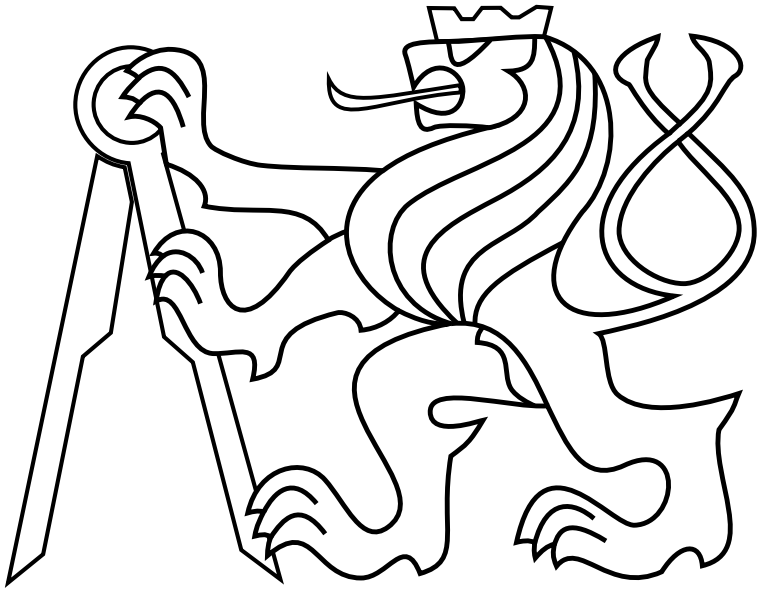
CZECH TECHNICAL UNIVERSITY IN PRAGUE

Faculty of Electrical Engineering

MASTER'S THESIS



Matching of multimodal features

Martin Fischer

Thesis supervisor: **Ing. Pavel Petráček**

Department of Cybernetics

January 2024

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

MASTER‘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 Computer Science**

Study program: **Open Informatics**

Specialisation: **Artificial Intelligence**

II. Master’s thesis details

Master’s thesis title in English:

Matching of multimodal features

Master’s thesis title in Czech:

Zarovnávání multimodálních příznaků

Guidelines:

The thesis aims to study relations among features typically detected in two sensory modalities: 2D RGB images and 3D data from rotating LiDARs or depth cameras. The task is to review and compare algorithms for detecting salient features in both modalities and select methods to be utilized and evaluated in matching two multimodal sets of features based on the comparison. The following tasks are supposed to be solved:
1) Review and compare algorithms for detecting 2D features in RGB images and 3D features in scans from 3D LiDARs or depth cameras [1].
2) Review algorithms for associating features among the two modalities and summarize how deeply the topic is studied in the literature [2].
3) Select feature-detection methods most suitable for cross-modality association and argue why.
4) Utilize the selected methods in an optimization task maximizing the alignment (i.e., perform matching [3]) between two cross-modality feature sets.
5) Evaluate the performance of the matching for selected pairs of feature detectors. Define advantages and disadvantages in contrast to the matching of single-modality sets.

Bibliography / sources:

[1] Yali Li, Shengjin Wang, Qi Tian, Xiaoqing Ding, „A survey of recent advances in visual feature detection,“ Neurocomputing, 2015.
[2] Debeunne, César, and Damien Vivet, "A Review of Visual-LiDAR Fusion based Simultaneous Localization and Mapping," Sensors, 2020.
[3] Arun, Somani; Thomas S. Huang; Steven D. Blostein, "Least-square fitting of two 3-D point sets," IEEE Pattern Analysis and Machine Intelligence, 1987.

Name and workplace of master's thesis supervisor:

Ing. Pavel Petráček Multi-robot Systems FEE

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

Date of master's thesis assignment: **14.02.2023** Deadline for master's thesis submission: **09.01.2024**

Assignment valid until: **22.09.2024**

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

III. Assignment receipt

The student acknowledges that the master'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 master'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 from the Multi-Robot Systems group of the Faculty of Electrical Engineering at the Czech Technical University in Prague, for his patience, guidance, valuable advice, improvements, and especially for the time, which was very helpful to me and this thesis.

I would also like to express my gratitude to my family for their support throughout my studies and the writing of this thesis.

I am also grateful to all my classmates who helped me with studying at the university and supported me in achieving better results.

Last but not least, I wish to thank my good friend Tereza for spending her time correcting this work.

Sincerely, thank you.

Abstract

This thesis deals with relationships among features detected in sensor data with different modalities: 2D RGB images and 3D data from rotating LiDARs or depth cameras. It describes fundamental features for both modalities and selects specific deterministic keypoint detectors. An algorithm was designed to match features extracted from different modalities for localization purposes. The algorithm uses the unique characteristics of each modality to find corresponding pairs of their keypoints by utilizing projection to 3D space, kd-tree structures and a nearest neighbour search. These identified pairs are subsequently iteratively aligned in 3D space to obtain a transformation between two data frames. The proposed deterministic algorithm was implemented and evaluated in a simulated environment across all selected detector pairs. The experiments showcase the best performance for the pair of the SIFT used in image data and the Shi-Tomasi detector used in 3D data, confirming the feasibility of such an approach for rapid odometry in control systems.

Keywords: multimodal, matching, registration, LiDAR, depth camera, position estimation, features, keypoints, Canny edge detector, Shi–Tomasi corner detector, SIFT

Abstrakt

Tato práce se zabývá vztahy mezi prvky detekovanými v datech senzorů s různými modalitami: 2D daty RGB snímků a 3D daty z rotujících LiDARů nebo hloubkových kamer. Popisuje základní příznaky pro obě modality a vybírá konkrétní deterministické detektory klíčových bodů. Pro účely lokalizace byl navržen algoritmus, který zarovnává příznaky extrahované z rozdílných modalit. Algoritmus využívá jedinečné charakteristiky každé modality k nalezení odpovídajících párů jejich klíčových bodů pomocí projekce do 3D prostoru, kd-stromových struktur a algoritmu hledání nejbližšího souseda. Takto identifikované páry jsou následně iterativně zarovnány ve 3D prostoru pro získání transformace mezi dvěma datovými rámci. Navržený deterministický algoritmus byl implementován a vyhodnocen v simulovaném prostředí pro všechny dvojice vybraných detektorů. Experimenty ukázaly nejlepší výsledek pro dvojici detektorů SIFT používaný v obrazových datech a detektor Shi-Tomasi používaný v 3D datech, což potvrdilo použitelnost takového přístupu pro rychlou odometrii pro řídicí systémy.

Klíčová slova: multimodální, zarovnávání, registrace, LiDAR, hloubková kamera, odhad pozice, příznaky, klíčové body, Cannyho hranový detektor, Shi–Tomasi detektor rohů, SIFT

Contents

List of Figures

List of Tables

Chapter 1: Introduction

Contents

With the advancement of multisource and multimodal sensors, various data representations of the same scene are captured through various sensor perceptions. The 2D images captured by lightweight cameras as a set of two-dimensional grids are the most popular data source representing scene information. Another prevalent category of sensors encompasses those employed in 3D imaging techniques. These techniques can be categorized into two primary groups: the first relies on 3D inference from 2D matching relationships, often utilizing structured light or stereo vision. The second category, more popularly adopted, is a 3D inference based on time-of-flight measurements, exemplified by LiDAR or depth cameras. However, in this latter category, models directly obtained from 3D sensors lack inherent corresponding relationships with 2D images, and they need to be coupled through matching algorithms.

The challenge of establishing corresponding relationships between 2D images and 3D data could also be represented as determining the position and orientation of a camera relative to a 3D model of a scene, also referred to as the image-based localization problem. It is an essential step in many applications, such as location recognition, Augmented Reality (AR), and visual navigation for autonomous vehicles.

Accurate localization serves as a foundational prerequisite for numerous navigation tasks. For instance, precise information about their spatial orientation in the environment allows autonomous mobile robots or pedestrians to plan optimal routes toward designated goal locations. Although the Global Navigation Satellite System (GNSS) provides accurate position estimates on a global scale, there exist situations where it faces significant errors or fails to provide position estimates altogether. Also, an apparent drawback of GNSS is its inability to provide information about other objects or the surrounding environment. Further details concerning these challenges and additional motivations can be found in Section 1.1.

Against the backdrop of these challenges, this thesis is dedicated to tackling localization in an intricate realm of multimodal domain. The proposed approach addresses the limitations of GNSS by harnessing the potential of two distinct sensors: a conventional RGB camera and a source of 3D information (a LiDAR or a depth camera). It uses a popular approach to robot

localization to match sensor data against a previously acquired map. These chosen sensors capture disparate modalities' data, spanning two-dimensional and three-dimensional spatial representations. The multimodal domain yield several challenges discussed in Section 1.2.

The premise of this thesis assumes the availability of an environment map where an autonomous robot operates. In the first scenario, the map is acquired through the deployment of either LiDAR or a depth camera. This map is represented in the form of a 3D point cloud. In contrast, the robot is equipped solely with a camera that captures the surrounding environment using colour images. In the second scenario, the premise involves the availability of an RGB environmental map captured by a camera (e.g., Google Street View). In this case, the map is represented by a collection of images, and the robot is equipped solely with a LiDAR that captures the surrounding environment using a 3D point cloud. In both cases, consequently, the need arises to fuse pre-existing map data with the obtained information.

Fusing image data with 3D point clouds presents a challenge due to differences in appearance and modalities. Existing methods address this issue by projecting the 3D data onto a 2D plane or reconstructing 3D point clouds from 2D images to facilitate data alignment for pose estimation [1]. To establish mutual correspondence between these distinct data sets, it will be necessary to identify features that accurately describe the respective space, regardless of the modality. Our intuition says that there should exist geometric structures, such as lines, planes, and corners, that can be captured in 3D maps and 2D images regardless of appearance differences, modality gaps, and scale. Nongeometrical features will also be used to avoid being limited to intuitive solutions. Therefore, one of the objectives of this thesis is to review and compare the 2D and 3D geometric and nongeometric feature descriptor correspondences regarding accurate and long-term camera localization.

Apart from selecting appropriate feature detectors, the challenges contain determining a suitable feature association and finding the transformation between the two feature sets. Such an algorithm should robustly identify corresponding point pairs from the given sets of points in different modalities. Subsequently, based on these identified pairs, the algorithm should calculate the mutual transformation between these modalities. This process aims to estimate a transformation between the 2D image and the 3D point cloud, which should be used for the localization in the 3D map.

1.1 Motivation

Autonomous robot navigation has been a significant subject of research for a long time. For example, self-driving cars (also known as autonomous and driverless cars) have been simultaneously studied and developed by universities, research centres, car companies, and companies of other industries worldwide since the middle 1980s [2], yet it is still a challenging and unsolved task even today. The underlying prerequisites for achieving successful autonomous navigation encompass precise localization of the robot and a comprehensive understanding of its surrounding environment. Currently, the GNSS solutions provide the primary localization mechanisms, furnishing unparalleled absolute positioning accuracy on the surface of the Earth.

Nevertheless, the efficacy of GNSS-based systems can be hindered by various factors. For instance, environments characterized by obstructions like tunnels or caves may impede

the availability of GNSS signals, rendering the systems ineffective. Furthermore, in scenarios such as urban canyons, the absence of a direct line-of-sight to satellites degrades the accuracy of GNSS solutions, which may result in considerable positioning errors. Given the paramount importance of ensuring safe and reliable autonomous navigation even these small inaccuracies are problematic. Also, GNSS systems cannot inherently offer information about the robot's surroundings. GNSS cannot detect or identify objects such as other vehicles, pedestrians, road signs, or obstacles in the environment. This limitation restricts the robot's ability to perceive and respond to dynamic changes in its surroundings and it makes it impossible to plan a path overall.

Autonomous robot navigation systems often incorporate complementary sensors such as LiDAR or depth cameras to address these limitations. Through these sensors, it is feasible to ensure relative localization until satellite-based localization can be re-established. These 3D sensors can also be employed for precise localization, provided an existing 3D environmental map is available beforehand. An example of such a map is shown in Figure 1.1b.

For instance, this 3D map can be obtained through a prior robotic exploration utilizing Simultaneous Localization and Mapping (SLAM) techniques. Alternatively, another viable option is to leverage recent advances in Structure-from-Motion (SfM) techniques [3], [4] and the fact that growing percentage of our world is covered by photos available on websites, such as Flickr, Google Street View (see Figure 1.1a for example), and Mapillary, which then make it possible to reconstruct the 3D structures on a large scale efficiently. These reconstructed maps together with data from 3D sensors can be used for precise localization.

The widespread use of LiDAR sensors is limited due to their high cost and heavy weight. Compared to that, cameras are low-cost and lightweight sensors, readily available and commonly used for visual-inertial-based pose estimation and mapping methods for various robot systems [5], [6]. Thus, the challenge for this thesis remains in utilizing conventional cameras, without the use of LiDAR, for localization within a 3D map.

The spatial relationship between 2D and 3D space provides the promotion and reference significance in developing computer vision applications and shares characteristics with robot visual navigation. In augmented reality, the precise alignment of 2D images and 3D point clouds is crucial for generating realistic and immersive virtual content that seamlessly interacts with the real-world environment. The spatial relationship between these modalities allows for creating compelling AR experiences, where virtual elements can be seamlessly integrated into the physical world, enhancing user interaction and understanding. Principally, it requires acquiring and maintaining an estimate of the camera position and orientation relative to some geometric representation of its surroundings.

The fusion of images and point clouds is nowadays increasingly sought due to its valuable spatial information about the external environment, not only for autonomous vehicles but across various applications. Both sensors provide rich and complementary data which can be fused and used by various algorithms and machine learning to sense and make vital inferences about the surroundings. The basis of multi-sensor fusion is accurate extrinsic calibration between sensors, that is, precise estimation of the relative transformation between sensors that establishes a geometric relationship between their coordinate systems. The quest for a robust and highly accurate algorithm for extrinsic calibration remains a contemporary challenge, as the outcome of the external calibration significantly influences the subsequent quality of the fusion of this data. Several methods have been developed to address the LiDAR and camera

extrinsic, for example using mutual information (MI) [7] or standard calibration objects, such as planar boards [8], planar boards with checkerboard patterns [9] or other calibration objects such as a circle-based object in [10] and board with four circular holes and a single metal trihedral corner as in [11]. But for solving this task, it is also possible to use place recognition and localization by estimating transformation from 2D-3D matches. If the proposed method of place recognition and localizations are sufficiently accurate, it is also possible to use it for the initial approximate alignment of this calibration between the camera and LiDAR or depth camera sensors if used on the same rigid robot body.

(a) An image from the Google Street View website [12]

(b) A point cloud map generated from LiDAR sensor data [13]

Figure 1.1: Illustration of an image and a point cloud map acquired by a camera and a LiDAR sensor, capturing an urban cityscape

1.2 Challenges of using multimodal data

Across various disciplines, data about the same environment can be collected using different types of detectors, operating under distinct conditions, or at varying observation times. The term "modality" serves to categorize each of these data acquisition frameworks. Given the intricate characteristics of natural and urban environments that robots commonly navigate, relying solely on a single modality seldom yields comprehensive situational awareness. Consequently, the adoption of multiple modalities concurrently has been introduced as a solution. However, this approach introduces several new challenges related to data alignment or heterogeneous data representation.

Data alignment is a primary challenge in multimodal data fusion. Different modalities often employ distinct coordinate systems, scales, and measurement units. Ensuring that data aligns accurately is fundamental for a wide array of applications, from robotics [14, 15] to healthcare [16, 17]. Robust registration techniques are required to establish correspondences between modalities and minimize alignment errors.

Furthermore, the representation of data across various modalities can be highly heterogeneous. While some modalities may provide 2D data, others offer rich 3D information. Although methods exist for projecting multidimensional data onto a unified framework, ad-

dressing issues like viewpoint variations, occlusions, and perspective distortions remains challenging.

In addition to addressing the heterogeneity of the data, extracting semantic information from multimodal data is another challenge. This is because specific modalities may excel in capturing visual details while others might provide depth or spatial information. To illustrate, the description of corner detection in a 2D image necessitates entirely different specifications compared to characterizing a corner detected within 3D data. Consequently, combining these diverse data sources to derive meaningful semantic insights is a complex and multifaceted task.

1.3 Related Work

Matching multimodal features, especially in the context of combining 2D and 3D data, has been a topic of considerable research interest in recent years. This is primarily driven by its relevance in applications such as robotics, computer vision, and augmented reality, where the fusion of different sensor modalities can provide a more comprehensive understanding of the environment. To accomplish this, many research efforts have been directed towards effectively aligning and matching 2D and 3D features.

While 2D modalities rely on cameras, LiDAR sensors that actively measure range information are frequently used for 3D modalities. Both LiDAR-based odometry and mapping [18,19] and visual-inertial systems [20,21], and even other techniques like thermal vision-based solution [22] have successfully been used to solve the SLAM problem. Some existing approaches tackle visual localization from a camera by detecting and matching visual features [23]. Other methods have explored solutions for visual localization using visual features extracted from LiDAR data [24]. These methods demonstrated sufficient accuracy and robustness in tested datasets.

