertheless, several keyframes with the detected board in camera and LiDAR data were captured. Table 6.1 compares precise and final obtained calibration parameters from simulation data after just six keyframes. Figure 6.1 also quantifies the errors between the correct transformation and the data obtained from the program depending on the number of keyframes.

Parameter	Correct	Estimated	Error
x [m]	0.200	0.196781	0.00322
y [m]	0.000	0.004171	-0.00417
z [m]	-0.106	-0.126256	0.02026
roll [rad]	-1.571	-1.56713	-0.00387
pitch [rad]	0.000	-0.00165	0.00165
yaw [rad]	-1.571	-1.54927	-0.02173

Table 6.1: Comparison of the true and the estimated values of calibration parameters from simulation data.



(a) Translation errors from simulation data.

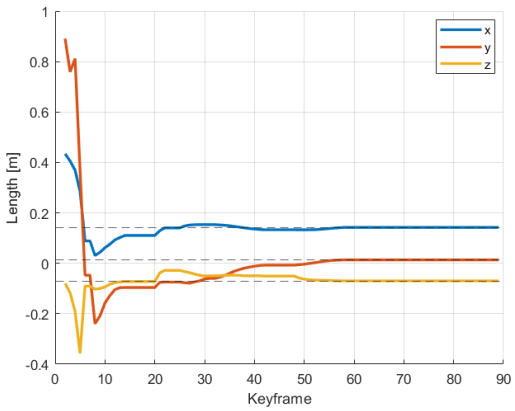
(b) Angles errors from simulation data.

Figure 6.1: Errors from simulation data depending on the number of keyframes.

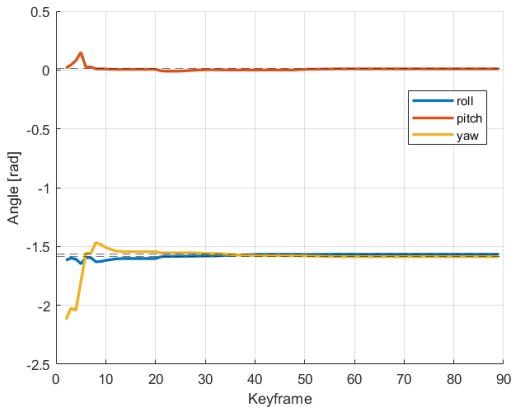
The performance of the LiDAR-camera extrinsic calibration on real data is tested using 128 channel Ouster OS1-128 LiDAR and mvBlueFOX colour camera with resolution 640x480 pixels as described in Chapter 3. Both sensors are mounted together on the UAV, as shown in Figure 6.2. UAV was steadily placed with the calibration checkerboard pattern board moving and thus changing the angle of inclination relative to the camera. Board was moving within the overlapping field of view of the camera and LiDAR.

The same calibration board with the same parameters was used as in the simulation test. A minimal distance of 15 pixels was set to differentiate individual keyframes in the image data. That means that a board had to be moved at least 15 pixels to capture the next keyframe. 89 keyframes were captured during the test with real data. At every keyframe, the checkerboard pattern board was detected in the data of both sensors. From the second keyframe, the calibration parameters were estimated until the end of the test. The Figure 6.3 shows that at the beginning, the parameters of translation vector and quaternions changed a lot. Later they settled at almost the same values. The image from the camera, shown in the Figure 6.4, is overlaid with the data from LiDAR after obtaining the final extrinsic calibration.

Figure 6.2: Camera-LiDAR setup on UAV.



(a) Translation vectors from real data.



(b) Angles from real data.

Figure 6.3: Estimated geometrical transformation from real data depending on the number of keyframes.

6.2 Fusion

To verify and validate the functionality and the quality of the proposed method, the fusion process was performed on precise simulated data first. Having a precise known extrinsic calibration, the obtained fusion results can be precisely compared without errors caused by inaccuracies in the calibration and in the state estimation of the robot using different filters type. Then, the fusion process was performed on precise real-world data to verify functionality on the inaccurate data.

Figure 6.4: Camera image overlay with the estimated camera-to-LiDAR geometrical transformation.

6.2.1 Simulated data

During all the fusion tests, a UAV equipped with two cameras and a LiDAR was moving inside a simulated building in the virtual environment (see Figure 6.5). Because the LiDAR sensor could catch some part of the UAV, e.g. rotors, points near the UAV are ignored. Also, points far from UAV are removed because the distance negatively affects the colourisation quality from the camera. The valid point cloud distance from the UAV was hence experimentally set from 1 to 37 metres. The voxel grid filter with a voxel size of 0.05 metres was used to reduce the number of points and hence to reduce the computational load. Several Gaussian kernel convolution filters and kernel matrix settings were tested for the quality of color information filtration. The influence of these filters on the resulting point clouds are compared below.

