TUM SCHOOL OF COMPUTATION, INFORMATION, AND TECHNOLOGY
TECHNISCHE UNIVERSITÄT MÜNCHEN

Cooperative SLAM for Multi-Robot Systems using Visual Odometry and Range Measurements

Young-Hee Lee

Vollständiger Abdruck der von der TUM School of Computation, Information, and Technology der Technischen Universität München zur Erlangung des akademischen Grades einer Doktorin der Ingenieurwissenschaften genehmigten Dissertation.

Die Dissertation wurde am 13.04.2022 bei der Technischen Universität München eingereicht und durch die TUM School of Computation, Information, and Technology am 18.04.2023 angenommen.

Abstract

A swarm of robots can carry out missions much faster and with increased robustness to isolated failures than a single robot. Thus, they can be useful for various tasks, such as exploring extraterrestrial areas and monitoring construction sites. To successfully carry out those missions, the robots need to accurately estimate their positions. Camera-based Visual Simultaneous Localization and Mapping (VSLAM) is a useful tool to achieve this goal in environments where the Global Navigation Satellite System (GNSS) is unreliable or entirely unavailable.

In VSLAM, each robot simultaneously estimates its egomotion and maps the environment using the onboard camera. To merge individual maps and estimate relative poses (positions and orientation) between the robots, the inter-agent loop closing technique is widely applied. This method can significantly reduce the relative positioning error as well as create a global map, using feature points commonly observed by different agents (inter-agent loops). However, a large amount of data needs to be exchanged to detect an inter-agent loop. In addition, substantial computing power is required to fuse multiple local maps. Moreover, loop measurements are only available when different robots observe the same scenes, which significantly constrains the robots' movement and mission planning.

This dissertation proposes collaborative SLAM using visual odometry and range measurements (VOR-SLAM). It employs agent-to-agent and agent-to-anchor ranges to mitigate positioning and mapping errors. By excluding inter-agent loop measurements from the framework, VOR-SLAM requires much less processing power and fewer communication capabilities than inter-agent loop closures. Moreover, the robots can move without significant constraints since ranges can still be obtained when the robots are distributed widely and observe different scenes.

In this dissertation, VOR-SLAM is first introduced using a single-agent setup. It is compared with the state-of-the-art VSLAM using loop closures in terms of positioning accuracy and computational requirement. Moreover, VOR-SLAM is tested with experimental data obtained using a rover equipped with an onboard camera and an Ultra-WideBand (UWB) ranging sensor. Collaborative VOR-SLAM (CoVOR-SLAM) is also evaluated using a multi-agent setup in indoor and outdoor scenarios. The evaluation results show that the robots can efficiently localize themselves even when the computing power and communication network are limited, e.g. deep sea and extraterrestrial mission.

Keywords: Visual SLAM, Visual odometry, Cooperative SLAM, Collaborative SLAM, Multi-robot system, Swarm robotics, Sensor fusion, Camera, Ranging, Ultra-wideband (UWB)

Acronyms and Notation

Acronyms

BA	Bundle Adjustment
BoW	Bag of Words
BRIEF	Binary Robust Independent Elementary Features
BRISK	Binary Robust Invariant Scalable Keypoints
CDF	Cumulative Distribution Function
DLT	Direct Linear Transformation
DoF	Degrees of Freedom
EKF	Extended Kalman Filter
ENU	East North Up
FAST	Features from Accelerated Segment Test
GBA	Global Bundle Adjustment
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
LB	Lower Bound
LoS	Line of Sight
LSE	Least Squares Estimation
MAP	Maximum A Posteriori
NLoS	Non-Line of Sight
ORB	Oriented FAST and Rotated BRIEF
PDF	Probability Density Function
PTAM	Parallel Tracking and Mapping
RANSAC	RANdom SAmple Consensus

RMSE	Root Mean Square Error
RTK	Real-Time Kinematic
SAM	Smoothing and Mapping
SIFT	Scale-Invariant Feature Transform
SLAM	Simultaneous Localization and Mapping
SNR	Signal-to-Noise Ratio
SoO	Signals of Opportunity
SSD	Sum of Squared Difference
SURF	Speeded Up Robust Features
SVD	Singular Value Decomposition
SVO	Semi-dense Visual Odometry
ToF	Time of Flight
UAV	Unmanned Aerial Vehicle
UB	Upper Bound
UWB	Ultra-Wide Band
VINS	Visual-INertial System
VO	Visual Odometry
VOR-SLAM	SLAM with visual odometry and range measurements
VSLAM	Visual Simultaneous Localization and Mapping

Notation

$\mathbf{R}^t_{\mathrm{AB}}$	3×3 rotation matrix from B to A in SO(3)
${}_{\mathrm{C}}\mathbf{t}^t_{\mathrm{AB}}$	3×1 translation vector from A to B defined in the C frame
$\mathbf{t}^t_{\mathrm{AB}}$	3×1 translation vector from A to B defined in the A frame
${}_{\mathrm{C}}\mathbf{p}^t_{\mathrm{AB}}$	3×1 position vector from A to B defined in the C frame
$\mathbf{p}^t_{\mathrm{AB}}$	3×1 position vector from A to B defined in the A frame
$\mathbf{T}^t_{\mathrm{AB}}$	4×4 6DoF pose matrix in SE(3)
$\mathbf{S}^t_{\mathrm{AB}}$	4×4 7DoF similarity matrix in Sim(3)
$\mathbf{X}^i_{\mathrm{L}}$	3×1 Euclidean coordinates of the i-th map point defined in the frame L
$\tilde{\mathbf{X}}^i_{\mathrm{L}}$	4×1 homogeneous coordinates of the i-th map point defined in the frame L
$\boldsymbol{\omega}$	Twist coordinate of so(3)
$\boldsymbol{\nu}$	Twist coordinate of the translation vector

$\boldsymbol{\xi}$	Twist coordinate of a Lie algebra
$\mathbf{G}_i$	Lie group generators
$\mathbf{M}^{+}$	Pseudo-inverse of the matrix $\mathbf{M}$
$[\mathbf{v}]_X$	Skew matrix of the vector $\mathbf{v}$
$\boldsymbol{\Theta}$	A set of unknowns
$\hat{\boldsymbol{\Theta}}$	Optimal estimates of the unknowns

Contents

1. Introduction

1.1. Research Motivation, Objectives, and Main Contributions

A camera-based Visual Simultaneous Localization and Mapping (VSLAM) is widely used to estimate the robots' poses (positions and orientation) when they carry out given tasks in an area without prior mapping or reliable connections to the Global Navigation Satellite System (GNSS). The Mars Exploration Rover (MER) missions with the Curiosity rover of NASA's Mars Science Laboratory could be one of the most well-known applications of VSLAM. VSLAM can be used for swarm robotic systems to conduct a mission faster and with greater resistance to system failures. The project VaMEx-CoSMiC [Sand et al., 2013] of the German Aerospace Center (DLR) proposes an autonomous swarm robotic system with VSLAM to explore the Valles Marineris canyon on Mars as shown in Fig. 1.1.

To avoid collisions between robots and to merge local maps into a global map, robots need to accurately estimate the relative poses. Loop closures [Zou et al., 2019] can be used to achieve this goal employing the map points commonly observed by multiple robots (inter-agent loop). Since this requires sizable processing power, a centralized system architecture has been proposed. In those studies, all the data required for computationally demanding tasks, such as inter-agent loop detection and global map fusion, is transmitted to a central server computer [Schmuck and Chli, 2017, Karrer et al., 2018, Schmuck and Chli, 2019] or to a cloud server [Riazuelo et al., 2014, Mohanarajah et al., 2015], and then the server processes the tasks, not robots' onboard computers. With this approach, each robot only needs to estimate its egomotion and local map, but the robots' movement is restricted as they need to stay around the server to receive up-to-date information.