However, SLAM frameworks that rely on a single modality are susceptible to sensor degradation specific to the sensor type. Therefore, a different solution is needed with combination of complementary sensor data in multimodal SLAM frameworks. A possible solution is the fusion of multimodal information, which stands out as a promising method, particularly exemplified by the Multi-Modal SLAM (MIMOSA) framework [25]. This approach is strategically devised to integrate data originating from diverse sensor modalities, such as cameras, LiDAR, and inertial sensors. The significance of such multimodal integration lies in its potential to address the intrinsic limitations of individual sensors, providing a more robust and accurate foundation for the SLAM process. MIMOSA is tailored to enabling resilient robotic autonomy in GNSS-denied and perceptually-degraded environments. This approach uses decoupled point cloud registration and a fusion of multiple odometry estimates relying on visible light and thermal vision. However, these frameworks integrate the outputs of the independent pipelines for each modality in the so-called back-end of the SLAM process and produce an output as a combination of the individual per-modality estimates. As written in previous sections, this thesis exploits the advantages of both sensors to localize accurately in 3D maps without being equipped with a LiDAR. Thus, the method of matching data from sensors directly is used instead of back-end integration in the proposed approach.

In recent research, addressing the challenge of multimodal feature matching predominantly relies on deep learning-based methodologies. One of the pioneering works in this

domain, dedicated to image-to-point cloud registration for robot localization, is the 2D3D-MatchNet [26]. This approach leverages the extraction of 2D and 3D keypoints through SIFT and ISS, respectively. Subsequently, a neural network with three branches is introduced to facilitate the learning of keypoint descriptors. The transformation estimation between the image and point cloud, established via 2D-3D correspondences, is realized using EPnP [27]. In the P2-Net [28], a batch-hard detector [29] is employed to generate a shared embedding between distinct sensor modalities. However, this method restricts its training and testing to sub-maps of sizes less than a meter, rendering it unsuitable for broader robotic applications. DeepI2P [30] split the image-to-point cloud registration problem into a classification and an optimization problem. It incorporates a cross-modality neural network to classify whether points fall within the image frustum. The classification outcomes are then utilized to construct a cost function governing the inverse camera projection. An optimization solver is subsequently engaged to derive the transformation minimizing the cost function. CorrI2P [31] adopts a cross-modality network for extracting overlapping regions and dense descriptors between the image and point cloud. This facilitates the establishment of dense image-to-point cloud correspondences, and an iterative RANSAC-based EPnP [27] is employed to estimate the relative pose. EFGHNet [32] embraces a divide-and-conquer strategy, breaking down the image-to-point cloud registration into four sub-networks that are sequentially applied and independently optimized. In contrast, I2PNet [33] introduces an end-to-end 2D-3D registration network, where all parts are differentially united and jointly optimized. This integration allows for the refinement of errors in subsequent modules, ultimately enhancing the robustness of the registration process.

However, using deep learning brings disadvantages. Deep learning models, especially deep neural networks, are inherently complex. This complexity can make them challenging to understand and interpret. This is particularly problematic in scenarios where model interpretability and explainability are crucial, such as safety-critical applications. Another disadvantage is that deep learning models often require vast amounts of labelled data to be trained effectively. In the context of multimodal feature matching, this can be a significant drawback, as collecting labelled data for both 2D and 3D modalities can be time-consuming and expensive. Additional issues might be associated with overfitting or a lack of generalization.

In contrast to deep learning-based methods, the literature contains fewer instances of approaches for camera localization in geometric maps constructed from LiDAR data via mathematical computation. One example introduced a technique for localizing autonomous vehicles in urban environments [34]. This method utilizes LiDAR intensity data to generate a synthetic representation of the mapped ground plane, subsequently matching it with the camera image by maximizing normalized mutual information. However, it offers a limited 3-DoF pose estimation. Conversely, another approach introduces estimation of the complete 6-DoF camera pose [35]. Its appearance prior (map) fuses geometric and photometric data to create a reference view, matched against the live image by minimizing the normalized information distance. Both methods conduct matching operations in 2D space, necessitating computationally intensive image rendering supported by GPU hardware. Additionally, their prior comprises LiDAR intensities or visual textures. In contrast, a method [36] exclusively relies on geometric information. In this work, researchers utilized the 3D positions of visual features with triangulated depths to construct local feature clouds, aligning them with the LiDAR map through graph optimization, obviating the GPU-intensive rendering step. However, this method has the drawback of requiring a robust initialization. Ground truth trajectory

data can be employed for this purpose, facilitating interpolating the first two keyframes' poses. In cases where such trajectory data are unavailable, an alternative initialization procedure is necessary, such as manual alignment between a reconstructed point set and the 3D LiDAR map.

Similar to the previous approach, this thesis uses key points that establish a shared embedding between the image and point cloud. Moreover, this thesis tries to avoid robust initialization conditions such as manual interventions, even in cases where the ground truth camera trajectory is unknown.

Many works exist that test and compare methods to extract 2D or 3D local features. A theoretical overview and evaluation of point cloud detectors and descriptors is given in [37] or with a specific focus on detectors in a publicly available Point Cloud Library[1] is given in [38]. [39] provides detailed descriptions for representative recent algorithms of visual feature detection, and [40] discusses the implementation and comparison of a range of the Open Computer Vision (OpenCV) library [2] feature detectors. Inspired by these works, this thesis tests and compares all pairs of selected 2D and 3D keypoints detectors for multimodal matching.

1.4 Outline

This thesis is partitioned as follows. Firstly, the principle of the pinhole camera is introduced, along with a detailed description of the sensors employed in this work, including colour cameras, depth cameras, and LiDAR, in Chapter 2. Additionally, in this chapter, the process of transforming these data between 2D and 3D spaces is described. Secondly, the detection of features from images is presented, in Chapter 3. This chapter outlines their basic categorization and provides an in-depth description of selected methods. Thirdly, the detection of features from 3D data is elaborated upon, in Chapter 4. This includes the direct detection of keypoints points from 3D space, as well as the conversion of 3D data into 2D and the application of methods for image feature detection. Next, a methodological approach is described for finding mutual transformations between modalities in 3D and 2D space, in Chapter 5. The chapter also introduces the developed algorithm for identifying mutual correspondences among keypoints and their subsequent alignment. Fifth, an analysis of multimodal matching is provided in Chapter 6. This chapter presents the simulation results and compares all proposed methods for feature detection, evaluating them in comparison to unimodal matching. Finally, the thesis is concluded in Chapter 7 by summarising the achieved objectives and future extensions of the work.

[1] Point Cloud Library, `https://pointclouds.org/`
[2] OpenCV, `https://opencv.org/`

1.5 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
bar index	$\bar{\mathbf{d}}$	a normalized column vector
lower index i	$\mathbf{R}_i$, $\mathbf{p}_i$	$\mathbf{R}$, $\mathbf{r}$ at discrete time step i

Table 1.1: Overview of the mathematical notation

1.6 Table of Symbols

Chapter	Symbol	Description
Preliminaries (Chapter 2)	f	Focal length
	m	Point on the image plane
	P	Point in space
	$\mathbf{K}$	Camera calibration matrix
	(c_x, c_y)	Principal point coordinates
	(u, v)	Image-pixel coordinates
	k_1, k_2	Radial distortion coefficient
	p_1, p_2	Tangential distortion coefficient
Image feature detection (Chapter 3)	I	Pixel intensity
	(x, y)	Point (pixel) coordinates
Methodology (Chapter 5)	I	Pixel intensity
	I_{max}, I_{min}	Image intensity extreme values
	(x, y)	Pixel coordinates

Table 1.2: Summary of symbols utilized

Chapter 2: Preliminaries

Contents

In this thesis, various sensors are used to capture the surrounding space. However, these sensors gather distinct data, resulting in different modalities. Specifically, a camera is utilized to capture RGB images, a depth camera records depth images, and a LiDAR captures 3D LiDAR range data. The data acquired from these sensors are registered in different dimensions, either 2D or 3D. The subsequent section will provide a more detailed description of these sensors to help understand how the data can be effectively transformed between 2D and 3D domains. Both the camera and the depth camera utilize the principle of a pinhole camera, which enables the representation from the 3D space into 2D and vice versa.

2.1 Pinhole Camera Model

The pinhole camera model is a fundamental concept in computer vision and computer graphics. It is a simplified mathematical representation of the formation of an image by the interaction of light rays with the camera's imaging sensor. In the pinhole camera model, a small aperture (or pinhole) is assumed to exist in front of the imaging sensor and light rays from the scene pass through this aperture to form an inverted image on the sensor plane. To simplify the mathematical depiction, the pinhole camera model incorporates an intermediary virtual image plane between the camera's focal point and the captured scene. Positioned at the focal length $f(m)$, this virtual image plane operates in parallel with the actual image plane behind the focal point, sharing an equivalent distance from the focal point. Employing the virtual image plane in front of the focus ensures that the projected image remains unrotated. The model also assumes that light rays travel in straight lines and do not undergo any distortion within the camera system. Consequently, the pinhole camera model is considered a perspective projection, where parallel lines in the 3D world converge to a point in the 2D image.

Mathematically, the pinhole camera model can be described using simple linear equations. Consider a pinhole camera model, where the centre of projection $\mathcal{F}_c$ serves as the origin of a Euclidean coordinate system. The plane $\mathcal{Z}$, defined by the equation $z = f$, is referred to as the image plane, where f represents the focal length, and $\mathcal{Z}$ coincides with the plane aligned along the optical axis. The intersection point of the optical axis with the image plane is termed the principal point, with the coordinates (c_x, c_y). Given a point P in the 3D space,

specified by its coordinates $P = (X, Y, Z)$, then its projection onto the image plane is the point $m = (u, v)$. This projection is defined as the intersection of the image plane with the line connecting point P and the centre of projection $\mathcal{F}_c$. This process is illustrated in Figure 2.1. Utilizing the theory of similar triangles, we can establish that point $(X, Y, Z)^T$ is projected onto the point $\left(f\frac{X}{Z}, f\frac{Y}{Z}, f\right)^T$, denoted as m, which lies on the image plane and can be termed the image point [41].

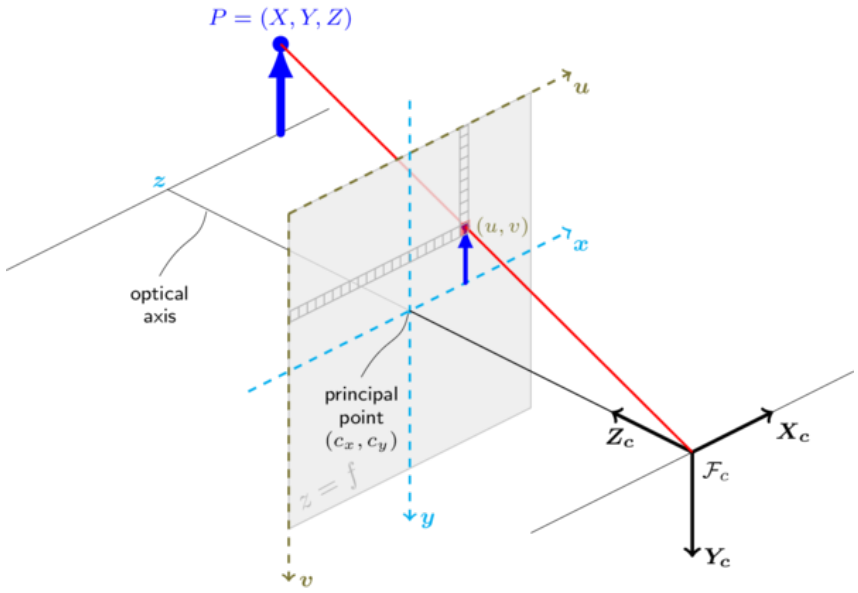


Figure 2.1: Projection of a point in space onto a point on the image plane [42]

When representing the points in space and on the image plane in matrix form, the central projection can be elegantly described as a linear mapping between their homogeneous coordinates. This mapping can be expressed using matrix multiplication

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}. \tag{2.1}$$

The matrix on the right-hand side of the equation is known as the camera matrix $\mathbf{K}$. Using the camera matrix, the Equation 2.1 can be written as

$$m \sim \mathbf{K}P. \tag{2.2}$$

By expressing the equations for image-pixel coordinates u and v, the central projection can be also described as

$$\begin{aligned} u &= \frac{fX}{Z} + c_x, \\ v &= \frac{fY}{Z} + c_y. \end{aligned} \tag{2.3}$$

In real-world camera systems, deviations from the ideal pinhole model occur due to various optical and mechanical imperfections. These imperfections lead to distortions in the

captured images. The two most common types of distortions are radial distortion and tangential distortion.

Radial distortion is caused by the lens system's inability to focus light rays ideally onto the image sensor. It results in straight lines appearing curved in the image. Radial distortion can be further divided into barrel distortion and pincushion distortion. Barrel distortion causes straight lines to curve outward from the centre of the image, giving a bulging effect. It is more prominent towards the edges of the picture. Pincushion distortion, on the other hand, causes straight lines to curve inward towards the centre, resulting in a pinched effect. Both effects are shown in Figure 2.2.

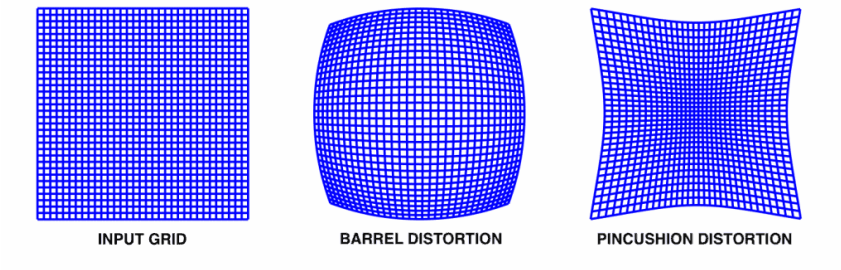


Figure 2.2: Illustration of barrel and pincushion distortion effect on a square grid [43]

Radial distortion is typically modelled using radial distortion coefficients, often denoted as k_1 and k_2. Mathematically, it can be expressed as

$$u_{distorted} = u(1 + k_1 r^2 + k_2 r^4), \tag{2.4}$$

$$v_{distorted} = v(1 + k_1 r^2 + k_2 r^4), \tag{2.5}$$

where $u_{distorted}$ and $v_{distorted}$ are the distorted image coordinates, u and v are the undistorted image coordinates, and r is the distance from the principal point.

Tangential distortion occurs when the lens and the image sensor are not perfectly parallel. This leads to a skewing effect where lines that should be parallel in the real world appear to converge or diverge in the image. Mathematically, tangential distortion can be represented as

$$u_{distorted} = u + (2p_1 uv + p_2(r^2 + 2u^2)), \tag{2.6}$$

$$v_{distorted} = v + (p_1(r^2 + 2v^2) + 2p_2 uv), \tag{2.7}$$

where $u_{distorted}$ and $v_{distorted}$ are the distorted image coordinates, u and v are the undistorted image coordinates, p_1 and p_2 are tangential distortion coefficients, and r is the distance from the principal point.

2.2 Sensors

As mentioned in the introduction of this chapter, this thesis utilizes three distinct sensors: LiDAR, depth camera, and RGB camera. Each of these sensors records its surrounding environment in different ways, resulting in diverse dimensions of data representation. A detailed explanation of the functioning of each sensor will be provided, along with the methods for representing the acquired data in both 2D and 3D formats.

2.2.1 LiDAR

Light Detection and Ranging (LiDAR) is an active sensor for capturing precise three-dimensional information about the surrounding environment. It operates by emitting laser pulses towards the target area and measuring the time the laser beam returns after reflection from objects in its path. By analysing the time-of-flight data, LiDAR systems can determine the distances to objects in the environment in a structured manner. The generated structure is a highly accurate (± 0.9 cm for ranges less than 10 m from LiDAR) 3D point cloud representing the scene.

Mathematically described, a LiDAR sensor emits a laser beam with a known wavelength $\lambda\,(m)$, propagating it towards the target surface. Upon encountering an object, such as the ground or a structure, the laser beam is scattered and reflected towards the sensor. The time taken for this round trip, known as "time of flight," is denoted as $\Delta\, t(s)$. The speed of light in the medium (usually air) is represented as $c\,(m/s)$. By applying the equation $c = \lambda \cdot f$, where $f\,(Hz)$ is the frequency of the laser pulses, the distance $d\,(m)$ to the object can be calculated as

$$d = \frac{c\Delta t}{2}. \tag{2.8}$$

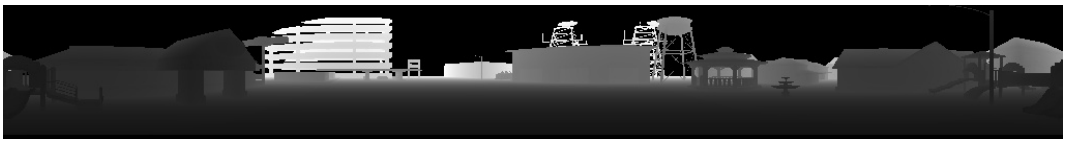
Since LiDAR measures the distances in the line of sight, additional angular information is required to reconstruct the 3D geometry of the scene accurately. For this purpose, LiDAR sensors are equipped with an azimuth angle $\theta(rad)$ and an elevation angle $\varphi(rad)$ encoders to determine the position of each laser point relative to the sensor's coordinate system. By combining distance measurements with angular information, LiDAR systems construct dense and precise 3D representations of any environment. An example of a 3D point cloud is shown in Figure 2.3b.