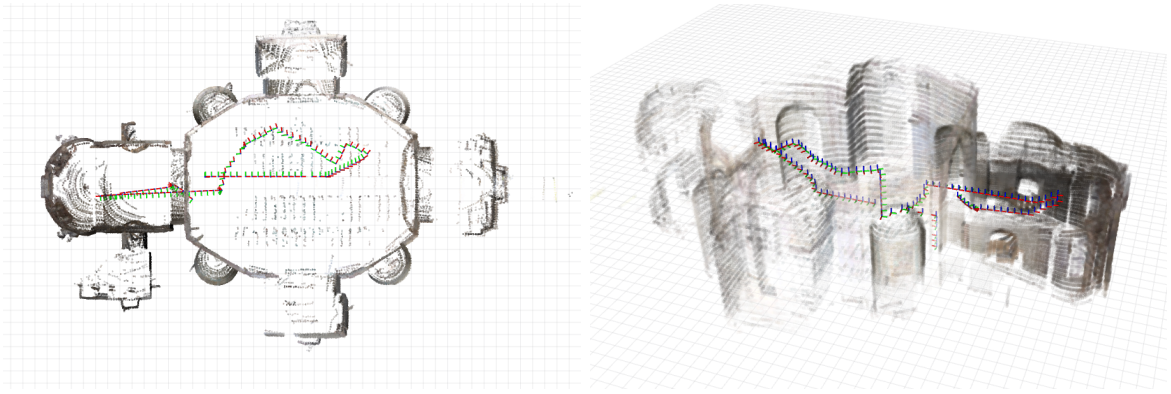
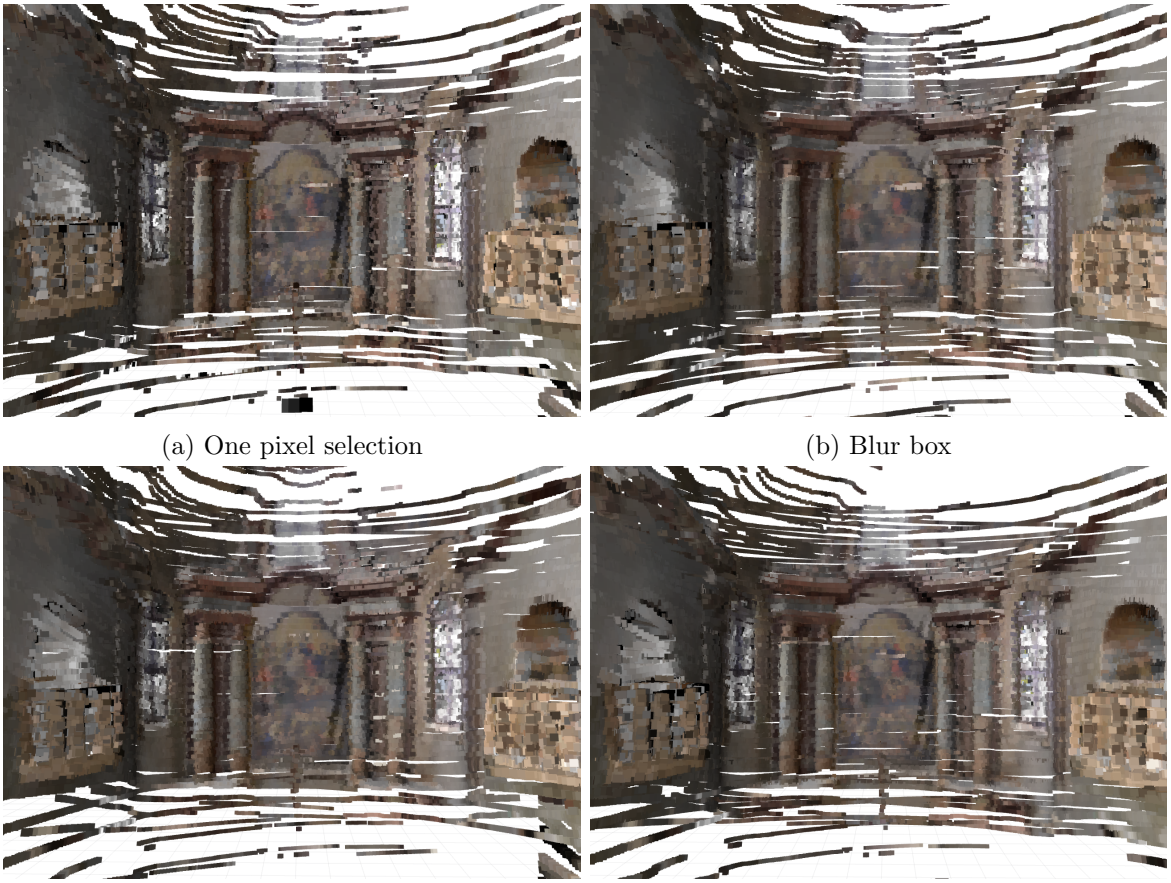


Figure 6.5: UAV trajectory during a simulated fusion test.