Decentralized collaborative VSLAM is proposed in [Cunningham et al., 2010, Cunningham et al., 2013, Cieslewski et al., 2018] to have a more flexible swarm architecture, independent on the server connections. In a decentralized system, each robot's onboard computer runs some parts of the inter-agent loop closures, and sends smaller data to the main robot. Then, the main

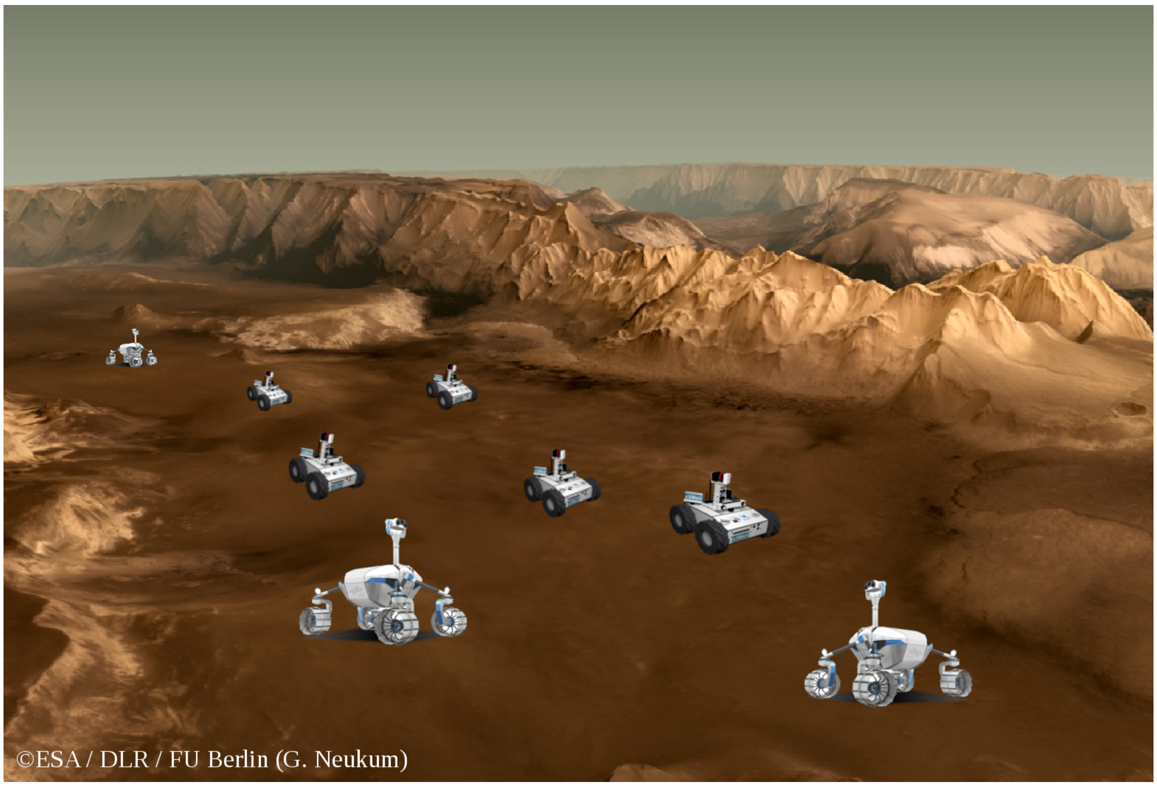


Figure 1.1.: Mars exploration mission using a swarm rover system (image: [Zhang et al., 2020])

robot of the group conducts the map fusion using the databases transmitted from the robots in its group. However, decentralized VSLAM might be still not feasible in environments where computation and communication capabilities are highly limited, e.g. when robots explore an extraterrestrial area without reliable computing power and communication channels. Moreover, robots' movement is still constrained because they need to stay close to each other to obtain inter-agent loops.

This dissertation proposes a SLAM approach using visual odometry and range measurements (VOR-SLAM). With this method, the estimation error can be substantially reduced, without requiring large computing power and adequate communication capabilities. In addition, VOR-SLAM does not require additional complex infrastructure because range measurements can be acquired employing available signals in various environments [Nikookar and Oonincx, 2016]. For example, ranges between robots can be obtained using time of flight measurements with inter-agent communication channels. Agent-to-anchor ranges can be obtained from cellular networks in urban areas as shown in Fig. 1.2. Additionally, low-cost ranging sensors, such as Ultra-WideBand (UWB) ranging modules, are already available.

The followings are the main contributions of this research:

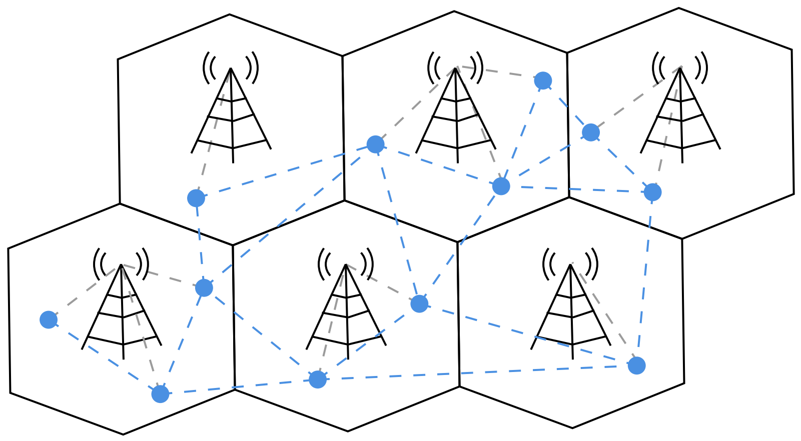


Figure 1.2.: Urban application of cooperative VOR-SLAM using a cellular network and inter-agent communication channels

1. Collaborative localization and mapping with fewer constraints on the mission planning than other approaches:

Range measurements can be obtained using time of flight measurements even when robots are widely distributed in different areas and cannot observe the same map points. Hence, VOR-SLAM has fewer constraints on the mission planning, compared to collaborative VSLAM using inter-agent loop closures. VOR-SLAM can be particularly useful when different types of platforms need to conduct a mission cooperatively. For example, when a rover and a drone carry out a task together as a swarm robotic system, a loop between two local maps might not easily be detected because the two robots observe the scenes from different angles, whereas range measurements can still be easily obtained between two robots using inter-agent communication signals.

2. Few computational and communication requirements:

Collaborative VOR-SLAM requires few computational and communication capabilities because visual information is not involved in the data fusion and optimization processes. Therefore, VOR-SLAM is useful when robots carry out a mission with limited processing power and a weak communication network, e.g. extraterrestrial areas as shown in Fig. 1.1. In addition, the system architecture can be chosen flexibly for each mission (centralized or decentralized) since no powerful server is required to process VOR-SLAM. Moreover, many more robots can be involved in a swarm system compared to VSLAM using loop closures, which enables faster and more efficient mission operations.

3. Accurate cooperative localization and mapping without complex additional infrastructure:

Range measurements can be obtained without complex infrastructure. For example, inter-agent communication channels can be used to obtain range measurements between robots, and the signals of cellular networks can be used to acquire ranges between robots and a base station (anchor point) in urban areas, as shown in Fig. 1.2.

4. General applications in the 3D world and intensive system evaluations using real experimental data

In [Zhu, 2019], a SLAM method similar to VOR-SLAM was proposed for 2D scenarios. For more general applications in the 3D world, this dissertation proposes VOR-SLAM, by improving the 2D approach. Moreover, 2D SLAM in [Zhu, 2019] was evaluated only using synthetic data, while VOR-SLAM is more intensively tested using both synthetic data and real experimental data obtained in two different outdoor environments, using an onboard camera and UWB ranging sensors.

1.2. Thesis Structure

In Chapter 2, the state-of-the-art monocular visual SLAM is reviewed. This chapter introduces the methods to extract feature points in images, as well as map point association and map initialization strategies. Then, the graph-based pose tracking and the local map correction methods (visual odometry process) are explained. In addition, this chapter introduces the loop closing technique which is the most effective method to mitigate the positioning and mapping errors without adding a new sensor to the system. At the end of this chapter, the inherent problems of monocular visual SLAM are discussed along with current solutions.

Chapter 3 introduces VOR-SLAM using the most simple system setup, i.e. a single agent and an anchor point. First, the system overview and setup are explained, followed by the details of the graph-based data fusion method. Then, VOR-SLAM is analyzed using real images of public datasets and range measurements synthetically generated. Moreover, VOR-SLAM is tested using real experimental data (both images and ranges) acquired in two outdoor environments, using a camera and UWB ranging module mounted on a rover.

In Chapter 4, the state-of-the-art collaborative VSLAM are introduced, focused on the approaches using inter-agent loop closures. First, this chapter explains the methods to efficiently search the common map points observed by different robots (inter-agent place recognition), as well as the multi-agent map fusion and optimization methods. In addition, the algorithms are analyzed in terms of computational and communication requirements.

Chapter 5 proposes a novel cooperative VOR-SLAM for swarm robotic systems. First, the system setup and measurement models are explained. Then, the data fusion of multiple robots' visual odometry and range measurements is detailed. Cooperative VOR-SLAM is evaluated employing a four-agent setup in indoor and outdoor application scenarios. Finally, conclusions are drawn in Chapter 6.