To acquire data from LiDAR in a digital form, the 3D point clouds are typically converted into a 2D matrix representation. This process involves discretizing the continuous 3D space into a grid and determining which grid cells are intersected by the LiDAR laser beams. Each cell in the grid corresponds to a pixel in the 2D matrix, and the intensity of the pixel is associated with the distance or reflectivity of the corresponding point in the 3D point cloud. The Figure 2.3a shows an example depth image constructed from LiDAR data. This projection of 3D point clouds to 2D matrices allows for efficient storage and processing of LiDAR data. It also allows for processing the data as images using computer vision analysis techniques.

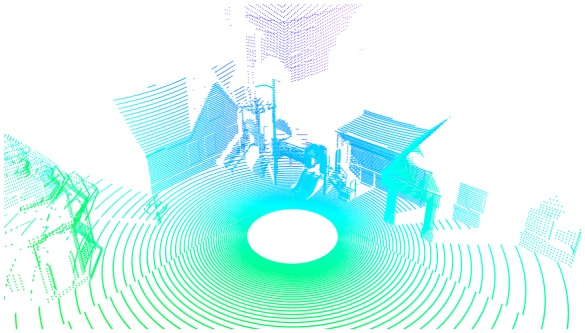
2.2.2 Depth camera

A depth camera, also known as a 3D camera or range camera, is an advanced imaging device that goes beyond the capabilities of traditional cameras by capturing both the visual appearance of a scene and the distance information from the camera to various objects in the scene. This allows to create depth maps or point clouds, providing a comprehensive representation of the 3D structure of the surrounding environment.

The depth camera's mathematical principles are closely related to the pinhole camera model, which forms the base model for traditional cameras. The pinhole camera model provides the foundation for the camera's imaging process, and the additional depth estimation



(a) Output in the form of a depth image



(b) Output in the form of a point cloud

Figure 2.3: Example and comparison of LiDAR outputs in 2D and 3D

techniques complement the pinhole camera model by providing valuable spatial information about the scene. The operation of depth cameras commonly uses one of two main approaches: triangulation and time-of-flight (ToF) measurement, which are essential components of the camera's depth estimation process.

In the triangulation approach, the depth camera emits a pattern of structured light or infrared signals onto the scene. The position of the light source emitting the structured light pattern differs from the position of the corresponding point on the object in the scene. The depth camera then captures the reflected light pattern, and by analysing the displacement of the pattern in the camera's image plane, the depth camera can calculate the distance $d\,(m)$ between the camera and the object using trigonometric principles. This process is known as the triangulation equation

$$d = \frac{Bf}{displacement}, \tag{2.9}$$

where $B\,(m)$ is the baseline distance between the light source and the camera's optical centre, and $f\,(m)$ is the focal length [44].

In the time-of-flight measurement approach, the depth camera emits short pulses of light or infrared signals, and the time taken for the signals to return to the camera's sensor is measured. The time of flight $\tau\,(s)$ is then converted to a distance measurement using the speed of light $c\,(m/s)$, as described by Equation 2.8 as

$$d = \frac{c\tau}{2}. \tag{2.10}$$

The output of the depth camera is a greyscale depth image, where shades of grey represent the distance of an object from the camera (see Figure 2.4a). The depth image obtained from the depth camera can be utilized to calculate the corresponding 3D point in the scene from a given pixel in the depth image. By applying the pinhole camera model and the known camera parameters, such as the focal length and camera intrinsic matrix, the depth

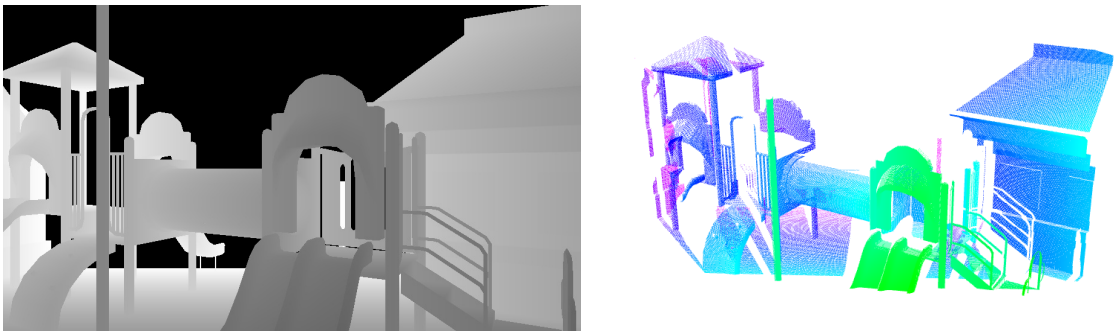
value of a specific pixel can be converted into 3D coordinates. The 2D-to-3D projection can be derived from Equation 2.3 as

$$X = \frac{(u - c_x)d}{f}, \tag{2.11}$$

$$Y = \frac{(v - c_y)d}{f}, \tag{2.12}$$

$$Z = d, \tag{2.13}$$

where u, v are the coordinates of the pixel in the depth image, d is the depth value of the pixel (u, v), and X, Y, Z are the 3D coordinates of the point in the camera's coordinate system corresponding to the pixel (u, v). The resulting point cloud obtained through this method is depicted in Figure 2.4b.



(a) Output in the form of a depth image

(b) Output in the form of a point cloud

Figure 2.4: Example and comparison of depth camera outputs in 2D and 3D

2.2.3 RGB camera

An RGB camera, or just a camera, is an essential imaging device. At its core, a camera captures visual information about the surrounding 3D environment and transforms it into 2D images (see Figure 2.5a). The fundamental principle underlying the operation of a camera is based on the pinhole camera model. While it may not fully represent real-world complexities, the pinhole camera model serves as a fundamental concept for camera imaging principles. Compared to the depth camera, the RGB camera utilizes three image sensors, each sensitive to one of the primary colours: red, green, and blue. By combining the intensity values from these sensors, the RGB camera generates full-colour images, representing the scene as perceived by the human visual system.

The mathematical description of the RGB camera's operation involves the principles of colour theory and image processing. Each image sensor in the RGB camera measures the intensity of light in its corresponding colour channel. Let $I_r(u, v)$, $I_g(u, v)$, and $I_b(u, v)$ denote the intensity values at pixel coordinates (u, v) in the red, green, and blue channels, respectively. These intensity values typically range from 0 (no light) to 255 (full intensity) for each channel. To create a full-colour image, the RGB camera combines the intensity values from the three channels. The combined colour image $I_{RGB}(u, v)$ at pixel coordinates (u, v) can be expressed as a vector in the RGB colour space

$$I_{RGB}(u, v) = [I_r(u, v),\ I_g(u, v),\ I_b(u, v)]^T. \tag{2.14}$$

Certain algorithms rely on single-channel images, necessitating the conversion of RGB images into a grayscale format. This transformation is essential for acquiring a black and white image representation, where each pixel contains a single intensity value. To achieve this, a weighted combination of the red, green, and blue colour channels is employed to represent each pixel in the grayscale image. The grayscale intensity $I_{gray}(u, v)$ at a pixel (u, v) can be mathematically expressed as

$$I_{gray}(u, v) = w_r I_r(u, v) + w_g I_g(u, v) + w_b I_b(u, v), \tag{2.15}$$

where w_r, w_g, and w_b determine the contributions of each colour channel to the grayscale intensity. Commonly used weights to achieve perceptually balanced grayscale images are $w_r = 0.2989$, $w_g = 0.5870$, and $w_b = 0.1140$, which ensure that the resulting grayscale image resembles the human perception of luminance. The grayscale image obtained through this process preserves the structure and visual information of the original RGB image while reducing the colour information to a single intensity value per pixel.

Applying the pinhole camera model with known camera parameters, including focal length and intrinsic matrix, enables the projection of 3D spatial coordinates of a specific point in space onto a 2D image plane. This plane constitutes the medium through which the sensor acquires data, manifested as colour or grayscale images, ultimately serving as the camera's output.

In the case of an RGB camera, obtaining information about the surrounding environment in the form of 3D data is significantly more complex. This complexity arises from the inherent limitation of RGB images in providing direct depth information, i.e., the distance of objects from the camera sensor. Unlike depth cameras or LiDAR, which can directly measure distances, RGB cameras capture colour information without any depth-related cues. As a result, the information about object distances is entirely missing in the RGB images, making the task of 3D reconstruction more challenging. A key concept in this process is the utilization of rays to establish correspondences between image pixels and 3D points.

In 3D reconstruction, each pixel in the image is considered to be the projection of a 3D ray originating from the camera's centre of projection (referred to as the focal point). These rays pass through the pixel coordinates on the image plane and project into 3D space.

A pinhole camera model is adopted, with the camera intrinsic matrix $\mathbf{K} \in \mathbb{R}^{3\times4}$. For a 3D point $P = (X, Y, Z)^T$ in the camera's coordinate system, it is projected onto the image plane as $m = (u, v)^T$ using the following equation

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim \mathbf{K} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}. \tag{2.16}$$

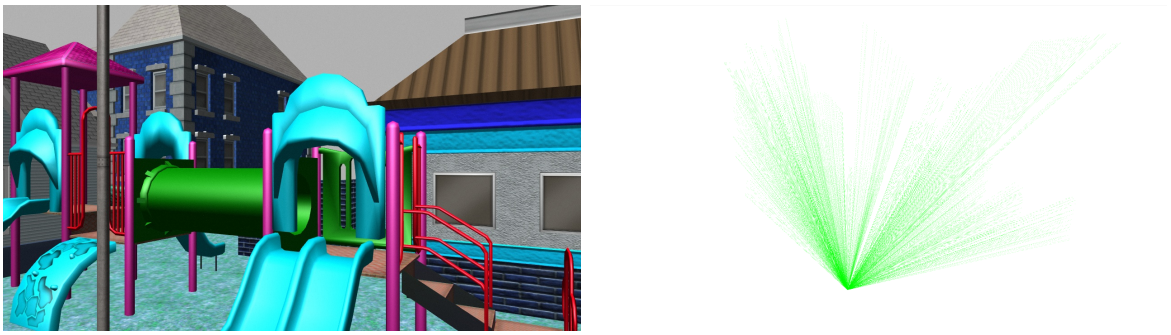
To perform 3D reconstruction, the process is reversed. Given a pixel (u, v) in the image, we consider it as the projection of a 3D ray originating from the camera's centre of projection. The direction vector of the ray $\mathbf{d} = (d_x, d_y, d_z)^T$ can be computed by applying the inverse of the camera intrinsic matrix $\mathbf{K}^{-1}$ to the pixel coordinates

$$\begin{bmatrix} d_x \\ d_y \\ d_z \end{bmatrix} = \mathbf{K}^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}. \tag{2.17}$$

Subsequently, the direction vector $\mathbf{d}$ is normalized to have a unit length

$$\bar{\mathbf{d}} = \frac{\mathbf{d}}{\|\mathbf{d}\|_2}. \tag{2.18}$$

Through the utilization of the direction vector $\bar{\mathbf{d}}$, rays are cast from the camera's centre of projection into 3D space. The resulting ray cloud obtained through this equation is depicted in Figure 2.5b.



(a) Output in the form of a colour image (b) Output in the form of a ray cloud

Figure 2.5: Example and comparison of RGB camera outputs in 2D and 3D

Chapter 3: Image feature detection

Contents

Visual features refer to visual structures in images and primitives and generally serve as distinctive image characteristics. Image feature detection involves identifying these structures, such as points, lines or curves, and regions, also more high-level classification as geometric, gradient-based, histogram-based and learning-based feature detectors, all to highlight salient visual cues in digital images. However, this is a low-level processing step that in all cases takes pixel intensities on input and produces image structures indicating different characteristic properties on output. The uses of visual features are numerous, ranging from classifying medical conditions in x-ray imagery to understanding the environment around a mobile robot. A shared concept among all application scopes is the focus on extracting salient (unique) and stable (temporally and spatially) features in an efficient manner.

Achieving the desired properties and robustness is still an ongoing problem, particularly due to many difficulties in the extraction, such as changes in scale, viewpoint, illumination and quality of images. A high-performance feature detector should show robustness to changing imaging conditions (for example, scene illumination, motion blur, and image noise) as well as satisfy other interests such as efficiency, accuracy and repeatability and generalizability. Besides, computational efficiency needs to be considered in real-time applications.

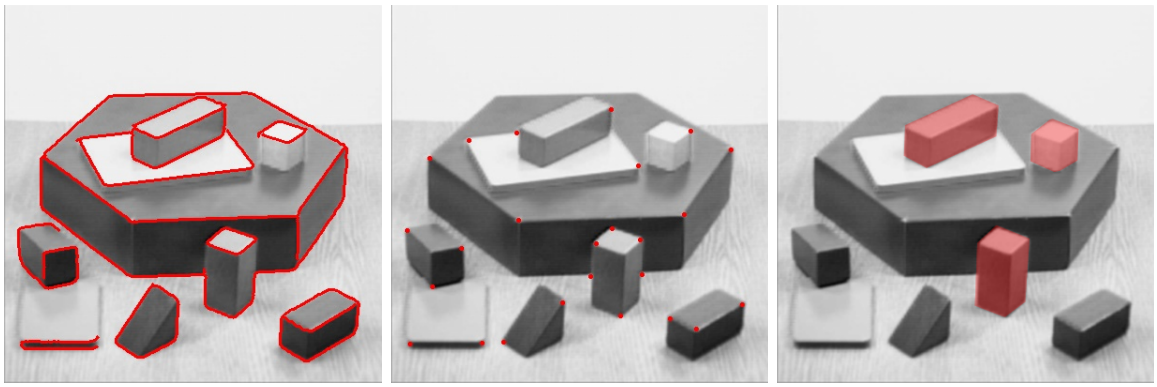
Visual features are closely related to human perceptual organization. Psychological research on Gestalt laws has demonstrated that the human visual system has a tendency to group low-level image components [45]. This system, according to Gestalt factors, organizes visual stimuli based on factors such as proximity, similarity, continuity, and closure. As computer vision seeks to simulate human visual perception using cameras and computers, visual feature detection draws inspiration from human visual perception. It follows that primitive features such as edges, contours, corners, and regions, highly relevant to human visual perception, are therefore utilized in visual feature detection [46]. This thesis is focused on geometric and gradient-based methods, but other types exist, such as local descriptors, histograms or learned vectors. In order to better describe the individual methods, it is essential to clarify the related concepts (see also Figure 3.1).

1. Edge refers to pixels at which the image intensities change abruptly, with pixel intensities being discontinuous at different sides of edges. In other words, there is a high-intensity gradient flow at an edge.

2. Contours/boundaries have ambiguous definitions, but in the context of low-level features, they refer to intersecting lines/curves of different segmented regions.

3. Corners are points at which two different edge directions occur in the local neighbourhood, being the intersection of two connected contour lines.

4. Regions refer to a closed set of connected points, with nearby and similar pixels being grouped to compose the region of interest [39].



(a) Edge (b) Corner (c) Region

Figure 3.1: Example of visual features in computer vision

The definitions mentioned above have natural and close connections. Specifically, contours or boundaries can be obtained through the tracking and connection of neighbouring edges. Corners are defined as the points where straight edge lines intersect, while the intersection curves between different regions make up boundaries. Visual feature detection methods are typically classified into three categories: edge detection, corner detection, and blob detection, which can also be referred to as the detection points or regions of interest. Edge and corner detection were chosen as suitable points of interest for cross-modal matching due to their clear definition and detection in both modalities: image data and 3D spatial data. In contrast to corners and edges, blobs refer to local regions of interest that typically lack a direct geometric representation in space. However, it is still possible to detect blobs in both modalities and due to their local region description, they also could have the potential for cross-modal matching. Therefore, they were also selected for this thesis.

The categories of edge, corner and block feature detection will be described in detail in the following section. To better showcase the distinctions of the categories, each section contains a detailed description of its representative method. Special emphasis is laid on methods with the potential of 3D space expansion, as is needed for connecting the image-feature theory with 3D-feature extraction, discussed in Chapter 4.

3.1 Edge detection

Edges are sharp transitions in image brightness that are commonly detected using differential operators. These operations capture the strength and position of discontinuities

in image brightness. Edges are also used for the detection of contours or boundaries, which represent more general information and describe the intersection of various regions. Thus edges have a crucial role in image interpretation.

When extracting edges from a two-dimensional image of a three-dimensional scene, the edges can be classified as a viewpoint dependent or a viewpoint independent. Viewpoint dependent edges may change as the viewpoint changes and typically reflect the geometry of the scene, such as objects occluding one another. Compared to that, a viewpoint independent edges represent inherent properties of the three-dimensional objects independent of the observer's position or orientation. It often corresponds to contours or boundaries representing distinct objects, surface markings, and surface shapes. These edges are crucial for object recognition, image matching, and 3D reconstruction, as they can be used to infer the underlying structure and geometry of the objects in the scene [47].

Various methods have been developed for edge detection, including gradient-based, Laplacian-based, and Canny [48] edge detection. Gradient-based methods use the first-order derivatives of the image intensity function to detect edges. Laplacian-based methods use the second-order derivatives to detect edges, and Canny edge detection uses a combination of both approaches, which have proved to be a robust and effective method due to its ability to identify edges with high accuracy while minimizing the detection of false edges [49].