The first test compared several setting of kernel matrices to select colour information from the image. Figure 6.6a shows the result of the fusion using only one pixel to select RGB information. Using single-pixel projection of the laser points onto the image plane can cause noise in colours, especially in practice where spatial and temporal inaccuracies are present.

Small noise can be even seen in Figure 6.6a. The use of noise reduction techniques (blurring) on the data is shown in the rest of Figure 6.6. The differences can be observed at the edges of the columns of the building. The blur box method (Figure 6.6b) minimizes the noise in colour, keeps the colours smooth and the spatial information sharp. For this reason, the blur box method is validated as feasible in our scenario.



(a) One pixel selection

(b) Blur box

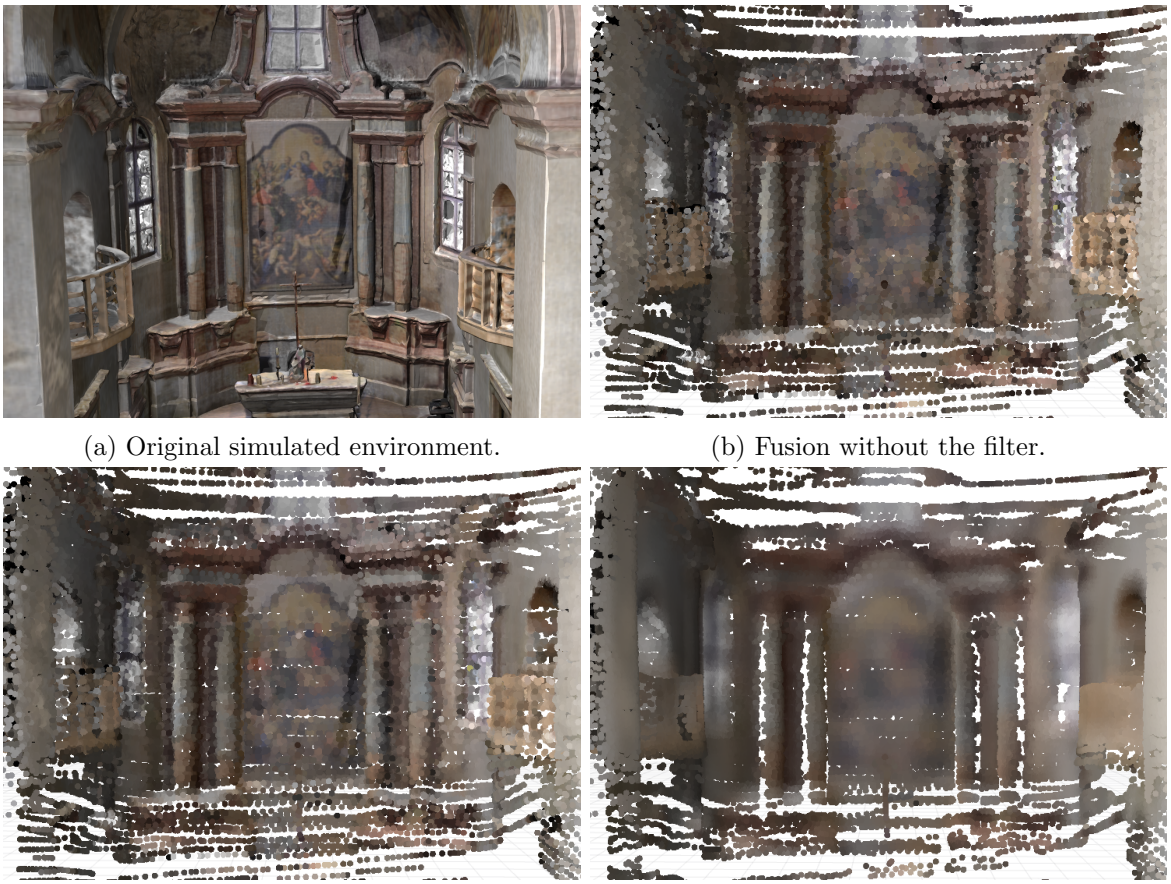
(c) Gaussian blur 3x3

(d) Gaussian blur 5x5

Figure 6.6: Comparison of the results of several kernel matrices for selecting colour information.