2. VSLAM: Visual Simultaneous Localization and Mapping - A Review

Robots can use Visual Simultaneous Localization and Mapping (VSLAM) to estimate their positions and orientation when exploring areas where the Global Navigation Satellite System (GNSS) is unreliable or unavailable. The basic principle of VSLAM is shown in Fig. 2.1. The same map point is projected at different locations in images because a camera moves to a different location or changes its orientation. VSLAM simultaneously estimates the pose between two cameras and the map point coordinates, using this difference of the feature point locations in consecutive images.

The overview of VSLAM is visualized in Fig. 2.2. As reviewed in [Cadena et al., 2016], the processes of VSLAM can be sorted into two groups: front-end and back-end. In the front-end, useful measurements are extracted from raw images, and then converted to the format that can be directly exploited to estimate the camera poses and reconstruct the 3D scenes in the back-end. Direct VSLAM approaches, such as DTAM [Newcombe et al., 2011] and LSD-SLAM [Engel et al., 2014], extracts the photometric intensity from images, which is a multi-dimensional matrix that provides the channel information of images. Using direct VSLAM, the 3D scenes are represented with the depth of each pixel. Instead of using all the pixels' intensity, feature-based VSLAM methods exploit only differentiable feature points (e.g. corner points) as the visual information. Mono-SLAM [Davison et al., 2007], Parallel Tracking and Mapping (PTAM) [Klein and Murray, 2007], and ORB-SLAM [Mur-Artal et al., 2015, Mur-Artal and Tardós, 2017a] are well known feature-based SLAM methods. Using feature-based VSLAM methods, environments are sparsely reconstructed as 3D coordinates associated with each feature point. In addition, semi-direct methods [Forster et al., 2014, Engel et al., 2017] use the intensity of only sparse pixel points to execute VSLAM in real-time without requiring GPU level computing power.

After extracting visual information from raw images, the measurements are examined to identify pixels in consecutive 2D images that are projected from the same 3D object. This is

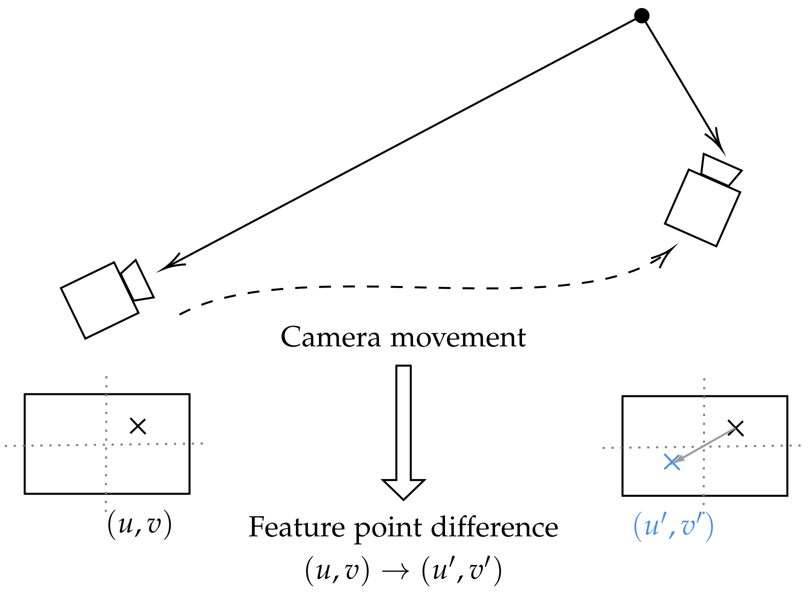


Figure 2.1.: The difference of feature point locations resulting from the camera motion

called place recognition or data association. In direct methods, the intensity matrices of all pixels are compared, whereas only the descriptors of sparse feature points are compared in feature-based methods.

The associated visual measurements are then used in the back-end to estimate camera poses and the structure of environments. Filter-based methods, such as Mono-SLAM [Davison et al., 2007], estimates the current camera pose and the map points observed in the current frame as probability distributions with means and covariance matrices. All the previous camera frames are marginalized in filter-based methods. Additionally, keyframe-based approaches using Bundle Adjustment (BA) [Triggs et al., 1999] can be used to estimate camera poses and map points. This method only marginalizes some parts of the image frames using sliding windowing [Mouragnon et al., 2006, Nistér et al., 2004] or a factor graph. The image frames remaining in the sliding window or graph are called keyframes. For example, ORB-SLAM [Mur-Artal et al., 2015, Mur-Artal and Tardós, 2017a] and iSAM [Kaess et al., 2012, Dellaert et al., 2017] are the state-of-the-art keyframe-based back-end of VSLAM.

VOR-SLAM employs a sparse feature-based method in the front-end since it is a SLAM algorithm for robots with limited computing power. In addition, a keyframe- and graph-based bundle adjustment method is used in the back-end to estimate camera poses and 3D structures, as this method can achieve better accuracy than filtering-based methods [Strasdat et al., 2012]. Thus, the following sections focus on the basic steps of monocular sparse feature- and graph-based SLAM using keyframes.

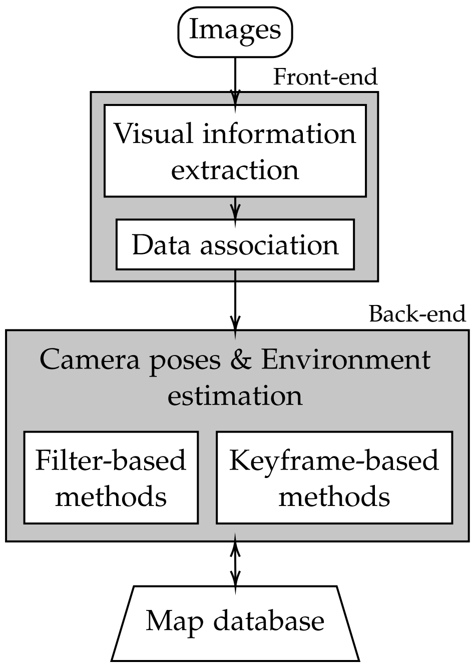


Figure 2.2.: An overview of visual SLAM

2.1. Feature Extraction and Data Association

In the front-end, feature points are first extracted from raw images. Feature points are the pixels at which the gradient of intensity is high in (almost) all directions, i.e. the intensity differs significantly in the region surrounding the feature point. For example, the corners of objects can be extracted as feature points as shown in Fig. 2.3.

The weighted Sum of Squared Difference (SSD) is used to examine pixels:

$$E(\Delta\mathbf{x}) = \sum_i w(\mathbf{x}_i)(I(\mathbf{x}_i + \Delta\mathbf{x}) - I(\mathbf{x}_i))^2, \tag{2.1}$$

where $I(\mathbf{x}_i)$ is the intensity at the pixel point $\mathbf{x}_i$ and $\Delta\mathbf{x}$ is a shift (change) to a neighboring pixel. The window function $w(\mathbf{x}_i)$ limits the search area. It returns 1 when the pixel $\mathbf{x}_i$ is within the area of interest, but 0 for the rest of the pixels.

Applying the Taylor expansion, Eq. (2.1) can be expanded to

$$\begin{aligned} E(\Delta\mathbf{x}) &\approx \Delta\mathbf{x}^T \left\{ \sum_i w(\mathbf{x}_i)\nabla I(\mathbf{x}_i)^T \nabla I(\mathbf{x}_i) \right\} \Delta\mathbf{x} \\ &= \Delta\mathbf{x}^T \mathbf{M} \Delta\mathbf{x}. \end{aligned} \tag{2.2}$$