The Canny edge detector is based on the computational theory of edge detection, which models edge detection as an optimization problem with three criteria; detecting edges with a low error rate, meaning the detector should accurately capture as many edges as possible, accurately localizing the point detected by the operator in the edge's centre and avoiding multiple markings of a given edge in the image while minimizing false edges [50].

The Canny edge detector involves a multi-stage process that begins by applying a Gaussian filter to smooth the image and reduce noise. The Canny algorithm explicitly uses a two-dimensional Gaussian function

$$G(x,y) = \frac{1}{2\pi\sigma^2} exp^{-\frac{x^2+y^2}{2\sigma^2}}, \tag{3.1}$$

where σ is the size of the standard deviation of the distribution determining the desired degree of smoothing and the scale of the edges in the image. Typically, a convolution kernel approximating a Gaussian filter with a σ of 1.0 is used. The smoothed image is then used to calculate the gradient, representing the rate of change in image intensity. The gradient is calculated using a Sobel operator, which returns a value for the first derivative in the horizontal direction G_x and the vertical direction G_y

$$G_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}, \quad G_y = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix}. \tag{3.2}$$

It is also possible to use different operators to approximate the first derivative of the image, for example, the Sobel–Feldman operator, the Scharr filter or the Prewitt operator.

The second stage of the Canny edge detector involves identifying the direction of edges by calculating the magnitude and direction of the gradient. The gradient magnitude G is calculated as the square root of the sum of the squares of the x and y gradients

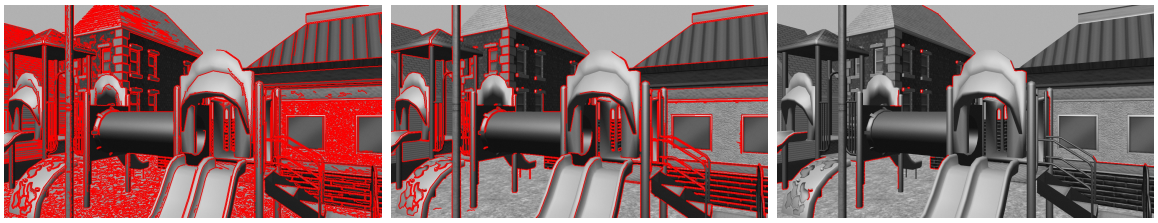
$$G = \sqrt{G_x^2 + G_y^2}, \tag{3.3}$$

whereas the gradient direction theta is given as the arctangent of the ratio of the y gradient to the x gradient with the formula

$$\theta = \arctan\left(\frac{G_y}{G_x}\right). \tag{3.4}$$

The gradient magnitude G of each pixel and its corresponding gradient direction θ is then used to suppress non-maximum edges. Each pixel is checked if its gradient magnitude is more significant than its two neighbouring pixels along the gradient direction. If the current pixel's gradient magnitude is the local maximum, it is retained as a firm edge point. Otherwise, it is suppressed. This step helps to reduce the thickness of the edges and remove weak or false edges.

The final stage of the Canny edge detector applies hysteresis thresholding, where the remaining edge pixels from non-maximum suppression are classified as either firm or weak edges based on their gradient magnitude. This thresholding step ensures that only the pixels with maximal magnitude in the gradient direction are recorded as strong edge points, highly likely to represent actual edges in the image. Figure 3.2 illustrates the contrast between different threshold values set in the image processing. To form the final edge map, the Canny edge detector performs edge tracking by connecting weak edges to firm edges if they are spatially adjacent. A weak edge is also classified as a firm edge if it is connected to a firm edge [48].



(a) Threshold = 20 (b) Threshold = 70 (c) Threshold = 180

Figure 3.2: The Canny edge detector with different threshold values

3.2 Corner detection

Corner detection is an important aspect of image understanding because it is a feature of the image that is invariant to transition, rotation, and partially scale changes. Corners are defined as the intersecting points of two connected straight edge lines. Alternatively, they are also known as junctions, where the edges of an image meet at a point with a high degree of angular deviation. The definition of a corner, as a connection of straight edge lines, mathematically refers to the point where two dominant and different gradient orientations exist. Corner detection provides a wealth of information on the neighbourhood of corners. Corners are also unique in local regions compared to edges, making them favoured for wide baseline matching.

The definition of a corner suggests that corners are dependent on scale. Corner detection and multi-scale analysis are straightforward and essential ways to identify points of interest. Reversely, corners can be viewed as points of interest at a fixed scale. There are three classes of

corner detection methods: classical gradient-based, template-based, and contour-based detection. Classical gradient-based corner detection relies on gradient calculation. Template-based detection compares pixels, and in recent years, templates have been combined with machine learning techniques. Contour-based detection, on the other hand, relies on contour and boundary detection results to identify corners [39].

The Harris corner detector [51] proposed in 1988 is probably the most widely used detector for corner detection. It is an eigenvalue-based feature point detector known for its strong invariance to image noise and rotation [52]. The Harris corner detector operates by computing the local auto-correlation function of a signal, which measures the local changes of the signal with patches shifted by a small amount in different directions. The auto-correlation function defined for a given shift $(\Delta x, \Delta y)$ and a point (x, y) is given as

$$E\left(u,v\right)=\sum\nolimits_{x,y} w(x,y)\left[I\left(x+u,y+v\right)-I(x,y)\right]^2, \tag{3.5}$$

where E is the difference between the original and the moved window, u and v refer to the window displacement in the x and y direction, $w(x,y)$ refers to the weighting function of the window at position (x,y), commonly represented by a Gaussian or a window of ones, $I(x+u,y+v)$ is the intensity of the moved window and $I(x,y)$ is the intensity of the original image [53].

By using the Taylor series and expanding the square in Equation 3.5 it can be further rewritten as

$$E(u,v)\approx [u\ v]\left(\sum\nolimits_{x,y} w(x,y)\begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix}\right)\begin{bmatrix} u \\ v \end{bmatrix}. \tag{3.6}$$

By substitution, the equation can be further simplified as

$$E(u,v)\approx [u\ v]H\begin{bmatrix} u \\ v \end{bmatrix}, \tag{3.7}$$

where H represents Harris-Matrix

$$H=\sum\nolimits_{x,y} w(x,y)\begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix}. \tag{3.8}$$

The Harris corner detector then defines a corner response R

$$R=Det\left(H\right)-k\left(Trace\left(H\right)\right)^2=\lambda_1\cdot\lambda_2-k\left(\lambda_1+\lambda_2\right)^2, \tag{3.9}$$

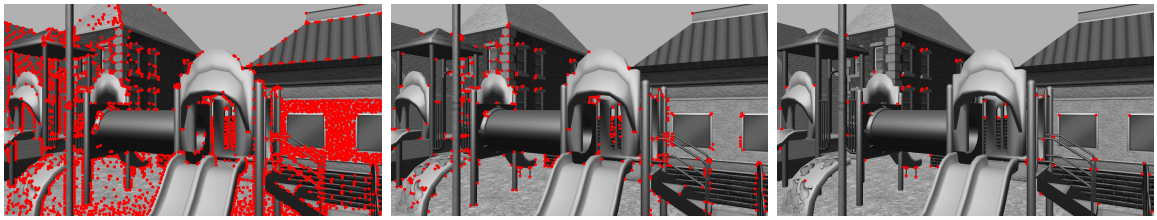
where λ_1 and λ_2 are eigenvalues of H, $Det(H)=\lambda_1\cdot\lambda_2$, $Trace(H)=\lambda_1+\lambda_2$, and k is a constant [54].

The Harris corner detector is favoured for its robustness against image noise and ability to handle rotational transformations. It is suitable for various computer vision applications, for instance, camera calibration, augmented reality, structure from motion or visual simultaneous localization and mapping.

Harris was further improved by Shi and Tomasi in 1994 [55], with their aim to address some limitations of the original detector and provide a more reliable corner detection method. Shi and Tomasi redefined the corner response as minimum eigenvalue

$$R=min(\lambda_1,\lambda_2), \tag{3.10}$$

which produces feature points that are more stable and accurate for tracking than the Harris detector [56]. Additionally, Shi and Tomasi introduced a threshold parameter to distinguish corners from non-corners. Pixels with a corner response value more significant than the threshold are identified as corners. This step also helps to reduce false positives and improves the overall accuracy of corner detection. Figure 3.3 demonstrates the distinctions resulting from various threshold settings.



(a) Threshold = 0.01 (b) Threshold = 0.1 (c) Threshold = 0.33

Figure 3.3: The Shi-Tomasi corner detector with different threshold values

3.3 Keypoint detection

Keypoints or interest points are referred to as local extremes in scale-location spaces, describing surrounding areas, typically circular or square regions. Compared to that, interest regions are referred to as segmented irregular regions with defined constancy. Interest point detection aims to find local extremes in pyramid spaces, and interest region detection aims to identify regions with constancy by segmentation techniques.

Interest points can provide an informative representation of digital images. They refer to local extrema in 3-dimensional scale spaces with locations and scales as axes. Thus, interest points can be mathematically denoted as (x, y, σ), where (x, y) indicate the location and σ indicates the scale. A corner can be viewed as a point of interest at a fixed scale. Furthermore, every point of interest is given a feature descriptor, a compact and informative representation of the local image content surrounding this point. These descriptors can encode relevant information about the intensity, texture, or gradients in the neighbourhood of the interest points. They can be obtained inside square or circular regions centered at (x, y) with the size determined by the scale σ [57]. Various interest point detection methods are proposed, such as Hessian–Laplacian [58], SIFT [59], SURF [52], MSER [60], BRISK [61], and FREAK [62]. Figure 3.4 illustrates the contrast between different sigma values set in the SIFT keypoints detector.

Classical methods including Laplacian of Gaussian (LoG), difference of Gaussian (DoG) and Hessian–Laplacian are based on Gaussian pyramid construction. The scale space of an image is defined as a function $L(x, y, \sigma)$ that is the result of the convolution of a variable-scale Gaussian $G(x, y, \sigma)$ with an image $I(x, y)$ as

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y), \tag{3.11}$$

where $*$ is the convolution operation in x and y, and the Gaussian scale-space kernel is defined as

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} exp^{-\frac{x^2+y^2}{2\sigma^2}}. \tag{3.12}$$

The Gaussian pyramid is then constructed by increasing σ. LoG is based on the Laplacian of Gaussian filtered scale space. Each layer of LoG pyramid is defined as

$$\nabla^2 L = L_{xx} + L_{yy}, \tag{3.13}$$

where L_{xx} and L_{yy} are the second partial derivatives. Unlike LoG, DoG layers are obtained by the difference of two nearby Gaussian smoothed layers, without the computation of the second partial derivatives. DoG can be seen as the approximation of the LoG with low computational cost. DoG function, which can be computed from the difference of two nearby scales separated by a constant multiplicative factor, is defined as

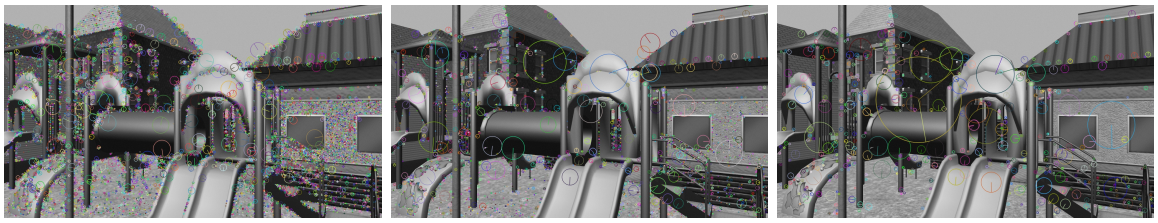
$$D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y), \tag{3.14}$$

where k is a multiplicative factor. The local extremes of LoG and DoG pyramid are recorded as LoG and DoG interest points, respectively.

SIFT (Scale Invariant Feature Transform), proposed by Lowe [59], is one of the most popular feature detectors and descriptors in the literature [63]. SIFT can efficiently identify object points in noisy, cluttered, and occluded environments due to its high invariance to translation, scaling, and rotation. This method extracts interest points from the image in two steps. First, the local extremes in DoG pyramid are recorded as potential keypoints, and a 3D quadratic function is used to approximately locate the interpolated location of candidate keypoints. Second, the measurement function computed with the trace and determinant of the Hessian matrix is used to eliminate keypoints with strong edge responses and sub-pixel localization [39]. The interest points extracted this way show scale invariance and rotation invariance.

In addition, it is possible to compute the descriptor for each detected keypoint. This descriptor captures information about the local image gradient patterns surrounding the detected keypoint, enabling robust keypoint matching described in the following sections.

The descriptor extraction process commences with dividing a localized region centred on the keypoint into smaller, well-defined subregions or bins. Typically, these subregions are organized in a grid, often with divisions like 4x4. Within each of these delineated subregion, histograms of gradient orientations are constructed. These histograms represent the distribution of edge orientations within the subregion. The gradient magnitudes and their corresponding directions are leveraged to populate these histograms (see Figure 3.4, which shows calculated magnitudes and directions for each detected keypoint). All these histograms are concatenated into a single high-dimensional vector, creating the descriptor for the given keypoint. This vector typically consists of hundreds of values, and in essence, it encapsulates the intricate local gradient patterns encompassing the keypoint [64].



(a) $\sigma = 0.5$ (b) $\sigma = 1.6$ (c) $\sigma = 5.0$

Figure 3.4: The SIFT keypoint detector with different sigma values

Chapter 4: 3D feature detection

Contents

Three-dimensional feature detection is a fundamental task that involves identifying and extracting distinct spatial structures in the point clouds acquired, for example, from LiDARs or depth cameras. As in 2D image feature detection, the primary goal is to find characteristic properties representing salient and stable cues in the 3D environment. In the context of 3D data, visual features refer to specific patterns and primitives that differentiate various objects and surfaces in a point cloud. Unlike traditional images, which are composed of pixels in a two-dimensional grid, point clouds capture the spatial coordinates of individual points in a 3D scene. Therefore, 3D feature detection involves identifying points of interest, lines, planes, and higher-level structures within the point cloud. These 3D features are important in various applications, for instance, localization, mapping, and navigation of robots and autonomous vehicles, 3D object recognition, using augmented reality or environmental monitoring and land cover classification.

Detecting features in 3D comes with its own set of challenges. The process must be robust to handle variations in scale, viewpoint, illumination, and quality of the acquired data. Point clouds are often affected by noise, occlusions, and varying point densities, making feature extraction more complex than from 2D images. A high-performance feature detector should show robustness to these problems and satisfy other interests such as efficiency, accuracy, and repeatability. Also, it needs to consider computational efficiency since computational complexity is significantly higher than in image feature detectors.

In the realm of 3D feature detection, this thesis focuses on two fundamental categories: geometric feature detection and gradient-based methods. This is the same approach as in the context image feature detection outlined in Chapter 3. Geometric feature detection primarily involves the extraction of basic geometric primitives, including points, lines, and planes, within the 3D space. These features are frequently associated with object boundaries and salient surface characteristics. Complementing this, gradient-based methods harness gradient information and surface normals to identify features corresponding to object boundaries and distinctive surface attributes in the 3D domain. It is worth noting that within 3D feature detection alternative methodologies exist, including histogram-based [65] and learning-based [66] feature detectors.

The selected methods for 3D point detection align with those chosen for image feature detection from cameras, as described in the previous Chapter 3. Specifically, these methods

include edges, corners, and key points, which were selected due to their clear definition and simple detection in both modalities. Additionally, it is conceivable to leverage plane detectors, commonly employed in 3D data processing. However, using them for cross-modal matching brings many complications.

The application of plane detectors, widely used in the analysis of 3D data, poses challenges when adapted for cross-modal matching. The difficulties come from the diverse characteristics of data captured by different sensors, introducing variations in perspective, resolution, and environmental conditions. Also, the global nature of plane features and the impossibility of precisely locating them at one point makes them harder to detect and match between different types of data reliably. The potential differences in the appearance of planes in images and point clouds, coupled with sensor-specific noise, introduce other problems. Therefore, despite their effectiveness in 3D analysis, using plane detectors for cross-modal matching introduces complexities that may hinder the robustness and generalizability of the matching process. The chosen approach is focused on more localized features, such as edges, corners, and keypoints, offering a more adaptable and reliable approach for achieving effective cross-modal correspondences.

The following section will provide a detailed description of the method for edge detection in 3D space, which corresponds to the edge detection approach described in the preceding chapter. Subsequently, the possibility of applying 2D points of interest detectors to 3D data obtained from LiDAR or depth cameras will be explained. This approach has been chosen due to its significant time and computational efficiencies compared to other alternative 3D methods.

4.1 Edge detection

Edge detection in 3D aims to identify regions in the point cloud with significant changes in surface orientation, indicating transitions between different objects or surfaces. Specifically, this process involves determining points in the point cloud that are situated on sharp discontinuities or boundaries between surfaces. The orientation of the surface at each point in the point cloud is represented by the local surface normal computed at that point. Techniques such as Principal Component Analysis (PCA) [67] or Least Squares Fitting [68] are employed to estimate the surface normal using a neighbourhood of points surrounding each point.