In the second test, several Gaussian kernel convolution filters were compared (see Figure 6.7). During this test, ground truth location was used. That means that the true geometrical transformation between LiDAR and stable coordinate systems was known at every time. The Figure 6.7b shows the final point cloud without using the convolution filter. Due to precise known extrinsic transformation, the resulting point cloud is adequately colourised without significant visible noise. The resulting point cloud with the convolution filter set at a small distance is shown in Figure 6.7c. The result is adequately colourised, still has sharp colours and also does not contain noise. Compared to that in Figure 6.7d, with the convolution filter set at a larger distance, the point cloud colours are blurred. This test shows that if the precise value of the transformation between LiDAR and the stable frame is known at any time, it is enough to set the filter to short distance values of filtration or not to use it at all. Therefore, these filters are more important in the real world applications, where inaccuracies

of many types occur.



(a) Original simulated environment.

(b) Fusion without the filter.

(c) Filter with a distance of 0.08 meters.

(d) Filter with a distance of 0.3 meters.

Figure 6.7: Comparison of the original simulated environment and resulting point clouds with several setting of Gaussian kernel convolution filters with precise transformation.

To simulate the real-world conditions on the simulated data, the same qualitative analysis is performed after an artificial noise is added in the state estimation of the aerial robot (see Figure 6.8). In other words, an error is added to the transformation between the LiDAR sensor and a stable coordinate system. Normal distribution with zero means of the distribution and 0.1 standard deviations was used. This better corresponds to testing on real data, where it is not possible to use ground truth location and the estimated pose of the robot always contains inaccuracies. The resulting point cloud coloured without using the convolution filter contains a colour error and noise (see Figure 6.8a). The Figure 6.8b shows that the resulting point cloud with the convolution filter set at a small distance filtering has significantly better colourisation quality. In this point cloud, the points are still not completely spaced smoothly. This problem is solved by a convolution filter set at a larger distance filtering, but the resulting point cloud has blur colourisation (see Figure 6.8c). This test shows that if the precise value of the transformation between LiDAR and the stable frame is not known, the convolution filter set at a feasible distance has sufficient colourisation quality and spatial smoothing.

(a) Fusion without the filter.

(b) Filter with a distance of 0.08 meters.

(c) Filter with a distance of 0.3 meters.

Figure 6.8: Comparison of resulting point clouds with several setting of Gaussian kernel convolution filters with precise transformation.

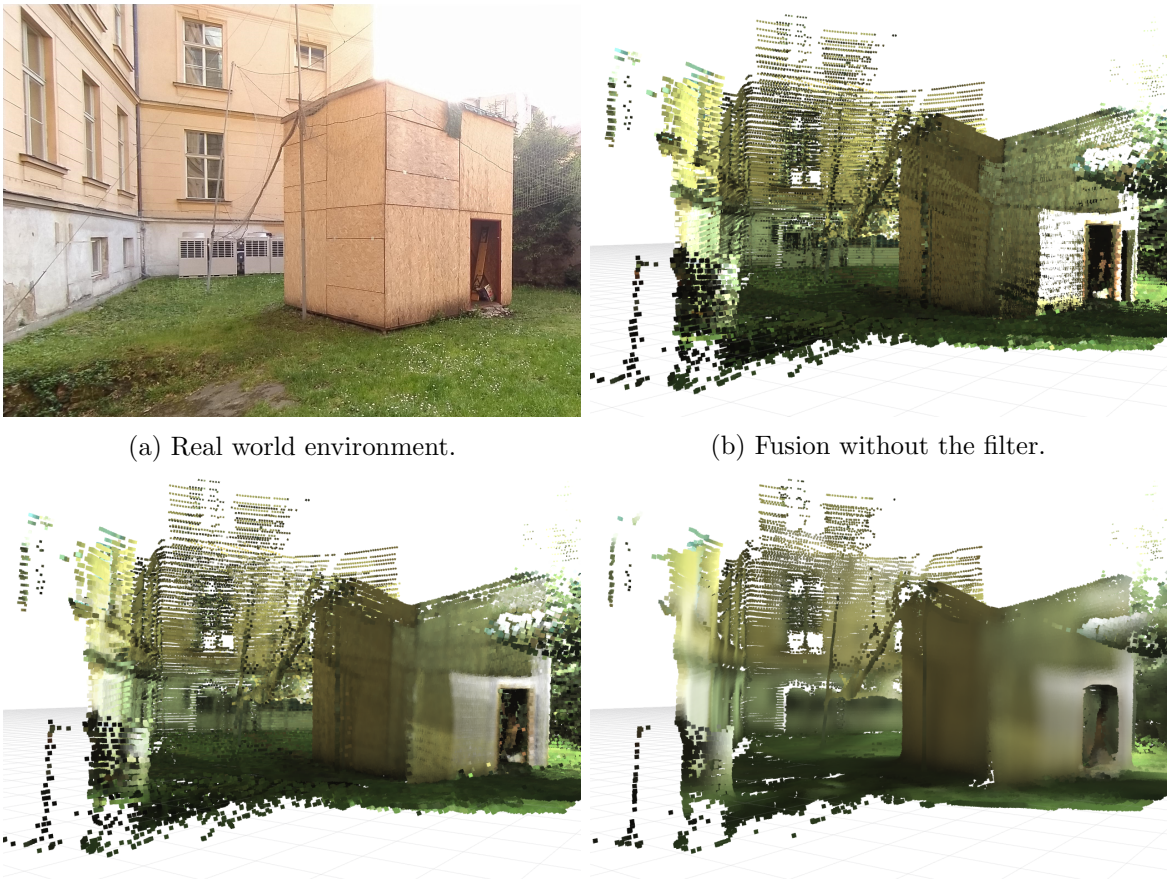
The final result is that the blur box method to select colour information and convolution filter set to short distance values of filtration is validated as feasible in our scenario with or without the precise value of the transformation to a stable frame.