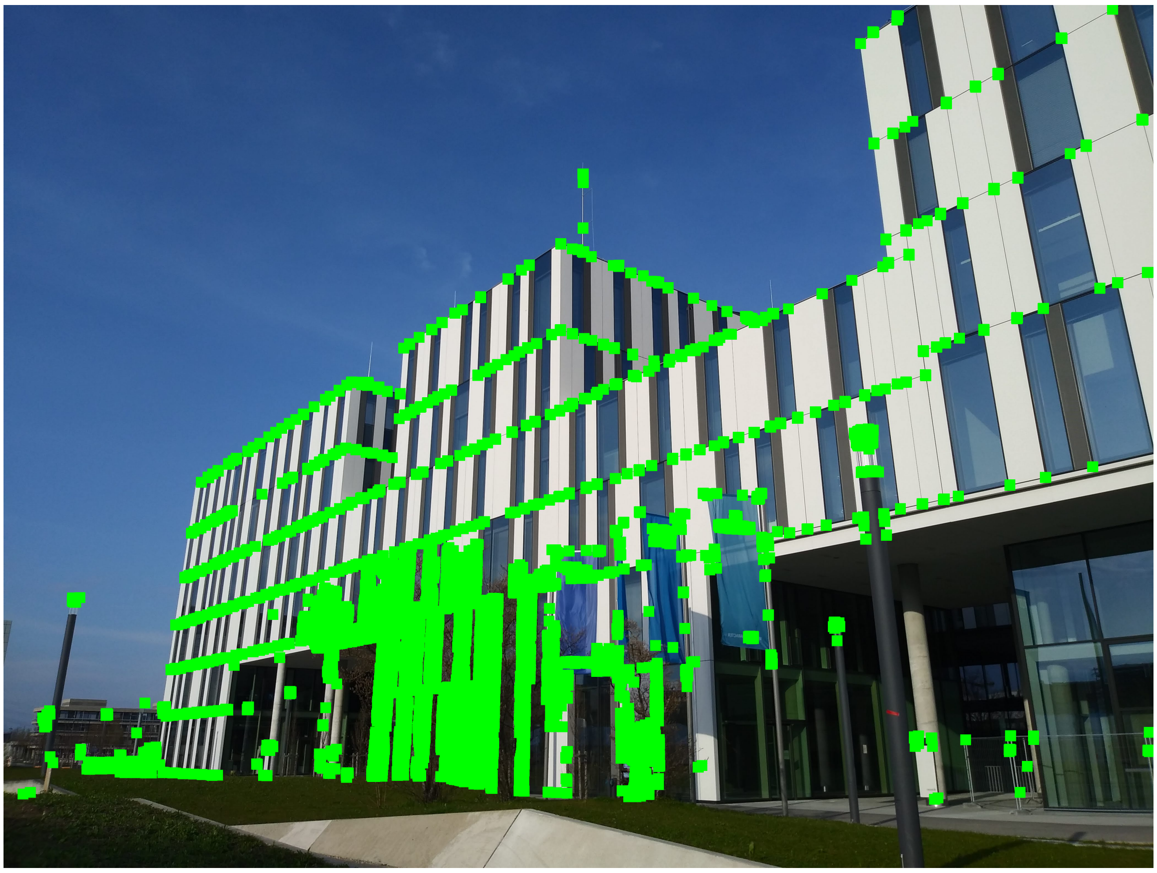


Figure 2.3.: Features points extracted using the Harris-Stephenson corner detector, in the image of the Galileo building on the Garching Campus of the Technical University of Munich

In this equation, **M** is called the Harris matrix, which can be diagonalized as

$$\mathbf{M} = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} \mathbf{u}_1{}^{\mathrm{T}} \\ \mathbf{u}_2{}^{\mathrm{T}} \end{bmatrix}, \tag{2.3}$$

where λ_1 and λ_2 are eigenvalues of the Harris matrix and $\mathbf{u}_1$ and $\mathbf{u}_2$ are eigenvectors corresponding to the eigenvalues. As shown in Eq. (2.2), the weighted SSD is quadratic to the pixel change $\Delta\mathbf{x}$ with the Harris matrix **M** as the structure tensor (second-moment matrix). Thus, the following ellipse defined with the eigenvalues (λ_1, λ_2) and eigenvectors ($\mathbf{u}_1$, $\mathbf{u}_2$) of the Harris matrix **M** can be used to gain an intuitive view of the gradient's shape:

$$\frac{(\mathbf{u}_1{}^{\mathrm{T}}\Delta\mathbf{x})^2}{\left(\frac{1}{\lambda_1}\right)^2} + \frac{(\mathbf{u}_2{}^{\mathrm{T}}\Delta\mathbf{x})^2}{\left(\frac{1}{\lambda_2}\right)^2} = \text{constant} \tag{2.4}$$

The eigenvectors are the axes of this ellipse, and they show the direction of the gradient changes.

The eigenvalues represent the level of intensity changes in the corresponding directions. By examining the eigenvalues, the pixels can be sorted into three groups: flat, edge, and corner. When a pixel is located on a flat surface (e.g. white wall), both eigenvalues of the Harris matrix are small. The pixels located on edges have one large and one small eigenvalues, so the ellipse is eccentric. When pixels are located on corner points, both eigenvalues of the Harris matrix are large, so the ellipse is small and circle-like.

Feature detectors Harris-Stephenson corner detector [Harris et al., 1988] uses the following value $r(\mathbf{M})$ computed with the eigenvalues of the Harris matrix $\mathbf{M}$ to detect corners as feature points:

$$\begin{aligned} r(\mathbf{M}) &= \lambda_1\lambda_2 - \alpha(\lambda_1 + \lambda_2)^2 \\ &= \det(\mathbf{M}) - \alpha(\operatorname{tr}(\mathbf{M})^2), \end{aligned}$$

where α is an empirical design factor ($0.04 < \alpha < 0.06$). If $r(\mathbf{M})$ is larger than the threshold, the pixel is recognized as a feature point. Fig. 2.3 shows the feature points extracted using the Harris-Stephenson corner detector in the image of the Galileo building on the Garching Campus of the Technical University of Munich.

Instead of using both eigenvalues of the Harris matrix, Shi-Tomasi detector [Shi et al., 1994] only examines the smaller eigenvalue. In this method, the pixels are accepted as feature points when the minimum eigenvalue is greater than the criteria. Both the Harris-Stephenson corner detector and Shi-Tomasi detector can extract the same feature points after the images have been rotated (rotation invariant) because the magnitude of the Harris matrix's eigenvalues are not changed after the images are rotated. However, these methods cannot extract the same features in the original images when the image is scaled (scale variant).

To resolve this scale variant problem, the Scale-Invariant Feature Transform (SIFT) [Lowe, 2004] proposes to search feature points in multiple scales by filtering the intensity $I(\mathbf{x}_i, \sigma)$ in the scale space using the Gaussian kernel $G(\mathbf{x}_i, \sigma))$:

$$D(\mathbf{x}_i, \sigma) = (G(\mathbf{x}_i, k\sigma) - G(\mathbf{x}_i, \sigma)) * I(\mathbf{x}_i, \sigma).$$

$D(\mathbf{x}_i, \sigma)$ is called the Difference of Gaussian (DoG), and the image scale is adjusted with σ. When images are blurred by setting σ to a large value, big corners in wider views can be detected. Each pixel's DoG is compared with the surrounding 8 pixels on the same scale and

Table 2.1.: Summary of feature detectors introduced in this section

Algorithm	Rotation	Scale	Remarks
Harris [Harris et al., 1988]	Inv.	Var.	-
Shi-Tomasi [Shi et al., 1994]	Inv.	Var.	-
SIFT [Lowe, 2004]	Inv.	Inv.	-
SURF [Bay et al., 2008]	Inv.	Inv.	Faster SIFT
FAST [Rosten and Drummond, 2006]	Inv.	Var.	-
BRISK [Leutenegger et al., 2011]	Inv.	Inv.	Scale invariant FAST

the 9 pixels of the upper and lower scales, and then a pixel is recognized as a feature point when its DoG is a local extrema. SIFT can extract feature points accurately, but it requires significant computing power, as it compares pixels across multiple scales.

Speeded Up Robust Features (SURF) [Bay et al., 2008] is a faster feature detector that requires less computing power than SIFT. Instead of comparing pixels to pixels, the SURF detector uses the integral of the image intensity to speed up the computational process. In addition, this method is more robust to image transformation compared to the SIFT detector.

Features from Accelerated Segment Test (FAST), proposed in [Rosten and Drummond, 2006], can detect feature points faster than SURF. This method examines the image intensity of the 16 pixels surrounding each pixel, evaluating whether the intensities of at least 12 contiguous pixels are greater than the intensity at the center pixel.

Although features can be extracted much quicker with FAST, it is scale variant. To address this, Binary Robust Invariant Scalable Keypoints (BRISK) [Leutenegger et al., 2011] examines images still in multiple scales (as SIFT), while retaining the binary searching method of FAST. All the feature detectors introduced in this section are summarized in Table 2.1.

Feature descriptors Each feature point is stored with its descriptor, which contains a function of the pixel intensity values in a region surrounding the feature point. The SIFT descriptor stores the non-binary information of 16×16 neighboring pixels. These pixels are sorted into 4×4 sized subgroups, and each group is represented by an orientation histogram with 8 bins. Consequently, $4 \times 4 \times 8$ numbers represent one feature point. The SURF descriptor [Bay et al., 2008] speeds up this process by comparing the integral of image gradients. In addition, the FAST descriptors converts the intensity values of 16 surrounding pixels to $\pm$ tags, and uses these tags as the descriptors.