As these techniques can be computationally and temporally demanding, this thesis draws upon the paper of Ji Zhang and Sanjiv Singh [18]. This paper leverages the characteristic of LiDAR or depth camera, which generates the returned points in an ordered manner. Although LiDAR naturally generates unevenly distributed points, the returns from the laser scanner have the same angular resolution within a scan. The feature points are extracted using only information from individual scans, with a co-planar geometric relationship. Consider X_i as a point obtained from LiDAR, and let S be the set of consecutive points of X_i returned by the laser scanner in the same scan. Given that the laser scanner generates point returns in clockwise or counterclockwise order, S encompasses half of its points on each side of X_i, exhibiting uniform angular intervals. To assess the smoothness of the local surface, Zhang

and Singh introduced a term c

$$c = \frac{1}{|S| \cdot \|X_i\|} \left\| \sum_{j \in S, j \neq i} (X_i - X_j) \right\|. \tag{4.1}$$

Subsequently, the points in a scan are sorted based on their respective c values, and the feature points are selected based on the maximum c values and labelled as potential points.

To further enhance accuracy, additional constraints are imposed to optimize the distribution of feature points across the environment and to avoid selecting points that are already surrounded by feature points. This involves dividing the environment into four identical subregions, ensuring the number of selected edge points within each subregion adhere to the specified maximum limit. Another condition is that a point is eligible for selection only if none of its neighbouring points have already been chosen as feature points. Furthermore, to ensure reliable feature point selection, points situated on surface patches that align closely with the laser beam direction or are positioned at the boundary of occluded regions are excluded from consideration. These measures collectively aid in the extraction of robust and accurate edge points. Figure 4.1 illustrates detected edges by this algorithm in data obtained from the depth camera and LiDAR.

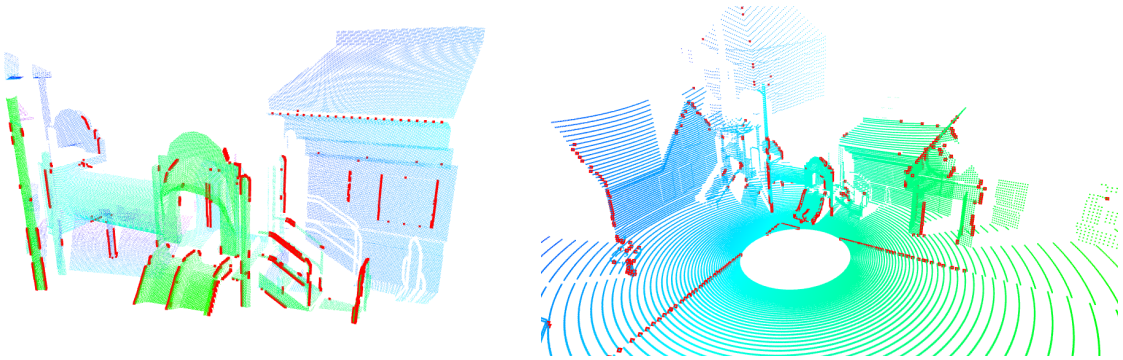


Figure 4.1: Detected edges in data obtained from the depth camera (left) and LiDAR (right)

4.2 Using 2D detectors

Using image (2D) feature detection on a projected image of a point cloud offers several advantages over employing 3D feature detection on the same point cloud. One primary reason is the computational efficiency achieved by reducing data dimensionality. When using 3D feature detection directly on the point cloud, the processing complexity increases substantially due to the large amount of raw 3D data involved. This results in higher computational costs and potentially limits real-time performance in certain applications.

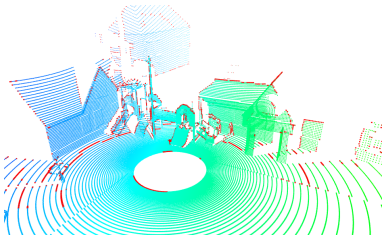
In addition, projecting the point cloud onto a 2D image plane allows us to leverage well-established 2D feature detection algorithms, which are generally computationally more efficient. The projected image presents a condensed representation of the 3D data, reducing the number of data points processed and analyzed. As a result, the computational complexity is significantly reduced, enabling faster and more efficient feature detection.

Furthermore, image feature detection techniques often benefit from a wealth of existing research and development, making them more refined and accurate. Many advanced algorithms for image feature detection have been extensively studied and optimized, leading to robust and reliable methods for identifying salient features in 2D images.

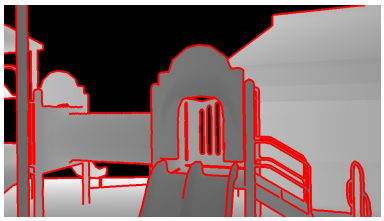
The utilization of 2D feature detection on 3D data has proven to be highly functional, as demonstrated by S. Urban and M. Weinmann [69]. Their approach has found effective applications in scenarios such as the registration of two sets of 3D point clouds acquired through laser scanning, as well as in place recognition using imaging LiDAR [24].

The image feature detectors described in the previous chapter (Chapter 3) will be employed for the detection of points of interest in the 3D data as well. This will be achieved by applying these detectors to the depth image obtained from the respective sensor. Through this process, crucial key points will be identified on the depth image, which will subsequently be projected into 3D space using methods outlined in Chapter 2 or selected as corresponding points within the point cloud from this sensor. Figure 4.2 shows detected edges by the Canny detector on data from a depth camera and LiDAR, on depth images and corresponding point clouds. Figure 4.3 shows the same with the detected corners by the Shi-Tomasi detector and Figure 4.4 highlights keypoints detected by the SIFT detector.

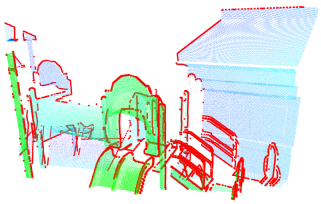
(a) LiDAR depth image



(b) LiDAR point cloud

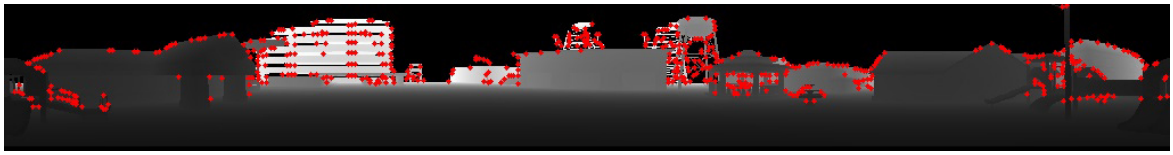


(c) Depth camera image

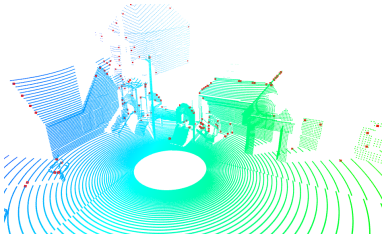


(d) Depth camera point cloud

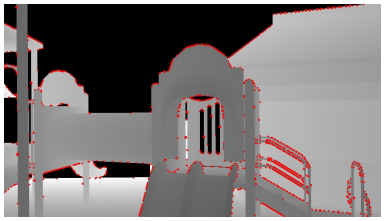
Figure 4.2: Examples of the detected edges using the Canny image detector on data acquired from the depth camera and LiDAR



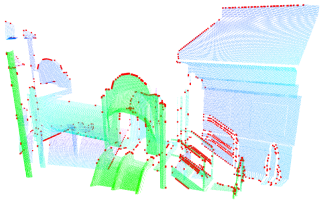
(a) LiDAR depth image



(b) LiDAR point cloud

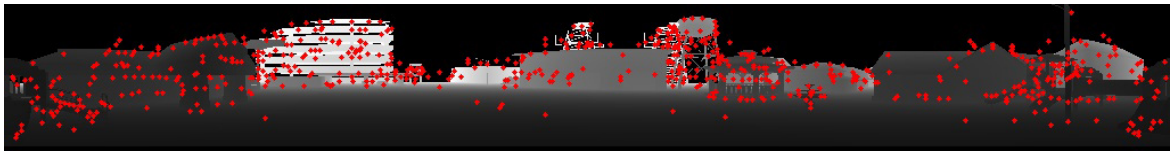


(c) Depth camera image

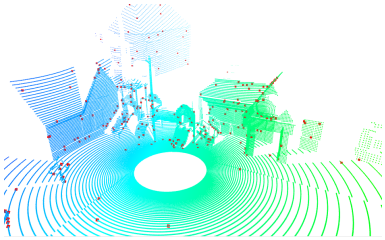


(d) Depth camera point cloud

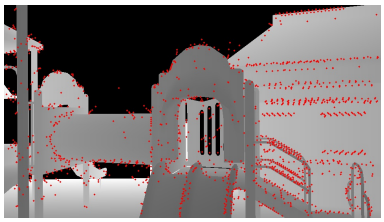
Figure 4.3: Examples of the detected corners using the Shi-Tomasi image detector on data acquired from the depth camera and LiDAR



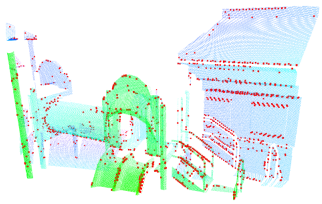
(a) LiDAR depth image



(b) LiDAR point cloud



(c) Depth camera image



(d) Depth camera point cloud

Figure 4.4: Examples of the detected keypoints using the SIFT image detector on data acquired from the depth camera and LiDAR

Chapter 5: Methodology

Contents

This chapter describes the methodology for addressing the challenge of establishing mutual correspondences between 3D data from LiDAR or depth cameras and 2D data from a colour camera. Firstly, it starts with data preparation and the process of detecting keypoints from both sensors. Secondly, the pursuit of mutual correspondences among these keypoints is described. Lastly, it explores the quest for optimal alignment between these datasets, based on the acquired mutual correspondences. The final two steps are described even in the scenario when data from the colour camera can be directly projected into the space, for instance, through a depth camera, thus being treated as 3D points for further alignment requirements. This configuration aligns with the same modality as data that came from LiDAR and depth camera sensors. This approach allows a single modal matching, that facilitates the assessment of whether the chosen methods for detecting keypoints in both types of sensors are adequate for characterizing the given 3D space and can be used for comparison with multimodal matching.

5.1 Feature extraction

Data preparation from the camera involves primarily converting colour images into grayscale, as described by Equation 2.15. This needs to be done because most algorithms for keypoint detection in images operate on a single channel rather than three. Subsequently, the grayscale image undergoes the chosen feature detection algorithm. The keypoints, or point of interest, are then stored as a set of pixel coordinates for further use.

In the case of a depth camera, data preparation involves generating a 3D point cloud through the projection method described in Chapter 2. For LiDAR data, the preparation includes creating a depth image by extracting data from a 2D matrix, as detailed in the same chapter. Following this step, the depth image is normalized to values ranging from 0 to 255 using the following equation:

$$I_{new}(x, y) = \frac{I(x, y) - I_{min}}{I_{max} - I_{min}} \cdot 255, \tag{5.1}$$

where $I_{new}(x, y)$ is new normalized intensity of pixel x, y, $I(x, y)$ is the old value of the same pixel and I_{max} and I_{min} are the maximal and minimal values of all pixels in the image. This normalization is performed to enable the use of 2D keypoint detectors.

In scenarios where depth data are missing in the depth images from one of the 3D sensors, interpolation can be applied. Interpolation employs a straightforward and intuitive nearest-neighbour method, which assigns the value of the nearest data point to the newly interpolated point. While this approach may result in blocky or pixelated images, it is computationally efficient and preserves edge sharpness. This preservation of edge sharpness is a critical characteristic given that many keypoint detection methods rely on image edges. Furthermore, especially with depth cameras in real-world environments, it often happens that data on the edges of objects are lost, as illustrated in Figure 5.1.

(a) Grayscale image from a depth camera (b) Corresponding image from a colour camera

Figure 5.1: An image pair from a depth camera and the corresponding colour camera image. In the left-hand image, completely black pixels indicate areas where the sensor did not capture any information. It is readily apparent that this occurs, for instance, at the edge of the arch and the edges of the staircase, which is also shown in the second image from the colour camera.

Subsequently, one of the selected feature detectors is applied to these data in the form of a 3D point cloud or depth image, as described in Chapter 4. Keypoints are consistently stored as sets of 3D points. For keypoint detection in depth images, this is achieved by identifying corresponding 3D points in space for the detected keypoints in the image, which are subsequently designated as keypoints.

5.2 Finding correspondences

Given that the successful execution of algorithms and methodologies used in the following steps relies on the attainment of a sound initial estimate, it becomes imperative to commence with the transformation of the data acquired from the colour camera into the most recent known coordinates of the second 3D sensor.

The nearest-neighbour algorithm is used to establish correspondences between 3D keypoints derived from the 3D sensor and those obtained from the colour camera projected into the spatial realm through the deployment of a depth camera, as explicated in the previous section of this chapter. Within this procedure, keypoints resident within one of the point clouds undergo scrutiny through comparison with their closest neighbours situated within the other,

herein referred to as the reference point cloud. The technique of choice for this comparative process rests upon the application of k-d trees [70], known for their optimization capabilities in the realm of search operations characterized by reduced computational complexity.

More precisely, a k-d tree, short for k-dimensional tree, is a widely used data structure in computational geometry and data science, especially for spatial partitioning. It proves particularly valuable in organizing and querying high-dimensional data, such as point clouds in three-dimensional space. The k-d tree recursively divides multidimensional space into regions, often using axis-aligned hyperplanes. Each node within the tree represents a distinct space region and stores a k-dimensional point selected from the dataset. These points are strategically arranged to facilitate efficient search operations. The tree's construction ensures an alternating split of points along different coordinate axes, resulting in a balanced binary tree structure. This partitioning strategy enables swift search operations, as entire subtrees can be pruned during the search for the nearest neighbours.

In pursuit of further acceleration prospects, a strategic reduction in the volume of keypoints in the denser areas of the point clouds by random means is a viable approach. An additional enhancement in computational efficiency is introduced via deploying the outlier filter. This filter serves as an instrument for the selective removal of erroneous correspondences among keypoints and thus operates upon keypoints that have already undergone the process of pairing and subsequent connection with corresponding keypoints within the second point cloud. The underlying rationale is the observation that correspondences featuring a smaller magnitude of relative distance are less likely to exhibit characteristics of outlying elements. To this end, the filter initiates a sorting operation among all ascertained correspondences between keypoints, predicated upon their respective distances. Subsequently, the filter removes correspondences within the uppermost 10% quantile. The practical outcome of this intricate procedural sequence is a reduction in the total number of keypoints, which leads to a significant enhancement in the speed of subsequent phases in the processing pipeline.

For the purpose of establishing correspondences between 3D keypoints derived from 3D sensors and 2D keypoints from the colour camera, a dedicated algorithm was developed. This algorithm must address the distinctive nature of data obtained from these disparate sensor modalities, ensuring an effective correlation between 3D spatial information and the corresponding 2D image features. An initial step involves projecting the camera keypoints into rays within 3D space, as described in Section 2.2.3. These rays are defined by two points: a common starting point located at the optical centre of the camera, which remains consistent for all rays originating from a given image, and a second point fixed at a predetermined distance of z-coordinate from the camera. The same z_0-coordinate for all vectors $\bar{\mathbf{d}} = (d_x, d_y, d_z)^T$ normalized using Equation 2.18 is achieved by multiplying all the coordinates of a given vector by a constant k such that $k = z_0/d_z$. Correspondences are then sought by minimizing the distance between these rays from the camera and the 3D keypoints. The equation expressing the distance between the point and the line is used,

$$d(A, B, P) = \frac{|(B - A) \times (A - P)|}{|B - A|}, \tag{5.2}$$

where A and B are the coordinates of the two points defining the line AB, and P is the point for which we calculate the distance from the line.

A correspondence is defined as a pair comprising a 3D point and a ray (or its corresponding 2D point in the colour camera image) if they exhibit the minimum distance among all

possible combinations for the given 3D point and all 2D points. This involves a minimization as described by the equation

$$min_{b \in B}\ d(Ab, P), \tag{5.3}$$

where P is a given 3D point, A is the first point of all rays of 2D points, and B is the set of all second points of rays.

Due to the computationally intensive nature of this process, which necessitates the calculation of pairwise distances between all 3D and 2D keypoints in the form of rays, it was necessary to find a solution for an expeditious retrieval. For this reason, a special algorithm employing k-d trees was devised for this thesis. However, it is important to note that while k-d trees are exceptionally efficient for finding the nearest neighbours to a given point in multidimensional space, they are not well-suited for finding the nearest point to a line segment or a line defined by two points. This limitation arises from the fundamental nature of k-d trees, due to their partition space using axis-aligned hyperplanes. Since lines in space are not constrained to be axis-aligned, the partitioning structure to a k-d tree does not align with the geometry of a line.