6.2.2 Real-world data

To obtain the transformation between the camera and the LiDAR in the real world, the calibration algorithm described in Chapter 4 was used. During the calibration process approximately 400 keyframes were obtained. For the fusion test on the real-world data, a UAV equipped with a single camera and a LiDAR was used in an outdoor environment. The valid point cloud distance from the UAV was experimentally set to the range from 1 to 37 metres, exactly the same as in the simulation. The same voxel grid filter with a voxel size of 0.05 metres was used to reduce the number of points and reduce the computational load. The same filtration settings were used as in the analysis on the simulated data (i.e., the kernel matrix and the blur box method).

In this test, several Gaussian kernel convolution filters were compared (see Figure 6.9).

Figure 6.9a shows the real-world environment. The resulting point clouds from the fusion process have a little different colour. This is caused by using white balancing in the image data preprocessing. The Figure 6.7b shows the resulting point cloud without the use of the convolution filter. This point cloud contains a visible colour error and noise. The resulting point cloud in Figure 6.9c shows that using the convolution filter set at a small distance has significantly better colourisation quality. Compared to that, Figure 6.9d shows the convolution filter set at a more large convolution kernel size. This results in to point cloud with heavily blurred colour and smooth spatial point distribution.



(a) Real world environment.

(b) Fusion without the filter.

(c) Filter with a distance of 0.08 meters.

(d) Filter with a distance of 0.3 meters.

Figure 6.9: Comparison of the real-world environment and resulting point clouds with several setting of Gaussian kernel convolution filters.

This test verified that the proposed method also works on real data. In addition, it also confirmed that the settings tested on the simulated data are also validated as feasible in our real-world scenario.

The quality of the coloured point clouds produced by the method described in this thesis is adversely impacted by localization errors, camera distortion parameters, quality of the extrinsic calibration of the camera and the LiDAR synchronisation depending on the speed of movement of the UAV. The feasible filter settings depend on the accuracy of these parameters and the desired quality of the inspection task. Glass is also one of the sources of errors, which are created by the LiDAR sensors themselves, and it affects the colour quality of

the resulting point cloud. The effect of the glass can be seen in Figure 6.10. Another source of errors are objects from fine materials with holes, such as a net (see Figure 6.9). The LiDAR detects these objects, but even if the precise calibration is known, the camera detects the background colour. The feasible filter settings in this thesis cannot solve the error caused by the glass and the specific objects made from fine materials.