Although features can be accurately matched with these non-binary descriptors, extensive computing power and large memory are required to create and match them. Binary descriptors

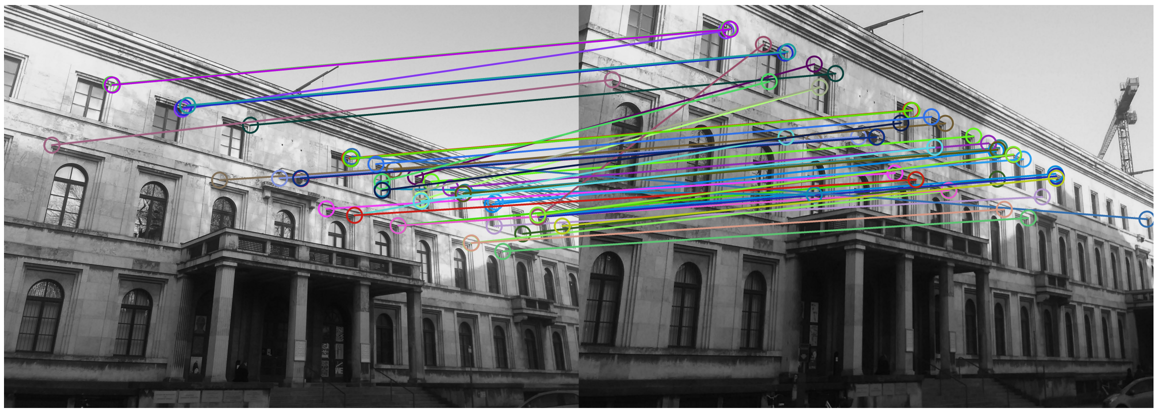


Figure 2.4.: Feature points matched using ORB, in the image of the Hochschule für Musik und Theater München, Munich

can be used to increase computation and memory efficiency. First, all non-binary information of the pixels is converted into 1s and 0s. Then, the similarity of two feature points can be simply computed as the hamming distance between two binary descriptors, which requires much less computing power than non-binary descriptors. For instance, the BRISK descriptor consists of the binary strings that result from the intensity comparison between the central feature point and a large number of pixels located on a series of concentric circles, centered at the feature location. Binary Robust Independent Elementary Features (BRIEF) descriptor describes each feature point using a set of feature vectors that represent the pixel intensity of the image patch including the feature point. The size of the BRIEF descriptor can be set as 16 or 32 or 64 bytes. Oriented FAST and Rotated BRIEF (ORB) is a combination of the FAST and Harris feature detectors with the BRIEF descriptor. The feature points are searched using FAST and Harris detectors at multiple image scales (scale invariant), and the features are described with the rBRIEF descriptor (32 bytes) which is the BRIEF descriptor with greater robustness to image rotation. Fig. 2.4 shows the feature points detected and associated using ORB, in the image of the Hochschule für Musik und Theater München, Munich.

Fast image classification and place recognition Before carrying out feature-to-feature comparison employing descriptors, similar images are classified into subgroups by place recognition (fast classification). Then, brute-force searches using feature descriptors need to be conducted only in the subgroups, which requires much less computing power. For the image classification, a hierarchical Bag of Words (BoW) model [Nister and Stewenius, 2006] can be applied. In this model, descriptors in each image are converted into a sparse numerical vector using visual vocabularies. As shown in Fig. 2.5, a visual vocabulary tree is created by the hierarchical

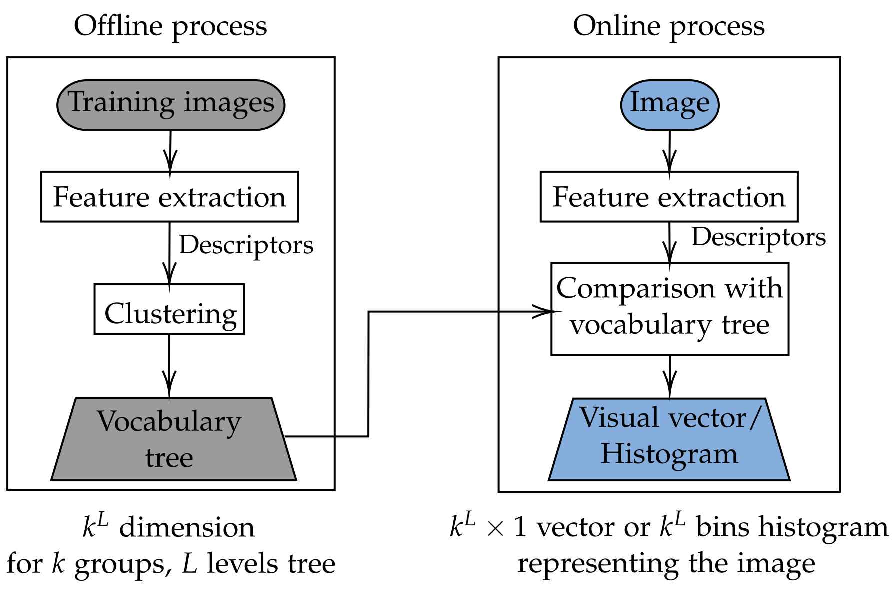


Figure 2.5.: An overview of the Bag of Words (BoW) method

recursive quantization of descriptors in the offline process. The descriptors of training images are sorted into k clusters, and each cluster is partitioned again into k subgroups. After repeating this clustering process L times, an L-level vocabulary tree is created. The size of the vocabulary tree generated with this process is Dk^L bytes for the D-dimensional descriptor, e.g. $D = 128$ if the SIFT descriptor is used to extract feature points in images.

After the vocabulary tree is created, the real time online process can be started to sort the incoming images into subgroups. The descriptors of the feature points detected in incoming images are compared to the vocabulary tree's root and the leaf nodes. At the end of the process, each image is represented as a visual vector $\mathbf{v}$ in the k^L dimension or as a histogram with k^L bins. The i-th element of the visual vector v_i (or the value of the i-th bin) is described as $v_i = n_i \omega_i$, where n_i is the number of descriptors that select the i-th node as the closest node, and ω_i is the weight of the node. Bags of Binary Words (DBoW) [Gálvez-López and Tardos, 2012] is an example of the BoW model created using binary descriptors. The visual vector represents each image's characteristics, so the hamming distance between two sparse vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ can represent the similarity of two images:

$$s(\mathbf{v}_1, \mathbf{v}_2) = \left\| \frac{\mathbf{v}_1}{||\mathbf{v}_1||} - \frac{\mathbf{v}_2}{||\mathbf{v}_2||} \right\|. \tag{2.5}$$

2.2. Map Initialization

When enough sets of feature point matches have been found between two different images, the relative pose between the camera frames can be estimated employing RANdom SAmple Consensus (RANSAC). Additionally, the 3D coordinates of map points can be estimated by back-projecting the feature matches from the 2D image frame to 3D space.

Fundamental matrix estimation with RANSAC As shown in Fig. 2.6, a 3×3 fundamental matrix $\mathbf{F}$ is first estimated using the feature matches. The fundamental matrix has a following relationship with the feature matches:

$$\tilde{\mathbf{x}}_{\prime}^{i\mathrm{T}} \mathbf{F} \tilde{\mathbf{x}}^{i} = 0, \tag{2.6}$$

where $\tilde{\mathbf{x}}^i$ and $\tilde{\mathbf{x}}_{\prime}^i$ are homogeneous coordinates of the feature in the first image and the feature match in the second image, respectively, i.e. $\tilde{\mathbf{x}}^i = [x^i, y^i, 1]^{\mathrm{T}}$ when x^i and y^i are the X and Y coordinates of the feature in the first image.

Eq. (2.6) can be re-written as

$$\begin{bmatrix} x'_1x_1 & x'_1y_1 & x'_1 & y'_1x_1 & y'_1y_1 & y'_1 & x_1 & y_1 & 1 \end{bmatrix} \mathbf{f} = 0, \tag{2.7}$$

where $\mathbf{f}$ is a 9×1 vector consisting of the fundamental matrix's elements f_{ij}, i.e. $\mathbf{f} = \begin{bmatrix} f_{11} & f_{12} & f_{13} & f_{21} & f_{22} & f_{23} & f_{31} & f_{32} & f_{33} \end{bmatrix}^{\mathrm{T}}$. This vector is 7DoF (not 9DoF) because the fundamental matrix is homogeneous and rank deficient. When M feature matches are found between two images, Eq. (2.7) can be expanded to

$$\underbrace{\begin{bmatrix} x'_1x_1 & x'_1y_1 & x'_1 & y'_1x_1 & y'_1y_1 & y'_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x'_Mx_M & x'_My_M & x'_M & y'_Mx_M & y'_My_M & y'_M & x_M & y_M & 1 \end{bmatrix}}_{M\times 9} \mathbf{f} = 0$$

$$\Leftrightarrow \mathbf{A}\mathbf{f} = 0. \tag{2.8}$$

An intermediate solution of Eq. (2.8) can be estimated employing the 8-point algorithm, as proposed in [Hartley, 1997]. Although $\mathbf{f}$ is 7DoF, more than seven sets of feature matches are used to estimate the vector because errors exist in feature matching and feature point locations.