Therefore, in the developed algorithm, when searching for the nearest line to a point, a necessary data transformation to a more suitable representation must be done to use the k-d tree data structure. This procedure capitalizes on the uniformity of rays, all of which share the same starting point and have the second point at an equidistant position on the z-axis. This z-axis is consistent with the optical axis, as illustrated in Figure 2.1. All of these rays can subsequently be transformed into a new 2D space using their coordinates defined by two angles orthogonal to the optical axis of the camera (see Figure 5.2). This can be described in the way similar to spherical coordinates, albeit without the coordinate r denoting the distance from the centre. Similarly, 3D keypoints from the sensor can be transformed into these 2D coordinates using the same two angles with respect to the camera's optical axis. Thus, a 3D keypoint is redefined as a vector emanating from the camera. Since all 2D keypoints from the camera lie in a single plane, the relative distances to a given vector remain preserved. It follows that the one that forms the smallest angle with the 3D point vector, or in other words, has the smallest difference between the angles of these vectors, is also the closest to the respective point. The advantage of this transformed 2D space is that a k-d tree structure can be efficiently used to find the nearest point, even though the actual value of the distance found in this 2D space does not provide information about the relationship between the line and the point.

This approach effectively enables the identification of correspondences between a 3D point and a ray, but conversely, it is not feasible. This limitation arises from the fact that 3D keypoints, unlike projected 2D keypoints, do not lie on the same z-coordinate. If the nearest 3D point were sought in the same manner for a projected 2D point, it is conceivable that a more distant 3D point, which actually has a greater Euclidean distance from the given line, may have a smaller angular difference and, therefore, could erroneously be identified as the closer point. This inconsistency arises from the projection process and the absence of the third dimension (z-coordinate) in the transformed 2D space. This is depicted in Figure 5.3.

An alternative approach harnesses SIFT descriptors, which are derived from the image regions surrounding keypoints. In this context, the spatial coordinates of keypoints do not contribute to the correspondence detection process; instead, only their descriptors play a pivotal role in the matching procedure. Keypoint matching, essentially identifying corresponding

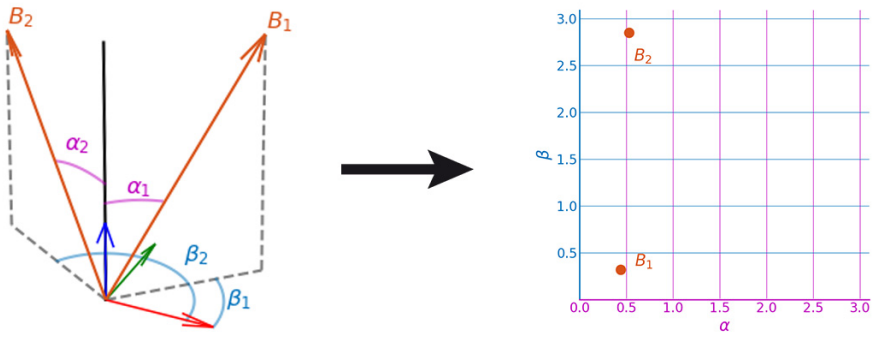
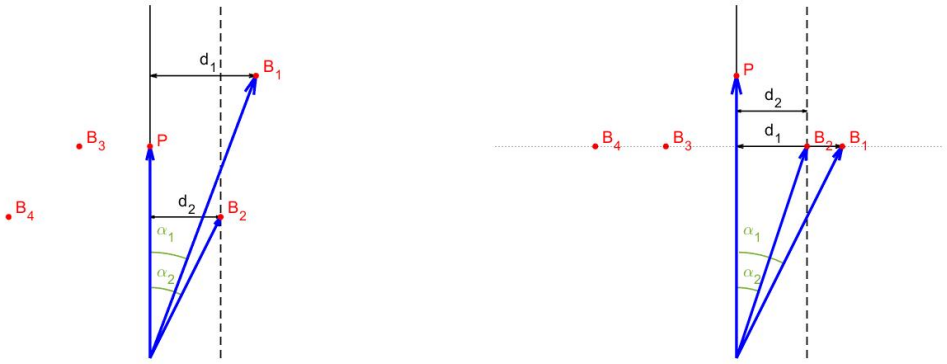


Figure 5.2: The illustration of the transformation of two 3D points, B_1 and B_2, into 2D space defined by two angles. In the left-hand image, both points are depicted as vectors, with angles α_1 and β_1 describing point B_1, and α_2 and β_2 for point B_2. These angles indicate the deviation from the optical axis of the camera. The right-hand image portrays the same points in 2D space, represented by identical angles α and β.



(a) Point-to-line (b) Line-to-point

Figure 5.3: Comparison of Line-to-Point and Point-to-Line approaches. The left-hand illustration shows that the angle α_1 for point B_1 is smaller than the angle α_2 for point B_2. Consequently, point B_1 will be identified as the closer point, even though the distance d_1 to the line defined by point P is greater than the distance d_2 from point B_2 to the same line. In the right-hand illustration, when all points lie in a single plane, point B_2 will be correctly identified as closer because its angle α_2 is smaller than angle α_1.

keypoints across images, hinges on assessing the similarity among these high-dimensional descriptor vectors. This task is achieved by performing the nearest neighbour search in the descriptor space.

To quantify the dissimilarity between descriptor vectors, the Euclidean distance metric is employed. For each descriptor in one image, the most akin descriptor, as indicated by the smallest Euclidean distance, within the set of keypoints from the other image is identified.

Nevertheless, the direct comparison of raw descriptor vectors can be sensitive to variations caused by lighting, perspective, and other factors. Consequently, post-processing steps,

including thresholding, are frequently applied to alleviate this sensitivity. Only matches with distances below the specified threshold are deemed valid during this stage, while others are filtered out. A ratio test may also be implemented, considering the two closest matches for each descriptor in one image. A match is considered valid if the ratio between the distances of the best match and the second-best match distances exceeds a predefined threshold. These measures aim to fortify the resilience of keypoint matching by diminishing sensitivity to specific variations while ensuring the retention of only dependable correspondences.

However, despite all these enhancements of post-processing steps, the practical application of these descriptors for matching keypoints identified by the SIFT detector proved to be impractical. This was observed even in tests conducted on paring SIFT points in images from a colour camera and a depth camera. The keypoints, and especially their descriptors characterizing their surrounding environment, exhibited significant disparities, likely deriving from the differences in modalities, even though both images may appear similar to the human eye. This resulted in entirely incorrect keypoint pairings, as illustrated in Figure 5.4. Consequently, SIFT keypoints will be treated similarly to other keypoints, and their cross-modal pairing will rely on a developed algorithm involving the search for correspondences between rays and points, as detailed earlier in this Section.

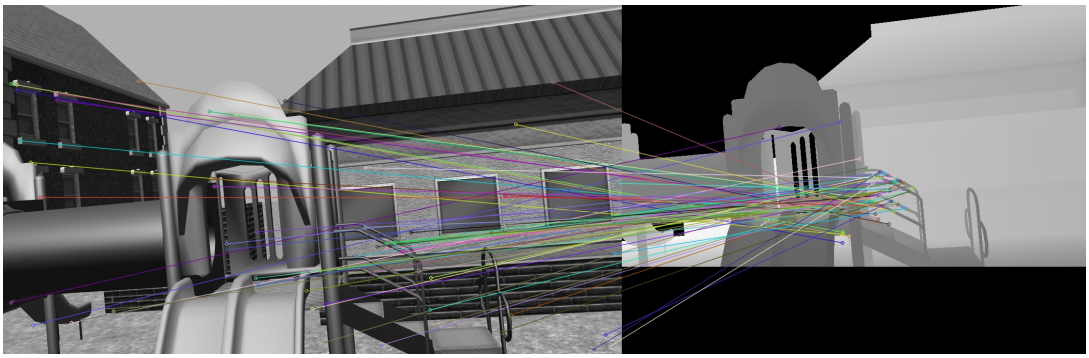


Figure 5.4: The illustration of incorrect keypoint paring using SIFT descriptors between data from a colour camera and a depth camera

5.3 Cross-modal alignment

After identifying mutual correspondences between keypoints from the 3D sensor and 3D keypoints from the colour camera, the Iterative Closest Point (ICP) algorithm is applied to obtain the mutual transformation. This algorithm is a widely used technique in the field of computer vision and robotics for aligning and registering two sets of 3D data points. ICP operates iteratively to refine the alignment between the source (or moving) point cloud and the target (or reference) point cloud. In each iteration, it establishes correspondences, as described in the previous section, between points in the source cloud and the target cloud.

Once correspondences are established, ICP computes the transformation that best aligns the source cloud with the target cloud. This transformation is a rigid transformation encompassing translation and rotation. The optimal transformation is computed by utilizing Singular Value Decomposition (SVD) on the covariance matrix of the paired points. SVD

helps decompose this matrix into three fundamental transformations: translation, rotation, and scaling [71].

The algorithm iteratively refines this transformation, converging towards an alignment that minimizes the misalignment between the source and target clouds. The iterations continue until a predefined termination condition is met, which could be a maximum number of iterations, a threshold on the change in alignment error, or other criteria. As the iterations progress, the source cloud progressively aligns with the target cloud, resulting in an accurate transformation that minimizes the misalignment between the two point sets.

After establishing mutual correspondences between 3D keypoints from the sensor and 2D keypoints from the camera, the developed algorithm performs a reprojection of these points into space in the form of rays emanating from the camera. For each correspondence between a 3D keypoint and a ray, the point on the ray that is closest to the corresponding 3D point is identified. This point is determined by the following equation

$$P_{closest} = Au(B - A), \tag{5.4}$$

where $P_{closest}$ is the closest point on the line segment, A and B are the two points defining the line segment and u is the parameter that determines the position of $P_{closest}$ on the line segment and can be calculated as

$$u = \frac{(P - A)(B - A)}{\|B - A\|^2}, \tag{5.5}$$

where P is the given 3D keypoint.

A new point cloud is generated from these obtained points on the rays, which serves as input for the ICP algorithm described above. The key difference compared to the standard ICP is that for each iteration of the algorithm, it is necessary to rediscover correspondences between 3D points and rays and again find the nearest point to the given 3D point on the rays.

Chapter 6: Experimental verification

Contents

This chapter presents the validation and verification of the proposed methodology, along with the testing and comparison of selected methods for keypoint detection in images and 3D point clouds. The performance and reliability of the chosen localization methods are systematically assessed through realistic simulations. The utilization of simulations is imperative for several reasons: it provides a controlled environment where ground truth data can be precisely defined, enabling a meticulous evaluation of algorithmic performance. Moreover, simulations offer a focused environment for the exclusive comparison of the performance of proposed methods without the introduction of noise factors, such as motion-induced blur or erroneous 3D sensor reflections. These elements would necessitate additional algorithms for filtering, which would be beyond the scope of this thesis. The methods described in Chapter 3 and Chapter 4 undergo cross-validation, considering both the colour camera and both 3D sensors — LiDAR and depth camera, such that each method used to detect points of interest in the image is validated with every method used to detect points of interest in the data from 3D sensors. Furthermore, this multimodal matching is compared with the outcomes of unimodal matching of 3D point clouds using the same methods for detecting keypoints.

The following sections provide a detailed description of the evaluation metrics used to assess localization accuracy. An analysis of simulation results follows, comparing individual methods and their errors with respect to ground truth. Table 6.1, Table 6.2 and Table 6.3 summarize the parameters of the selected keypoint detection methods and the proposed procedure used for testing.

6.1 Evaluation metric

To systematically evaluate the algorithm described above, an analytical tool for evaluating localization estimates is introduced. A quantitative analysis is conducted in reference to ground truth pose data obtained during simulations. Established trajectory evaluation metrics used in visual or inertial odometry, including Root Mean Square Error (RMSE) and Absolute Pose Error (APE), are applied. The most straightforward quality assessment of a

	Specification	Symbol	Value
ICP	Termination criteria		
	Max iterations	N	50
	MSE translation threshold	ϵ_{trans}	0.01 m
	MSE rotation threshold	ϵ_{rot}	0.001 rad
	Smooth Length	-	4
	Trimmed Distance Outlier filter ratio	-	0.35

Table 6.1: General parameters of the proposed methodology presented in Chapter 5

	Specification	Symbol	Value
Canny edge detector	Low Threshold	T_{low}	70
	Threshold Ratio	-	3
	Kernel Size	k	3
Shi-Tomasi corner detector	Max Corners	N	600
	Quality Level	R_{min}	0.01
	Min Distance	d	5 pixels
	Block Size	k	5
	Gradient Size	-	3
SIFT keypoint detector	Number of Octave Layers	O	3
	Contrast Threshold	C	0.04
	Edge Threshold	E	10
	Sigma	σ	1.6

Table 6.2: Parameters of the selected methods for image keypoint detection from colour camera for simulation presented in Chapter 3

localization history is determined by the Root Mean Square Error, mathematically defined as

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} \|\mathbf{p}_i - \hat{\mathbf{p}}_i\|^2}{N}}, \tag{6.1}$$

where N represents the number of pose samples, $\mathbf{p}_i$ denotes the ground truth pose, and $\hat{\mathbf{p}}_i$ signifies the corresponding estimated pose. RMSE provides a measure of the overall accuracy of the estimated trajectory by quantifying the root of the mean squared differences between the estimated and ground truth poses. Lower RMSE values indicate higher accuracy and closer alignment with the true trajectory.

The Absolute Pose Error provides a measure of the accuracy of the estimated pose

	Specification	Symbol	Value
Canny edge detector	Low Threshold	T_{low}	10
	Threshold Ratio	-	3
	Kernel Size	k	3
Shi-Tomasi corner detector	Max Corners	N	5000
	Quality Level	R_{min}	0.001
	Min Distance	d	2 pixels
	Block Size	k	5
	Gradient Size	-	3
SIFT keypoint detector	Number of Octave Layers	O	3
	Contrast Threshold	C	0.001
	Edge Threshold	E	100
	Sigma	σ	1.1
Edge detector	Min Curvature	c	0.01

Table 6.3: Parameters of the selected methods for keypoint detection at data from 3D sensors for simulation presented in Chapter 3 and Chapter 4

compared to the ground truth pose. For a specific pose i it is defined as

$$APE_i = \arccos\left(\frac{\text{trace}(\mathbf{R}_i^\top \hat{\mathbf{R}}_i) - 1}{2}\right) + \|\mathbf{t}_i - \hat{\mathbf{t}}_i\|, \tag{6.2}$$

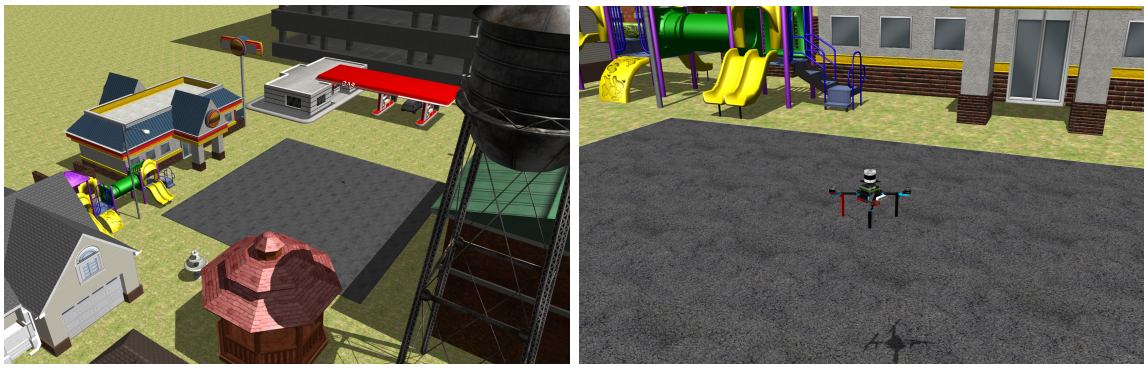
where $\mathbf{R}_i$ and $\mathbf{t}_i$ represent the ground truth rotation matrix and translation vector for pose i, while $\hat{\mathbf{R}}_i$ and $\hat{\mathbf{t}}_i$ represent the corresponding estimated rotation matrix and translation vector. APE values indicate the algorithm's ability to accurately estimate both the rotational and translational aspects of the pose.

6.2 Simulation

The Gazebo[1] [72] simulator is employed for its capabilities to simulate a realistic environment. In the simulator, identical unmanned aerial vehicle (UAV) sensors, including sensor noise, are simulated in the same way as in real robots. However, this simulation does not introduce physical challenges for sensors, such as issues with reflections on object edges detected by the depth camera or LiDAR, as illustrated in Figure 5.1. These realistic problems would need to be addressed using separate algorithms, potentially impacting the comparison of the proposed methods. Furthermore, the simulation environment provides ground truth data used for the evaluation of the proposed method, which is hard to obtain in the real world. Hence, to manifest the reality as closely as possible, an urban environment with diverse structures is used in the simulation, as shown in Figure 6.1. A UAV flight was executed and recorded in

[1]Gazebo, `https://gazebosim.org/`

this environment, capturing raw sensor outputs and ground truth localization in each frame. The trajectory of this flight is shown in Figure 6.2. The AUV was equipped with a colour camera, two 3D sensors, a depth camera, and LiDAR at the same time to compare results on both types of sensors. Subsequently, the same recording was employed for the cross-validation of all selected methods, ensuring uniform testing conditions.