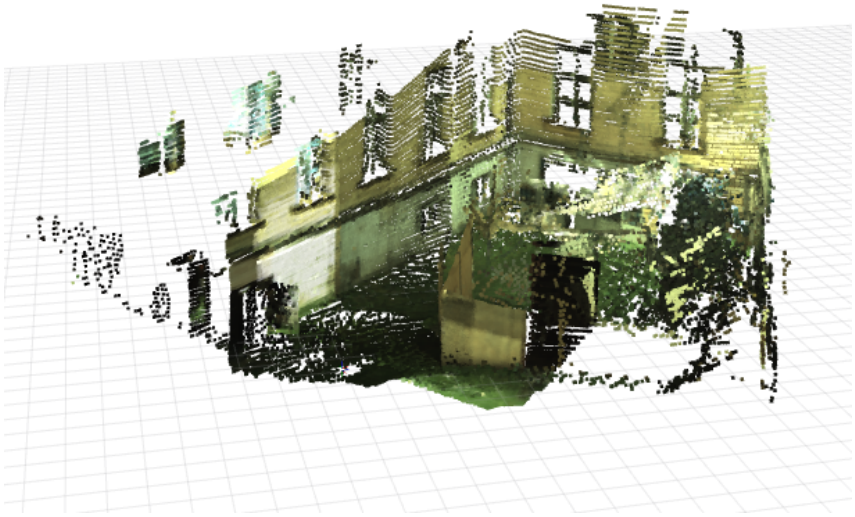


Figure 6.10: Resulting point cloud of the real-world data fusion.

Chapter 7: Conclusion

Contents

In this thesis, a method to fuse the point cloud data and the RGB information from a set of cameras was developed. Firstly, the optimization task was formulated and solved to obtain precise extrinsic calibration among fused sensors. The developed method goes beyond [16] by removing the need for a calibration pattern placement and does not need any apriori information. Next, the image projection was used to fuse 3D point clouds with colour information from cameras to colourize spatial information measured by precise laser scanners mounted onboard an autonomous robot. Several spatial and colour filtering methods were tested, implemented, and applied to reduce inaccuracies in sensing and fusion. The calibration of extrinsic parameters was validated and verified in the simulation as well as on the real-world datasets. It was demonstrated experimentally that the method is able to obtain consistent results, which are improved as more samples are added into the optimization process. The multimedia materials used in this thesis are available at `http://mrs.felk.cvut.cz/theses/fischer2021`.

In this thesis, the tasks given by the following list were successfully completed.

- The problem of extrinsic parameters calibration of sensors was tackled in Chapter 4. The algorithmic solution can be run for each LiDAR-to-camera link to obtain individual calibrations.
- A method to fuse the point cloud measured by a LiDAR with the RGB information from a set of cameras was developed. The fusion algorithm and implemented filtration of data are described in Chapter 5.
- The algorithm described in Chapter 4 was tested and analysed in simulated and real-world conditions in Section 6.1. Furthermore, the functionality of the calibration algorithm was quantitatively compared with the precise transformation from simulation.
- Results of the fusion algorithm within simulated conditions were tested, analysed, and described in Section 6.2. Also, the settings of the implemented filters were compared and qualitatively analysed for their effect on the resulting point cloud.
- Real-world data calibration algorithm results were used as input to the fusion algorithm in Section 6.2.2. Results in real-world conditions were tested, analysed, and described.

7.1 Future Work

In the thesis, calibration methodology with functional results was presented. The accuracy of the calibration method can still be improved by other optimization, such as Kalman filtering.

The calibration of colour information from the camera can further extend the work. White balance, especially for indoor measurements, will improve the appearance and refine the result. This may help the human operator with assessing and segmentation of the results. Another extension can implement filtration depending on the brightness of the colour while flying in dark areas while the robot carries its own light source. With the prioritization of lighter colour information, a more realistic result can be obtained.

Bibliography

[1] V. De Silva, J. Roche, and A. Kondoz, "Robust fusion of lidar and wide-angle camera data for autonomous mobile robots," *Sensors*, vol. 18, no. 8, Aug. 2018.

[2] A. Asvadi, L. Garrote, C. Premebida, P. Peixoto, and U. J. Nunes, "Multimodal vehicle detection: fusing 3D-LIDAR and color camera data," *Pattern Recognition Letters*, vol. 115, Nov. 2018.

[3] J. Nikolic, M. Burri, J. Rehder, S. Leutenegger, C. Huerzeler, and R. Siegwart, "A UAV system for inspection of industrial facilities," in *2013 IEEE Aerospace Conference*. IEEE, Mar. 2013.

[4] G. Copani, W. Terkaj, and T. Tolio, Eds., *Factories of the Future: The Italian Flagship Initiative*, 1st ed. Cham: Springer International Publishing : Imprint: Springer, 2019.

[5] G. Albeaino, M. Gheisari, and B. W. Franz, "A systematic review of unmanned aerial vehicle application areas and technologies in the aec domain," *Journal of Information Technology in Construction*, vol. 24, 2019.

[6] M. Beul and S. Behnke, "Analytical time-optimal trajectory generation and control for multirotors," in *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, Jun 2016.