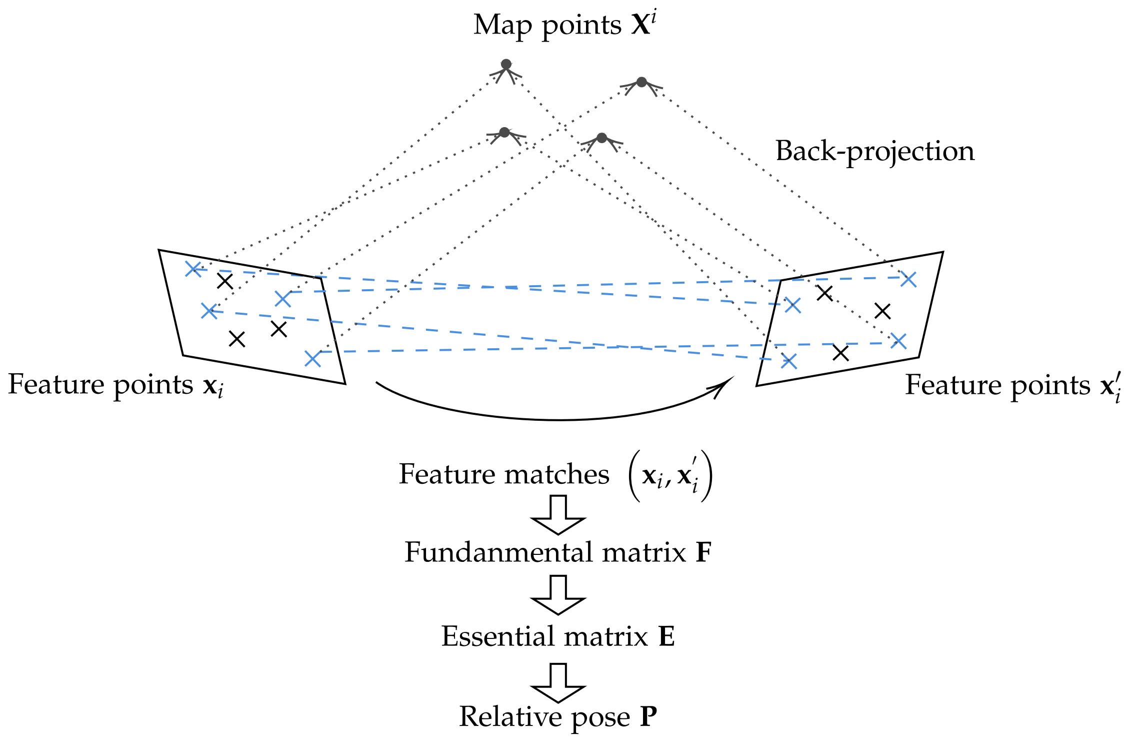

Map points $\mathbf{X}^i$
Back-projection
Feature points $\mathbf{x}_i$
Feature points $\mathbf{x}_i'$
Feature matches $\left(\mathbf{x}_i, \mathbf{x}_i'\right)$
Fundanmental matrix $\mathbf{F}$
Essential matrix $\mathbf{E}$
Relative pose $\mathbf{P}$

Figure 2.6.: Relative pose estimation between two image frames using the feature point matches and back-projection of map points

First, a least-square problem is defined using $\mathbf{A}$ in Eq. (2.8):

$$\mathbf{f}^* = \underset{\mathbf{f}}{\operatorname{argmin}} ||\mathbf{A}\,\mathbf{f}||^2. \tag{2.9}$$

This problem can be deterministically solved using the Singular Value Decomposition (SVD). When $\mathbf{A}$ is decomposed as $\mathbf{A} = \mathbf{UW}(\mathbf{V})^\intercal$, the solution of Eq. (2.9) is the eigenvector $\mathbf{v}_{_i}$ of $\mathbf{A}$ corresponding to the smallest eigenvalue. Using this intermediate solution $\mathbf{f}^*$, a 3×3 matrix $\mathbf{F}^*$ is built as

$$\mathbf{F}^* = \begin{bmatrix} f_{11}^* & f_{12}^* & f_{13}^* \\ f_{21}^* & f_{22}^* & f_{23}^* \\ f_{31}^* & f_{32}^* & f_{33}^* \end{bmatrix}. \tag{2.10}$$

Then, another least-square problem is defined, considering the fundamental matrix's constraint, $\det \mathbf{F} = 0$:

$$\hat{\mathbf{F}} = \underset{\det \mathbf{F}=0}{\operatorname{argmin}} ||\mathbf{F} - \mathbf{F}^*||^2. \tag{2.11}$$

After estimating the solution of this least-square problem, $\hat{\mathbf{F}}$ is decomposed as

$$\hat{\mathbf{F}} = \mathbf{U} \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix} (\mathbf{V})^{\mathrm{T}}, \tag{2.12}$$

and then σ_3 is replaced to 0 to ensure the rank deficiency of the fundamental matrix:

$$\hat{\mathbf{F}} = \mathbf{U} \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & 0 \end{bmatrix} (\mathbf{V})^{\mathrm{T}}. \tag{2.13}$$

The following cost is computed using each feature match $(\tilde{\mathbf{x}}^i, \tilde{\mathbf{x}}^i_{\prime})$ and the fundamental matrix $\hat{\mathbf{F}}$ estimated as Eq. (2.13) to sort only the inliers out of the M feature matches:

$$\mathcal{F}(\tilde{\mathbf{x}}^i, \tilde{\mathbf{x}}^i_{\prime}, \hat{\mathbf{F}}) = d^2(\tilde{\mathbf{x}}^i, \tilde{\mathbf{x}}^i_{*}, \hat{\mathbf{F}}) + d^2(\tilde{\mathbf{x}}^i_{\prime}, \tilde{\mathbf{x}}^i_{\prime *}, \hat{\mathbf{F}}). \tag{2.14}$$

In this equation, the feature location $\tilde{\mathbf{x}}^i_{*}$ is obtained by projecting the feature point in the second image $\tilde{\mathbf{x}}^i_{\prime}$ to the first image using $\hat{\mathbf{F}}$. Similarly, $\tilde{\mathbf{x}}^i_{\prime *}$ is computed using the feature point in the first image $\tilde{\mathbf{x}}^i$ and $\hat{\mathbf{F}}$. The squared distance between $\tilde{\mathbf{x}}^i$ and $\tilde{\mathbf{x}}^i_{*}$ is denoted with $d^2(\tilde{\mathbf{x}}^i, \tilde{\mathbf{x}}^i_{*}, \hat{\mathbf{F}})$. Feature match $(\tilde{\mathbf{x}}^i, \tilde{\mathbf{x}}^i_{\prime})$ is sorted as an outlier when the cost $\mathcal{F}(\tilde{\mathbf{x}}^i, \tilde{\mathbf{x}}^i_{\prime}, \hat{\mathbf{F}})$ is greater than the threshold. $\hat{\mathbf{F}}$ is accepted as a solution only when the total number of the inliers is sufficient. When sufficient inliers are found, the fundamental matrix is re-estimated with LSE using only the inliers:

$$\hat{\mathbf{F}} = \underset{\mathbf{F}}{\operatorname{argmin}}\, d^2(\tilde{\mathbf{x}}^i, \tilde{\mathbf{x}}^i_{*}, \mathbf{F}) + d^2(\tilde{\mathbf{x}}^i_{\prime}, \tilde{\mathbf{x}}^i_{\prime *}, \mathbf{F}). \tag{2.15}$$

Essential matrix and relative pose estimation After the fundamental matrix has been successfully estimated, the first camera frame is defined as the reference frame of VSLAM, i.e. the first frame's camera pose matrix is defined as

$$\mathbf{P}_1 = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix}.$$

Then, the relative pose from the first camera frame to the second one $\mathbf{P}_{21}$ can be defined as the pose matrix of the second camera frame:

$$\mathbf{P}_2 = \mathbf{P}_{21} = \begin{bmatrix} \mathbf{R}_{21} & \mathbf{t}_{21} \\ \mathbf{0} & 1 \end{bmatrix}.$$