(a) An overview of the urban environment map depicting distinct structures

(b) A close-up view of the UAV conducting the test flight for data collection

Figure 6.1: An illustration of the simulation environment employed for data collection for testing purposes

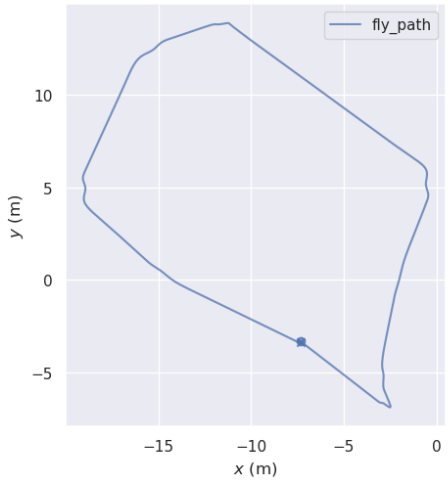


Figure 6.2: Top view of the ground truth trajectory of recorded UAV flight in the simulated environment

The record of the UAV flight was divided into individual frames in the time when the data from sensors were synchronized, and the absolute position was simultaneously recorded. For each frame i, keypoints are detected in the data from the colour camera using a selected method. For the previous frame $i-1$, keypoints are also detected in the data from the 3D sensor using another selected method. After transforming the keypoints from the colour camera in frame i into rays, following the algorithm described in Section 5.2, pairing is performed with the keypoints from the 3D sensors detected in frame $i-1$, as described in the same Section. From these corresponding pairs of points, the projected 3D keypoint cloud from

the colour camera is aligned with the point cloud of 3D keypoints detected from the depth camera or LiDAR using the ICP algorithm. The obtained transformation from ICP is then compared with the transformation between frame i and frame $i-1$, obtained from ground truth localization.

The cross-validation results for individual methods using a colour camera and LiDAR or a depth camera, described by RMSE, are presented in Table 6.4 and Table 6.5. From these data, it is evident that the best result for cross-modal matching between a colour camera and LiDAR involves matching keypoints from the SIFT detector and the Shi-Tomasi corner detector. Similarly, for matching data between a colour camera and a depth camera, the metrics show the best results for both pairs SIFT and SIFT detectors and Shi-Tomasi and SIFT detectors. Notably, except for an exception, matching keypoints from the same detector on both sensors appears significantly worse.

Colour camera data / LiDAR data	Canny edge detector	Shi-Tomasi corner detector	SIFT keypoint detector
Canny edge detector	52.22	9.99	4.15
Shi-Tomasi corner detector	0.51	1.24	**0.41**
SIFT keypoint detector	3.53	0.63	0.78
Edge detector	2.81	0.53	0.65

Table 6.4: RMSE values in meters corresponding to multimodal keypoint matching for each pair of selected detectors at the colour camera and LiDAR data

Colour camera data / Depth camera data	Canny edge detector	Shi-Tomasi corner detector	SIFT keypoint detector
Canny edge detector	0.85	0.68	0.73
Shi-Tomasi corner detector	2.12	0.64	0.59
SIFT keypoint detector	0.70	**0.48**	**0.48**
Edge detector	1.03	0.79	0.65

Table 6.5: RMSE values in meters corresponding to multimodal keypoint matching for each pair of selected detectors at the colour and depth camera data

Figure 6.3 illustrates the APE evolution in individual frames of a given UAV flight for all methods between the camera and LiDAR. This fact is further detailed in Figure 6.4, which displays the maximum, average, and median of APE and RMSE for all pairs of methods. This figure shows that utilizing keypoint matching with SIFT and Shi-Tomasi detectors yields the best results in all observed parameters.

A similar scenario is depicted in Figure 6.5, illustrating the APE evolution in individual frames for all pairs of methods between the depth camera and the colour camera. Figure 6.6 provides a detailed view of the maximum, average, and median of APE, along with RMSE for all methods for these two sensors. This figure similarly shows that regarding the RMSE parameter, the pairs of detectors SIFT – SIFT and Shi-Tomasi – SIFT yield the best results.

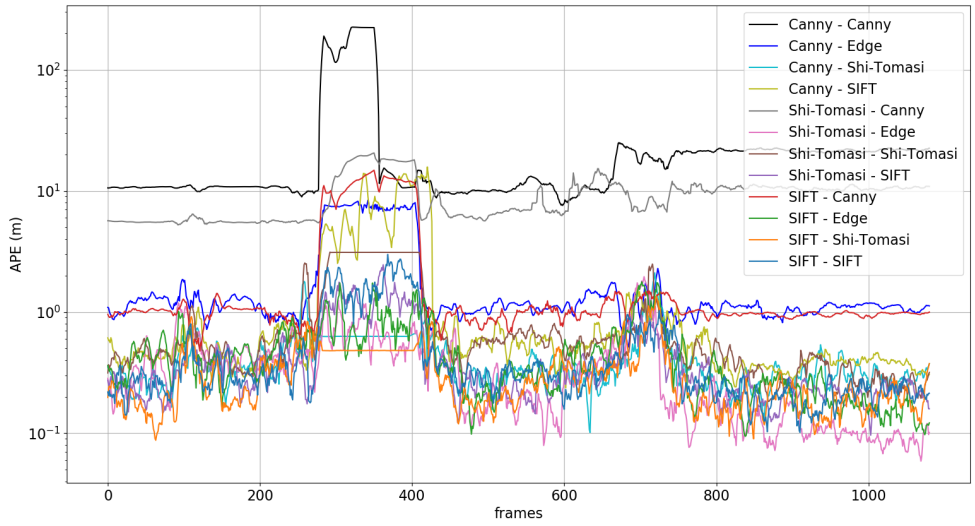


Figure 6.3: Comparison of the absolute pose error to the ground truth reference during the entire UAV flight of all pairs of methods for feature detection. First method named in each pairs is used on colour camera data and second on LiDAR data.

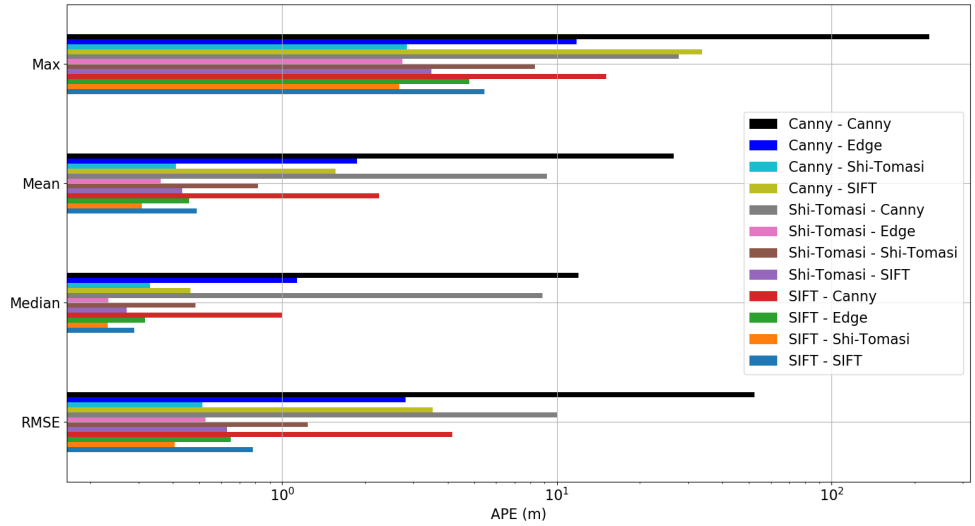


Figure 6.4: Summary of maximal, mean and median values of APE and RMSE of all pairs of methods for feature detection during UAV flight in the simulation world. First method named in each pairs is used on colour camera data and second on LiDAR data.

However, in other observed parameters, except for the maximum of APE, the pair of Shi-Tomasi and SIFT detectors demonstrate better performance.

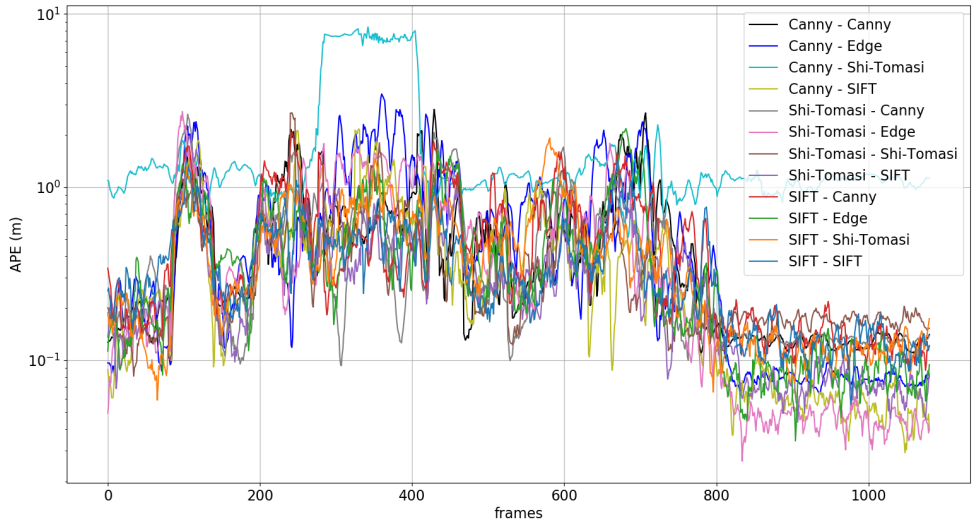


Figure 6.5: Comparison of the absolute pose error to the ground truth reference during the entire UAV flight of all pairs of methods for feature detection. First method named in each pairs is used on colour camera data and second on depth camera data.

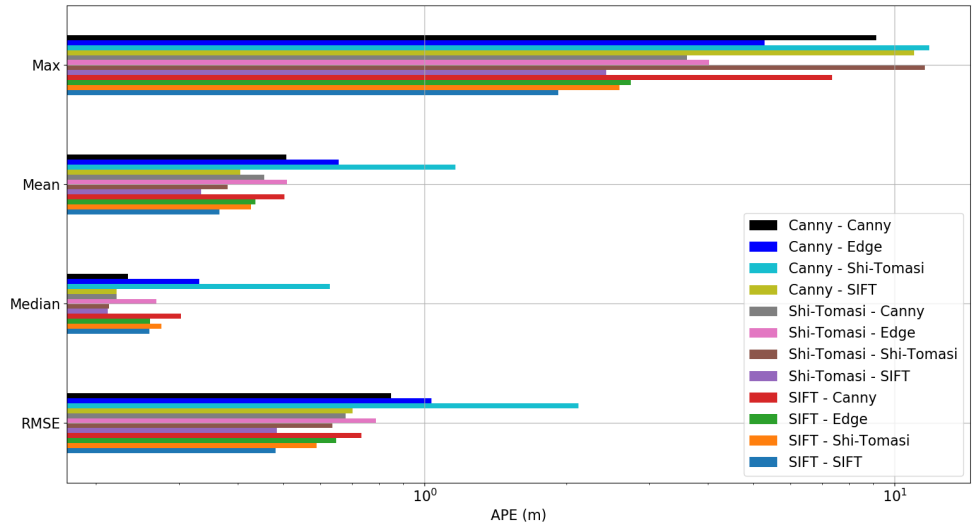


Figure 6.6: Summary of maximal, mean and median values of APE and RMSE of all pairs of methods for feature detection during UAV flight in the simulation world. First method named in each pairs is used on colour camera data and second on depth camera data.

6.3 Comparison of cross and single-modal matching

For the comparison between multimodal matching and single-modality matching, the algorithm outlined in Chapter 5 is used. This algorithm for each frame i detects keypoints using selected detectors in the colour camera data. Subsequently, these detected pixels in the colour image are directly re-projected into space using a depth camera with the exact origin coordinates in the same frame i. Using this process, image keypoints acquire 3D coordinates in the space. This obtained 3D point cloud is then further utilized for uni-modal matching with keypoints detected in the data from the 3D sensor in frame $i-1$ using other selected detectors. Uni-modal matching uses the nearest neighbour search in three dimensions and the ICP algorithm to align the cloud from the colour camera with the point cloud from the depth camera or LiDAR. Thus, the obtained transformation from ICP is compared with the transformation between frame i and frame $i-1$, obtained from ground truth localization.

This approach facilitates a comprehensive comparison between multimodal and single-modal registration using the same types of detectors. Table 6.6 provides a comprehensive overview of RMSE values for all pairs of selected detectors used with both LiDAR and depth camera sensors, for both multimodal and single-modal matching. From this table, it is evident that for the majority of pairs of keypoint detectors, single-modal registration from 3D to 3D yields better results, particularly when using the depth camera sensor, which is unsurprising. However, for some pairs of detectors, the use of multimodal registration yields better results. Notably, the pair consisting of the SIFT detector on colour camera data and the Shi-Tomasi corner detector on LiDAR data not only exhibits the best results for this pair of detectors but also achieves the best outcome for multimodal matching with this test dataset.

Colour camera feature detector	3D sensor feature detector	multimodal matching		unimodal matching	
		Depthcamera	LiDAR	Depthcamera	LiDAR
Canny	Canny	0.85	52.22	**0.38**	1.35
Canny	Edge	**1.03**	2.81	1.79	1.91
Canny	Shi-Tomasi	2.12	**0.51**	1.54	2.79
Canny	SIFT	0.70	3.53	**0.48**	1.59
Shi-Tomasi	Canny	**0.68**	9.99	0.80	1.80
Shi-Tomasi	Edge	0.79	0.53	**0.46**	2.57
Shi-Tomasi	Shi-Tomasi	**0.64**	1.24	1.21	3.30
Shi-Tomasi	SIFT	0.48	0.63	**0.36**	2.07
SIFT	Canny	0.73	4.15	**0.43**	1.79
SIFT	Edge	0.65	0.65	**0.26**	3.45
SIFT	Shi-Tomasi	0.59	**0.41**	0.84	3.28
SIFT	SIFT	0.48	0.78	**0.44**	2.98

Table 6.6: Summary of RMSE values in meters for pairs of methods for detecting features on colour camera data and data from both 3D sensors compared in multimodal (2D image and 3D point cloud) and unimodal (3D and 3D point clouds) matching during simulation.

6.4 Discussion of results

Matching single and multi-modal sets involves a compromise between specific advantages and disadvantages, primarily depending on the application. Certainly, simplicity and computational efficiency can be highlighted among the advantages of unimodal matching. In cases where the use of multiple modalities is redundant, matching one modality can provide satisfactory results. Such matching also leverages data consistency, and algorithms adapted to a single modality can be simpler. For instance, in the context of matching 3D modalities in the form of point clouds, direct application of the ICP algorithm without the need for feature detection can be effective. Similarly, for matching 2D colour images, the SIFT detector and descriptor can be used directly without further data transformation. However, no exact unimodal matching was used in the testing described above. This is because feature detection still occurred in two different modalities, but only their matching happened within one modality – the point cloud. Utilizing unimodal matching with detection in the same modality would likely yield significantly better results.

Unimodal approaches naturally lack the complex information available from multiple sources. This can result in poorer performance, especially in scenarios where various information is crucial for precise matches. Unimodal pairing can be more susceptible to problems specific to that modality, such as changes in lighting, noise, or changes in the viewing angle for camera data or long straight tunnels for 3D sensor data. Additionally, for the purposes of precise localization, for both a map and a robot that needs to be localized, the same modality needs to be used, which can be financially expensive in some cases for specific sensors.

Compared to that, the combination of information from multiple modalities often increases the robustness of comparison algorithms. By using additional data, the system becomes more resilient to challenges present in individual modalities. Multimodal matching can provide a richer representation of the scene, leveraging the strengths of each modality. This increased discriminative power is particularly valuable in complex environments or scenarios with ambiguous unimodal data or when precise localization in a map created by a modality different from the robots' sensors is needed. The data presented above demonstrates that this multimodal matching facilitates, at the least, basic orientation capabilities. This is contingent on the choice of an appropriate pair.

In the testing data above, the combination of the Shi-Tomasi detector, adept at identifying corners, and the SIFT detector, despite lacking precise geometric interpretation, appeared most effective. The robust performance of the Shi-Tomasi detector is unsurprising, given the well-defined nature of corners in both images and point clouds. Moreover, their sparse distribution minimizes the likelihood of confusion in the proposed point and ray matching algorithm. Conversely, the combination and good performance of the SIFT might be intuitively unexpected due to the non-geometric representation of the detected keypoints. However, David G. Lowe and his team [21] demonstrated that SIFT can serve as effective natural landmarks for tracking and localization.

The main disadvantages of fusing information from multiple modalities often include the almost unavoidable increase in computational complexity. Integrating data from different sources requires sophisticated algorithms and can be computationally demanding, potentially limiting real-time applications. This was evident during the testing described above, where some selected pairs would require significant computational optimization to approach real-

time use. Combining data from multiple modalities requires careful integration strategies. Therefore, designing a unique algorithm for matching rays and points and creating a new interpretation of ray data by transforming it into 2D space for the purpose of accelerated search using a kd-tree data structure was necessary. Another disadvantage is certainly the imprecision achieved by the proposed procedure. The proposed method could likely be used for rapid odometry in control in case of a GNSS signal outage, but certainly not for precise localization. For this further improvement and optimization of the proposed approach would be necessary.

In conclusion, the choice between single and multi-modality matching depends on the specific requirements of the application. While single-modality approaches offer simplicity and efficiency, multimodal methods provide enhanced robustness and a more comprehensive understanding of the environment.