[7] W. Kwon, J. H. Park, M. Lee, J. Her, S.-H. Kim, and J.-W. Seo, "Robust autonomous navigation of unmanned aerial vehicles (Uavs) for warehouses' inventory application," *IEEE Robotics and Automation Letters*, vol. 5, no. 1, Jan. 2020.

[8] K. P. Valavanis and G. J. Vachtsevanos, Eds., *Handbook of unmanned aerial vehicles*. Dordrecht: Springer Netherlands, 2015.

[9] P. B. Quater, F. Grimaccia, S. Leva, M. Mussetta, and M. Aghaei, "Light Unmanned Aerial Vehicles (UAVs) for Cooperative Inspection of PV Plants," *IEEE Journal of Photovoltaics*, vol. 4, no. 4, Jul. 2014.

[10] 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 *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. Limassol: IEEE, Sep. 2017.

[11] M. Kaamin, N. A. Idris, S. Mohd Bukari, Z. Ali, N. Samion, and M. Anjang Ahmad, "Visual Inspection of Historical Buildings Using Micro UAV," *MATEC Web of*

Conferences, vol. 103, 2017. [Online]. Available: http://www.matec-conferences.org/10.1051/matecconf/201710307003

[12] Universiti Tun Hussein Onn Malaysia, 86400 Parit Raja, Batu Pahat, Johor, MALAYSIA, H. Yusof, and M. Anjang Ahmad, "Historical Building Inspection using the Unmanned Aerial Vehicle (Uav)," *International Journal of Sustainable Construction Engineering and Technology*, vol. 11, no. 3, Jun. 2020.

[13] Qilong Zhang and R. Pless, "Extrinsic calibration of a camera and laser range finder (Improves camera calibration)," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 3. Sendai, Japan: IEEE, 2004.

[14] U. Ranjith and H. Martial, "Fast Extrinsic Calibration of a Laser Rangefinder to a Camera," Ph.D. dissertation, Carnegie Mellon University, Pittsburgh, Pennsylvania, Jul. 2005.

[15] W. Dong and V. Isler, "A Novel Method for the Extrinsic Calibration of a 2D Laser Rangefinder and a Camera," *IEEE Sensors Journal*, vol. 18, no. 10, May 2018.

[16] S. Verma, J. S. Berrio, S. Worrall, and E. Nebot, "Automatic extrinsic calibration between a camera and a 3D Lidar using 3D point and plane correspondences," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. Auckland, New Zealand: IEEE, Oct. 2019, pp. 3906–3912.

[17] W. Maddern and P. Newman, "Real-time probabilistic fusion of sparse 3D LIDAR and dense stereo," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Daejeon, South Korea: IEEE, Oct. 2016.

[18] S. M. Abbas and A. Muhammad, *Outdoor RGB-D SLAM Performance in Slow Mine Detection*. Frankfurt am Main: VDE-Verl., 2012.

[19] A. Mastin, J. Kepner, and J. Fisher, "Automatic registration of LIDAR and optical images of urban scenes," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*. Miami, FL: IEEE, Jun. 2009.

[20] T. Lowe, S. Kim, and M. Cox, "Complementary Perception for Handheld SLAM," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, Apr. 2018.

[21] J. Zhang and S. Singh, "Visual-lidar odometry and mapping: low-drift, robust, and fast," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. Seattle, WA, USA: IEEE, May 2015.

[22] P. Vechersky, M. Cox, P. Borges, and T. Lowe, "Colourising Point Clouds Using Independent Cameras," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, Oct. 2018.

[23] W. Moussa, M. Abdel-Wahab, and D. Fritsch, "Automatic Fusion of Digital Images and Laser Scanner Data for Heritage Preservation," in *Progress in Cultural Heritage Preservation*, D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, M. Ioannides, D. Fritsch, J. Leissner, R. Davies, F. Remondino, and R. Caffo, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, vol. 7616.

[24] A. Abdelhafiz, B. Riedel, and W. Niemeier, "Towards a 3d true colored space by the fusion of laser scanner point cloud and digital photos," 01 2005.