The essential matrix $\mathbf{E}$ needs to be estimated to determine the rotation matrix $\mathbf{R}_{21}$ and the translation vector $\mathbf{t}_{21}$. The essential matrix has the following relationship with the fundamental matrix $\mathbf{F}$:

$$\mathbf{E} = (\mathbf{K}')^{\mathrm{T}}\mathbf{F}\mathbf{K}, \tag{2.16}$$

where $\mathbf{K}$ is the camera calibration matrix. When the essential matrix is decomposed by SVD as

$$\mathbf{E} = \mathbf{U} \begin{bmatrix} \sigma & 0 & 0 \\ 0 & \sigma & 0 \\ 0 & 0 & 0 \end{bmatrix} (\mathbf{V})^{\mathrm{T}},$$

four sets of rotation and translation can be obtained as the candidates:

(a) $\mathbf{R}_{21} = \mathbf{U}\mathbf{W}\mathbf{V}^{\mathrm{T}}$, $\mathbf{t}_{21} = \mathbf{u}_3$

(b) $\mathbf{R}_{21} = \mathbf{U}\mathbf{W}^{\mathrm{T}}\mathbf{V}^{\mathrm{T}}$, $\mathbf{t}_{21} = \mathbf{u}_3$

(c) $\mathbf{R}_{21} = \mathbf{U}\mathbf{W}\mathbf{V}^{\mathrm{T}}$, $\mathbf{t}_{21} = -\mathbf{u}_3$

(d) $\mathbf{R}_{21} = \mathbf{U}\mathbf{W}^{\mathrm{T}}\mathbf{V}^{\mathrm{T}}$, $\mathbf{t}_{21} = -\mathbf{u}_3$

The DLT algorithm [Abdel-Aziz et al., 2015] can be used to select one of the rotation matrix and translation vector sets. First, feature point matches are back-projected to 3D space using each candidate. Then, one of the candidates is selected when the most points back-projected to 3D space are located in front of both image frames. The points back-projected to 3D space are stored as map points.

In addition, Global Bundle Adjustment (GBA) [Mur-Artal et al., 2015, Mur-Artal and Tardós, 2017a] can be run to obtain more accurate camera poses and map points coordinates. A factor graph is created including two camera frames' poses and the initial set of map points as state variables. Then, the LSE is executed to find the optimal poses and map point coordinates that minimize the sum of the squared re-projection error.

2.3. Tracking, Local Mapping, and Loop Closing

Once the map is successfully initialized, the VSLAM's back-end starts to sequentially estimate the camera poses using the previous pose estimates as well as the map points observed in the previous images (tracking). In addition, new map points are created by back-projecting the feature matches newly found between the current image and the previous images. The camera pose and map point estimation errors are accumulated over time because tracking is a dead-reckoning process. The errors are reduced using feature points observations. Only the latest camera poses and map points observed in those camera frames are involved in the process to avoid computational overload (local mapping). Loop closures can be additionally run to obtain more accurate estimates. Tracking, local mapping, and loop closures are run separately (not sequentially) using different processing units (multi-threading), as proposed in Parallel Tracking and Mapping (PTAM) [Klein and Murray, 2007].

Tracking For the tracking process, the features observed in the current image are compared to the features observed in the previous image. A search window can be set when the robot's motion model is available [Mur-Artal and Tardós, 2017a]. First, the current camera pose is predicted using the camera motion model. Then, each map point observed in the previous image is projected to the current frame using the predicted pose, as can be seen in Fig. 2.7. A search window is set in the current image as a circle centered on the projected point, and then only the feature points inside of the search window are examined. This is a much efficient method than examining the entire feature points observed in the current image. When sufficient feature matches are not found using this approach, the camera has to localize itself by examining all the map points stored in the database, which requires substantial computing power.

As shown in Fig. 2.8a, a factor graph is created including the current camera pose $\mathbf{T}_\mathrm{i}$ and map point observations $\mathbf{X}^j$ when sufficient feature matches are detected. Only the current camera pose is set as the state variable in this graph. The feature point observations of map points are depicted as the lines between the camera pose and map points. The least-square problem is defined using this graph to find the optimal current camera pose $\hat{\mathbf{T}}_\mathrm{i}$ that minimizes the sum of the squared re-projection errors:

$$\hat{\mathbf{T}}_\mathrm{i} = \underset{\mathbf{T}_\mathrm{i}}{\operatorname{argmin}} \sum_j \|\mathbf{x}^j - \pi(\mathbf{T}_\mathrm{i}, \mathbf{X}^j)\|^2_{\Sigma_j}, \tag{2.17}$$

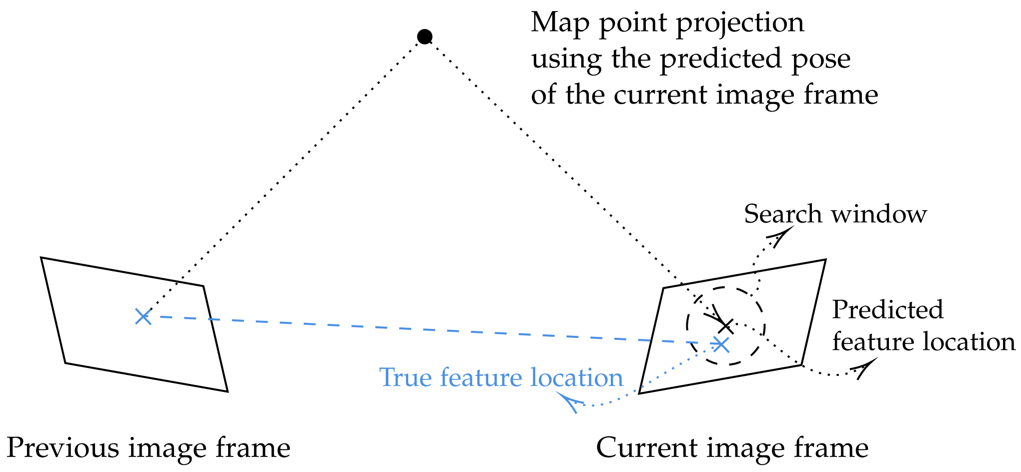


Figure 2.7.: Search window setting to detect the feature match in the current image frame

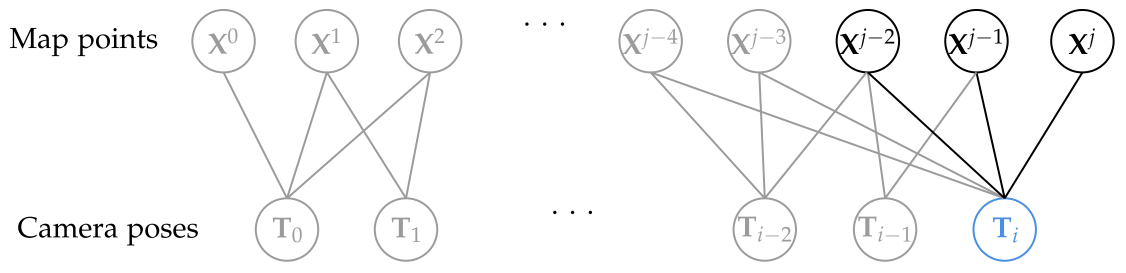
where $\mathbf{x}^j$ denotes the 2D feature point location in the current frame associated with the j-th map point $\mathbf{X}^j$. The function $\pi(\mathbf{T}_i, \mathbf{X}^j)$ computes the pixel location projected from the map point $\mathbf{X}^j$ to the current image using the current pose $\mathbf{T}_i$ as

$$\pi(\mathbf{T}_i, \mathbf{X}^j) = \begin{bmatrix} f_u \frac{x_j}{z_j} + c_u \\ f_v \frac{y_j}{z_j} + c_v \end{bmatrix},$$

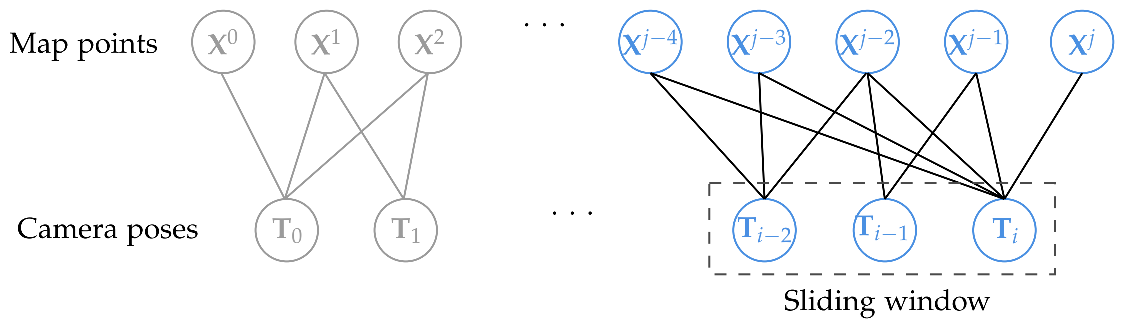
where (f_u, f_v) and (c_u, c_v) are the focal length and the principle point of the camera, respectively, and $[x_j, y_j, z_j]^{\mathrm{T}} = \mathbf{R}_i\mathbf{X}^j + \mathbf{t}_i$.