Chapter 7: Conclusion

Contents

This thesis addressed the challenge of multimodal alignment, focusing on 2D data from a colour camera and 3D data from LiDAR or depth camera. The motivation was introduced, outlining the issues surrounding the use of multiple modalities together with related work and their limitations. Subsequently, keypoint detectors for both 2D and 3D data were selected and presented, along with the possibility of employing 2D detectors on data from 3D sensors. These detectors were chosen to cover fundamental areas of computer vision and to logically describe features in the selected modalities—2D and 3D space. Following this, the developed algorithm for cross-modal keypoint matching using the nearest neighbour search was described. This included a specialized modification of the k-d tree for significant acceleration of the process. Subsequently, an algorithm for mutual alignment of modalities using an iterative approach, aiming to align the pairs of identified keypoints and points, was introduced. This proposed procedure was tested cross-modally on all pairs of selected keypoint detectors in a simulation environment. In the context of the experimental evaluation, the best performance was shown by the pair of the Shi-Tomasi corner detector on colour camera data and the SIFT detector on depth camera data, achieving the final RMSE of 0.45 m. When considering the fusion of data from the colour camera and LiDAR sensors, the combination of SIFT and Shi-Tomasi detectors appeared to be the optimal choice, reaching the RMSE of 0.41 m. Remarkably, this multimodal matching outperforms unimodal matching utilizing the same detectors. This was presented in the last section, where these results of multimodal alignment were compared with single-modal 3D-to-3D alignment.

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

- Chapter 3 reviews and compares algorithms for detecting 2D features in RGB images and Chapter 4 for 3D features in scans from 3D LiDARs or depth cameras.
- The review of algorithms for associating features among the two modalities and summarization of how deeply the topic is studied in the literature was presented in Section 1.3.
- The selection of the feature-detection methods most suitable for cross-modality association and argumentation was presented for image data in Chapter 3 and 3D data in Chapter 4.

- A process of an optimization task, maximizing the alignment between two cross-modal feature sets with selected feature-detected methods, was developed and implemented in Chapter 5.
- The evaluation of the matching performance for selected pairs of feature detectors was presented in Chapter 6. The same chapter also defines advantages and disadvantages in contrast to matching single-modality sets of 3D space.

7.1 Future Work

The introduced algorithm for multimodal matching in localization could be further optimized, for instance, by employing Linear Kalman Filters (LKF). LKF is a recursive algorithm that updates the current state based on previous and current observations. Optimization involves minimizing the mean squared error of the system state, potentially aiding in refining the accuracy of position estimates. Another improvement could be finding optimal parameters for a specific pair of detectors instead of using general parameters tested across all pairs.

An alternative option is exploring different methods for finding mutual pairs in distinct modalities, such as using neural networks. However, this option requires creating a substantial dataset first. Deploying the presented algorithm to address the entire problem, not just its test version, would also entail addressing additional issues related to the model of the entire map. This includes challenges like seeing through walls, where handling points theoretically projectable into the camera's image plane are actually behind an opaque barrier that needs attention. To adapt and test the algorithm in the real world, constraints and uncertainties associated with real sensors, such as the poor response of sensor rays at object edges, as described in Section 5.1, must be addressed.

Bibliography

[1] Y. Feng, L. Fan, and Y. Wu, "Fast Localization in Large-Scale Environments Using Supervised Indexing of Binary Features," *IEEE transactions on image processing*, vol. 25, no. 1, pp. 343–358, 2016.

[2] C. Thorpe, M. Hebert, T. Kanade, and S. Shafer, "Vision and navigation for the Carnegie-Mellon Navlab," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 3, pp. 362–373, 1988.

[3] N. Snavely, S. M. Seitz, and R. Szeliski, "Photo tourism: exploring photo collections in 3D," in *ACM SIGGRAPH 2006 Papers on - SIGGRAPH '06*. ACM Press, 2006, p. 835.

[4] J. L. Schonberger and J.-M. Frahm, "Structure-from-Motion Revisited," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2016, pp. 4104–4113.

[5] T. Qin, P. Li, and S. Shen, "VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[6] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[7] G. Pandey, J. McBride, S. Savarese, and R. Eustice, "Automatic Targetless Extrinsic Calibration of a 3D Lidar and Camera by Maximizing Mutual Information," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 26, no. 1, pp. 2053–2059, 2021.

[8] S. Mishra, G. Pandey, and S. Saripalli, "Extrinsic Calibration of a 3D-LIDAR and a Camera," in *2020 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2020, pp. 1765–1770.

[9] L. Zhou, Z. Li, and M. Kaess, "Automatic Extrinsic Calibration of a Camera and a 3D LiDAR Using Line and Plane Correspondences," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 5562–5569.

[10] S. A. Rodriguez F., V. Fremont, and P. Bonnifait, "Extrinsic calibration between a multi-layer lidar and a camera," in *2008 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*. IEEE, 2008, pp. 214–219.

[11] J. Domhof, J. F. Kooij, and D. M. Gavrila, "An Extrinsic Calibration Tool for Radar, Camera and Lidar," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8107–8113.

[12] T. Fish, "Google Maps: How to get Street View on Google Maps," 2020, accessed on October 16, 2023. [Online]. Available: https://www.express.co.uk/life-style/science-technology/1270478/google-maps-how-to-get-street-view-google-maps-desktop-phone

[13] M. Zaffar, "Visual Place Recognition for Autonomous Robots," Ph.D. dissertation, 09 2020.

[14] M. Bednarek, P. Kicki, and K. Walas, "On Robustness of Multi-Modal Fusion—Robotics Perspective," *Electronics*, vol. 9, no. 7, p. 1152, 2020.

[15] A. Cherubini, R. Passama, P. Fraisse, and A. Crosnier, "A unified multimodal control framework for human–robot interaction," *Robotics and Autonomous Systems*, vol. 70, pp. 106–115, 2015.

[16] D. Grajales, S. Kadoury, R. Shams, M. Barkati, G. Delouya, D. Béliveau-Nadeau, B. Nicolas, W. T. Le, M.-K. Benhacene-Boudam, D. Juneau, J. N. DaSilva, J.-F. Carrier, G. Hautvast, and C. Ménard, "Performance of an integrated multimodality image guidance and dose-planning system supporting tumor-targeted HDR brachytherapy for prostate cancer," *Radiotherapy and Oncology*, vol. 166, pp. 154–161, 2022.

[17] E. C. Robinson, K. Garcia, M. F. Glasser, Z. Chen, T. S. Coalson, A. Makropoulos, J. Bozek, R. Wright, A. Schuh, M. Webster, J. Hutter, A. Price, L. Cordero Grande, E. Hughes, N. Tusor, P. V. Bayly, D. C. Van Essen, S. M. Smith, A. D. Edwards, J. Hajnal, M. Jenkinson, B. Glocker, and D. Rueckert, "Multimodal surface matching with higher-order smoothness constraints," *NeuroImage*, vol. 167, pp. 453–465, 2018.

[18] J. Zhang and S. Singh, "LOAM: Lidar Odometry and Mapping in Real-time," in *Robotics: Science and Systems X*. Robotics: Science and Systems Foundation, 2014. [Online]. Available: http://www.roboticsproceedings.org/rss10/p07.pdf

[19] A. Nüchter, K. Lingemann, J. Hertzberg, and H. Surmann, "6D SLAM—3D mapping outdoor environments," *Journal of Field Robotics*, vol. 24, no. 8-9, pp. 699–722, 2007.

[20] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct EKF-based approach," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 298–304.

[21] S. Se, D. Lowe, and J. Little, "Vision-based global localization and mapping for mobile robots," *IEEE Transactions on Robotics*, vol. 21, no. 3, pp. 364–375, 2005.

[22] S. Khattak, C. Papachristos, and K. Alexis, "Keyframe-based Direct Thermal–Inertial Odometry," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 3563–3569.

[23] X. Wang and H. Zhang, "Good Image Features for Bearing-only SLAM," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006, pp. 2576–2581.

[24] T. Shan, B. Englot, F. Duarte, C. Ratti, and D. Rus, "Robust Place Recognition using an Imaging Lidar," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 5469–5475.

[25] N. Khedekar, M. Kulkarni, and K. Alexis, "MIMOSA: A Multi-Modal SLAM Framework for Resilient Autonomy against Sensor Degradation," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 7153–7159.

[26] M. Feng, S. Hu, M. H. Ang, and G. H. Lee, "2D3D-Matchnet: Learning To Match Keypoints Across 2D Image And 3D Point Cloud," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 4790–4796.

[27] 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, pp. 155–166, 2009.

[28] B. Wang, C. Chen, Z. Cui, J. Qin, C. X. Lu, Z. Yu, P. Zhao, Z. Dong, F. Zhu, N. Trigoni, and A. Markham, "P2-Net: Joint Description and Detection of Local Features for Pixel and Point Matching," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 16 004–16 013.

[29] A. Mishchuk, D. Mishkin, F. Radenovic, and J. Matas, "Working hard to know your neighbor's margins: Local descriptor learning loss," *Advances in neural information processing systems*, vol. 30, 2017.

[30] J. Li and G. H. Lee, "DeepI2P: Image-to-Point Cloud Registration via Deep Classification," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 15 960–15 969.

[31] S. Ren, Y. Zeng, J. Hou, and X. Chen, "CorrI2P: Deep Image-to-Point Cloud Registration via Dense Correspondence," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 33, no. 3, pp. 1198–1208, 2023.

[32] Y. Jeon and S.-W. Seo, "EFGHNet: A Versatile Image-to-Point Cloud Registration Network for Extreme Outdoor Environment," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7511–7517, 2022.

[33] G. Wang, Y. Zheng, Y. Guo, Z. Liu, Y. Zhu, W. Burgard, and H. Wang, "End-to-end 2D-3D Registration between Image and LiDAR Point Cloud for Vehicle Localization," *arXiv preprint arXiv:2306.11346*, 2023.

[34] R. W. Wolcott and R. M. Eustice, "Visual localization within LIDAR maps for automated urban driving," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 176–183.

[35] G. Pascoe, W. Maddern, and P. Newman, "Direct Visual Localisation and Calibration for Road Vehicles in Changing City Environments," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*, 2015.

[36] T. Caselitz, B. Steder, M. Ruhnke, and W. Burgard, "Monocular camera localization in 3d lidar maps," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 1926–1931.

[37] P. Stancelova, E. Sikudova, and Z. Cernekova, "3D Feature Detector-Descriptor Pair Evaluation on Point Clouds," in *2020 28th European Signal Processing Conference (EUSIPCO)*, 2021, pp. 590–594.

[38] S. Filipe and L. A. Alexandre, "A comparative evaluation of 3D keypoint detectors in a RGB-D Object Dataset," in *2014 International Conference on Computer Vision Theory and Applications (VISAPP)*, vol. 1, 2014, pp. 476–483.

[39] Y. Li, S. Wang, Q. Tian, and X. Ding, "A survey of recent advances in visual feature detection," *Neurocomputing*, vol. 149, pp. 736–751, 2015.

[40] F. K. Noble, "Comparison of OpenCV's feature detectors and feature matchers," in *2016 23rd International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*. IEEE, 2016, pp. 1–6.

[41] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, 2004. [Online]. Available: https://www.cambridge.org/core/product/identifier/9780511811685/type/book

[42] 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

[43] K. Sadekar. Understanding Lens Distortion. Learn OpenCV. Accessed on September 18, 2023. [Online]. Available: https://learnopencv.com/understanding-lens-distortion/

[44] R. Koch and J. Bruenger, *Depth Estimation*. Springer International Publishing, 2021, pp. 290–294.

[45] M. Wertheimer, *Experimentelle Studien Uber das Sehen von Bewegung*. J.A. Barth, 1912.

[46] A. B. Rosenfeld, "Computer vision: a source of models for biological visual processes?" *IEEE Transactions on Biomedical Engineering*, 1989.

[47] P. Dhankhar and N. Sahu, "A review and research of edge detection techniques for image segmentation," *International Journal of Computer Science and Mobile Computing*, vol. 2, no. 7, pp. 86–92, 2013.

[48] J. Canny, "A Computational Approach to Edge Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, 1986.

[49] Z. Musoromy, S. Ramalingam, and N. Bekooy, "Edge detection comparison for license plate detection," in *2010 11th International Conference on Control Automation Robotics & Vision*. IEEE, 2010, pp. 1133–1138.

[50] A. Pooja and K. P. Mahesh, "A Review on Edge Detection Technique," *International Journal of Advanced Engineering Research and Science*, vol. 2, no. 4, pp. 48–53, 2015.

[51] C. Harris and M. Stephens, "A Combined Corner and Edge Detector," in *Procedings of the Alvey Vision Conference 1988*. Alvey Vision Club, 1988. [Online]. Available: http://www.bmva.org/bmvc/1988/avc-88-023.html

[52] H. Bay, T. Tuytelaars, and L. Van Gool, *SURF: Speeded Up Robust Features*. Springer Berlin Heidelberg, 2006, vol. 3951, pp. 404–417. [Online]. Available: http://link.springer.com/10.1007/11744023_32

[53] M. Babiker, O. O. Khalifa, K. K. Htike, A. Hassan, and M. Zaharadeen, "Harris corner detector and blob analysis featuers in human activty recognetion," in *2017 IEEE 4th International Conference on Smart Instrumentation, Measurement and Application (ICSIMA)*, 2017, pp. 1–5.

[54] R. J. Mstafa, Y. M. Younis, H. I. Hussein, and M. Atto, "A New Video Steganography Scheme Based on Shi-Tomasi Corner Detector," *IEEE Access*, vol. 8, pp. 161 825–161 837, 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9186292/

[55] J. Shi and Tomasi, "Good features to track," in *1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 1994, pp. 593–600.

[56] B. Přibyl, A. Chalmers, and P. Zemčík, "Feature point detection under extreme lighting conditions," in *Proceedings of the 28th Spring Conference on Computer Graphics*. ACM, 2012, pp. 143–150.

[57] C. Ding, J. Choi, D. Tao, and L. S. Davis, "Multi-Directional Multi-Level Dual-Cross Patterns for Robust Face Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 3, pp. 518–531, 2016.

[58] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool, "A comparison of affine region detectors," *International Journal of Computer Vision*, vol. 65, no. 1-2, pp. 43–72, 2005.

[59] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[60] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," *Image and Vision Computing*, vol. 22, no. 10, pp. 761–767, 2004.

[61] S. Leutenegger, M. Chli, and R. Y. Siegwart, "BRISK: Binary Robust invariant scalable keypoints," in *2011 International Conference on Computer Vision*, 2011, pp. 2548–2555.

[62] A. Alahi, R. Ortiz, and P. Vandergheynst, "FREAK: Fast Retina Keypoint," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 510–517.

[63] D. Mukherjee, Q. M. Jonathan Wu, and G. Wang, "A comparative experimental study of image feature detectors and descriptors," *Machine Vision and Applications*, vol. 26, no. 4, pp. 443–466, 2015.

[64] J. Ma, X. Jiang, A. Fan, J. Jiang, and J. Yan, "Image Matching from Handcrafted to Deep Features: A Survey," *International Journal of Computer Vision*, vol. 129, no. 1, pp. 23–79, 2021.

[65] R. B. Rusu, N. Blodow, Z. C. Marton, and M. Beetz, "Aligning point cloud views using persistent feature histograms," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008, pp. 3384–3391.

[66] G. Zamanakos, L. Tsochatzidis, A. Amanatiadis, and I. Pratikakis, "A comprehensive survey of LIDAR-based 3D object detection methods with deep learning for autonomous driving," *Computers & Graphics*, vol. 99, pp. 153–181, 2021.

[67] S. Gumhold, X. Wang, R. S. MacLeod *et al.*, "Feature Extraction From Point Clouds," in *IMR*, 2001, pp. 293–305.

[68] S. Fleishman, D. Cohen-Or, and C. T. Silva, "Robust moving least-squares fitting with sharp features," *ACM Transactions on Graphics*, vol. 24, no. 3, pp. 544–552, 2005.

[69] S. Urban and M. Weinmann, "Finding a good feature detector-descriptor combination for the 2d keypoint-based registration of tls point clouds," *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. II-3/W5, pp. 121–128, 2015.

[70] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Communications of the ACM*, vol. 18, no. 9, pp. 509–517, 1975.

[71] P. 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, 1992.

[72] 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. IEEE, 2004, pp. 2149–2154.

Appendices

List of abbreviations

Table 1 lists abbreviations used in this thesis.

Abbreviation	Meaning
LiDAR	Light Detection and Ranging
AR	Augmented Reality
GNSS	Global Navigation Satellite System
SLAM	Simultaneous localization and mapping
SfM	Structure-from-Motion
MI	Mutual Information
MIMOSA	Multi-Modal SLAM
SIFT	Scale Invariant Feature Transform algorithm
ISS	Intrinsic Shape Signatures
EPnP	Efficient Perspective-n-Point
RANSAC	Random sample consensus
DoF	Degrees of Freedom
GPU	Graphics Processing Unit
RGB	Additive colour model
TOF	time-of-flight
LoG	Laplacian of Gaussian
DoG	difference of Gaussian
PCA	Principal Component Analysis
ICP	Iterative Closest Point algorithm
SVD	Singular Value Decomposition
RMSE	Root Mean Square Error
APE	Absolute Pose Error
UAV	Unmanned Aerial Vehicle
LKF	Linear Kalman Filters

Table 1: Lists of abbreviations