[25] H. Du, P. Henry, X. Ren, M. Cheng, D. B. Goldman, S. M. Seitz, and D. Fox, "Interactive 3D modeling of indoor environments with a consumer depth camera," in *Proceedings of the 13th international conference on Ubiquitous computing - UbiComp '11*. Beijing, China: ACM Press, 2011. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2030112.2030123

[26] What is camera calibration? The MathWorks. Accessed on February 4, 2021. [Online]. Available: https://www.mathworks.com/help/vision/ug/camera-calibration.html

[27] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge: Cambridge University Press, 2004. [Online]. Available: http://public.eblib.com/choice/publicfullrecord.aspx?p=256634

[28] Camera calibration and 3d reconstruction. Opencv dev team. Accessed on February 4, 2021. [Online]. Available: https://docs.opencv.org/2.4/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html

[29] G. R. Bradski and A. Kaehler, *Learning OpenCV : computer vision with the OpenCV library*. O'Reilly, 2008, accessed on February 4, 2021. [Online]. Available: https://www.bogotobogo.com/cplusplus/files/OReilly%20Learning%20OpenCV.pdf

[30] K. Sadekar. Understanding lens distortion. Learn OpenCV. Accessed on February 4, 2021. [Online]. Available: https://learnopencv.com/understanding-lens-distortion/

[31] H. Lee and O. Choi, "An efficient parameter update method of 360-degree vr image model," *International Journal of Engineering Business Management*, vol. 11, 04 2019, accessed on February 4, 2021. [Online]. Available: https://journals.sagepub.com/doi/full/10.1177/1847979019835993

[32] M. Achmad, G. Priyandoko, R. Roali, and M. Daud, "Tele-operated mobile robot for 3d visual inspection utilizing distributed operating system platform," *International Journal of Vehicle Structures and Systems*, vol. 9, no. 3, Sep. 2017, accessed on February 12, 2021. [Online]. Available: http://maftree.org/eja/index.php/ijvss/article/view/836

[33] Ouster os0-128 lidar sensor. General Laser. Accessed on February 4, 2021. [Online]. Available: https://www.general-laser.at/shop-de/lidar-de/ouster-os0-128-lidar-sensor-de

[34] Mvbluefox technical documentation. MATRIX VISION. Accessed on February 4, 2021. [Online]. Available: https://www.matrix-vision.com/manuals/mvBlueFOX/mvBF_page_introduction.html

[35] Micro Aerial Vehicles - platforms. Multi-robot Systems Group. Accessed on April 25, 2021. [Online]. Available: http://mrs.felk.cvut.cz/research/micro-aerial-vehicles

[36] P. An, T. Ma, K. Yu, B. Fang, J. Zhang, W. Fu, and J. Ma, "Geometric calibration for LiDAR-camera system fusing 3D-2D and 3D-3D point correspondences," *Optics Express*, vol. 28, no. 2, Jan. 2020, accessed on February 10, 2021. [Online]. Available: https://www.osapublishing.org/abstract.cfm?URI=oe-28-2-2122

[37] A. Duda and U. Frese, "Accurate detection and localization of checkerboard corners for calibration," 09 2018. [Online]. Available: http://bmvc2018.org/contents/papers/0508.pdf

[38] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, Jun. 1981.

[39] R. Honti, J. Erdélyi, and A. Kopáčik, "Plane segmentation from point clouds," *Pollack Periodica*, vol. 13, no. 2, Aug. 2018.

[40] Random sample consensus. Wikimedia Foundation, Inc. Accessed on April 25, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Random_sample_consensus

[41] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm," in *Proceedings Third International Conference on 3-D Digital Imaging and Modeling.* Quebec City, Que., Canada: IEEE Comput. Soc, 2001, pp. 145–152.

[42] Iterative closest point. Wikimedia Foundation, Inc. Accessed on April 25, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Iterative_closest_point

[43] V. Lepetit, F. Moreno-Noguer, and P. Fua, "EPnP: An Accurate O(n) Solution to the PnP Problem," *International Journal of Computer Vision*, vol. 81, no. 2, Feb. 2009.

[44] E. Riba Pi, "Implementation of a 3d pose estimation algorithm," Ph.D. dissertation, UPC, Escola Tècnica Superior d'Enginyeria Industrial de Barcelona, Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial, Jun 2015. [Online]. Available: http://hdl.handle.net/2117/77555

[45] International Conference on Frontiers of Intelligent Computing: Theory and Applications, S. C. Satapathy, V. Bhateja, B. L. Nguyen, N. G. Nguyen, and D.-N. Le, *Frontiers in intelligent computing: proceedings of the 7th International Conference on FICTA (2018). Volume 2 Volume 2.* Singapore: Springer, 2020, oCLC: 1122459182.

[46] Creating a datapointsfilter. Libpointmatcher. Accessed on February 4, 2021. [Online]. Available: https://libpointmatcher.readthedocs.io/en/latest/DataPointsFilterDev/

Appendices

List of abbreviations

Table 1 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
RGB	Additive colour model
SLAM	Simultaneous localization and mapping
ICP	Iterative closest point
RANSAC	Random sample consensus
PnP	Perspective-n-Point
EPnP	Efficient PnP

Table 1: Lists of abbreviations