Keyframes selection and new map points estimation To manage the data size, only the camera frames observing distinctive feature points are stored in the database, as proposed in [Mur-Artal et al., 2015, Mur-Artal and Tardós, 2017a]. These frames are called keyframes, and each keyframe is saved with its covisibility graph which includes the keyframes observing the common map points. The connections between the keyframes in the covisibility graph are weighted by the similarity score. The more common map points two keyframes observe, the higher weight is set between them. When the current image frame is selected as a keyframe, new map points are created by back-projecting the feature point matches that are newly found between the current keyframe and the previous keyframes.

Local mapping The tracking process consecutively estimates the camera poses subject to the previous pose estimates, so the error is accumulated over time. Moreover, map points are only exploited as measurements, but not updated (not set as the state variables) in the



(a) A factor graph created for tracking. Only the current camera pose is set as the state variable



(b) A factor graph created for local mapping. The keyframe poses in the sliding window and the map points observed in those keyframes are set as the state variables

Figure 2.8.: Factor graphs created for tracking and local mapping. Blue circular nodes are the state variables, and black circular nodes and lines are measurements. All the others in gray are not involved in the processes

tracking process. Thus, an additional process is required to mitigate the estimation errors of the keyframe poses and map point coordinates. It is computationally challenging to continuously process all the keyframes and map points stored in the database, so the factor graph is marginalized using a sliding window as can be seen in Fig. 2.8b. The keyframes' poses $\mathbf{T}_\mathrm{i}$ in the sliding window are set as the state variables, as well as the map points $\mathbf{X}^j$ observed in those keyframes. Then, the following least-square problem is defined employing this factor graph:

$$\hat{\mathbf{T}}_\mathrm{i}, \hat{\mathbf{X}}^j = \underset{\mathbf{T}_\mathrm{i}, \mathbf{X}^j}{\operatorname{argmin}} \sum_i \sum_j \|\mathbf{x}^j - \pi(\mathbf{T}_\mathrm{i}, \mathbf{X}^j)\|^2_{\Sigma_j} . \tag{2.18}$$

Loop Closing Local mapping copes with only the most recent keyframes and the map points observed in them, so the estimation errors are indefinitely accumulated over time. As proposed in [Mur-Artal and Tardós, 2014, Mur-Artal et al., 2015], the loop closing technique can substantially mitigate the estimation errors using the map points observed at the initial phase of the mission. The map point coordinates are more accurate at the initial phase than the ones recently estimated since the estimation errors are not significantly accumulated at the beginning of the mission.

First, the database is examined using fast association to find loop candidate keyframes that observe a similar scene as the current keyframe. Then, a one-to-one map point comparison is carried out only between the loop candidates and the current keyframe. When sufficient matches are found between the current keyframe and each loop candidate, the 7DoF relative pose matrix from the current keyframe to each loop candidate $\hat{\mathbf{S}}_{\mathrm{C_lC_i}}$ is estimated using the Horn's method [Horn, 1987].

The outliers of the map point matches are sorted using RANSAC. The projection error of each map point is computed using the following function:

$$\mathbf{e}_i^k = \mathbf{x}_i^k - \pi(\hat{\mathbf{S}}_{\mathrm{C_lC_i}}, \mathbf{X}_l^k) \tag{2.19}$$

$$\mathbf{e}_l^k = \mathbf{x}_l^k - \pi(\hat{\mathbf{S}}_{\mathrm{C_lC_i}}, \mathbf{X}_i^k). \tag{2.20}$$

The k-th map point's coordinates with respect to the loop candidate keyframe is denoted as $\mathbf{X}_l^k$ in Eq. (2.19). As can be seen in Fig. 2.9, $\mathbf{x}_i^k$ is the 2D feature measurement's location in the current keyframe associated to the k-th map point, while $\pi(\mathbf{S}_{\mathrm{C_lC_i}}, \mathbf{X}_l^k)$ is the 2D location projected from $\mathbf{X}_l^k$ to the current keyframe using $\hat{\mathbf{S}}_{\mathrm{C_lC_i}}$. The difference between the two locations is $\mathbf{e}_c^k$. Similarly, Eq. (2.20) computes the projection error in the loop candidate keyframe. The k-th map point is accepted as an inlier when both projection errors Eq. (2.19) and Eq. (2.20) are smaller than the criteria. The loop candidate that ensures the most inliers is selected as the final loop keyframe.

The relative pose matrix from the current keyframe to the final loop keyframe is re-estimated only using the inliers. Then, the current keyframe pose $\mathbf{T}_{\mathrm{i}}$ is converted to the 7DoF similarity matrix $\mathbf{S}_{\mathrm{i}}$ using the relative pose estimate $\hat{\mathbf{S}}_{\mathrm{C_lC_i}}$ and the loop keyframe's pose with respect to the VSLAM's local reference frame $\mathbf{T}_{\mathrm{C_lL}}$, i.e. $\mathbf{S}_{\mathrm{i}} = \hat{\mathbf{S}}_{\mathrm{C_iC_l}} \mathbf{T}_{\mathrm{C_lL}}$. The other keyframes' poses are also converted to the 7DoF matrices using the relative poses from the current keyframe to the other $\mathbf{S}_{\mathrm{C_jC_i}}$ and the 7DoF current keyframe pose $\mathbf{S}_{\mathrm{i}}$, i.e. $\mathbf{S}_{\mathrm{j}} = \mathbf{S}_{\mathrm{C_jC_i}} \mathbf{S}_{\mathrm{i}}$.

A factor graph is created including only the 7DoF keyframe poses $\mathbf{S}_{\mathrm{i}}$ as the state variables (blue circular nodes), as shown in Fig. 2.10a. Odometry measurements between the keyframes

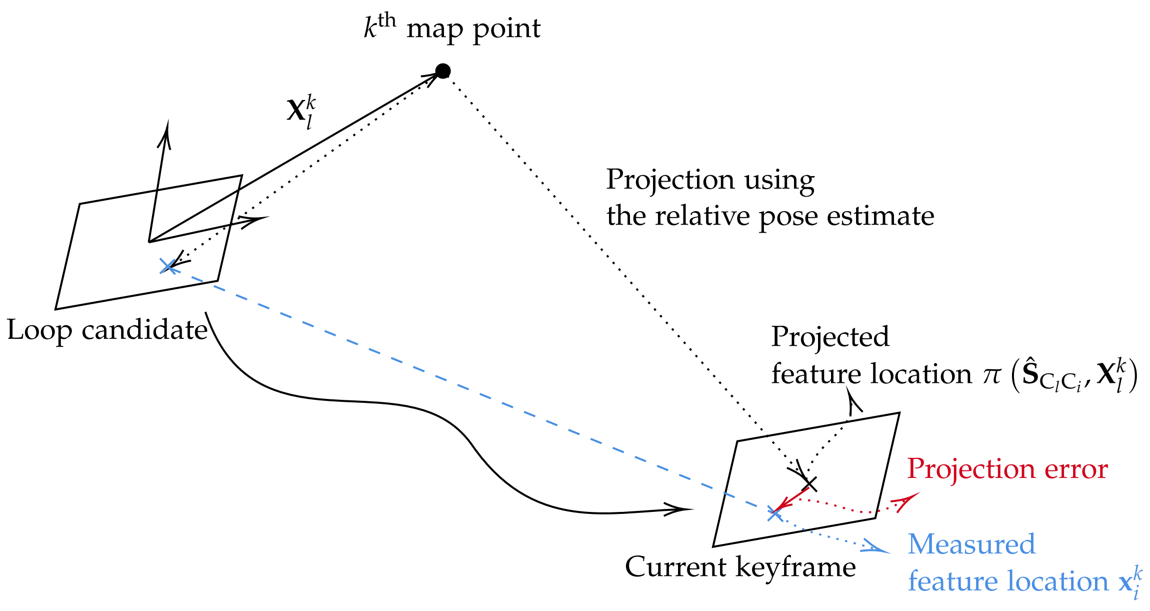

kth map point
$\mathbf{X}_l^k$
Projection using
the relative pose estimate
Loop candidate
Projected
feature location $\pi\left(\hat{\mathbf{S}}_{C_lC_i}, \mathbf{X}_l^k\right)$
Projection error
Measured
feature location $\mathbf{x}_i^k$
Current keyframe

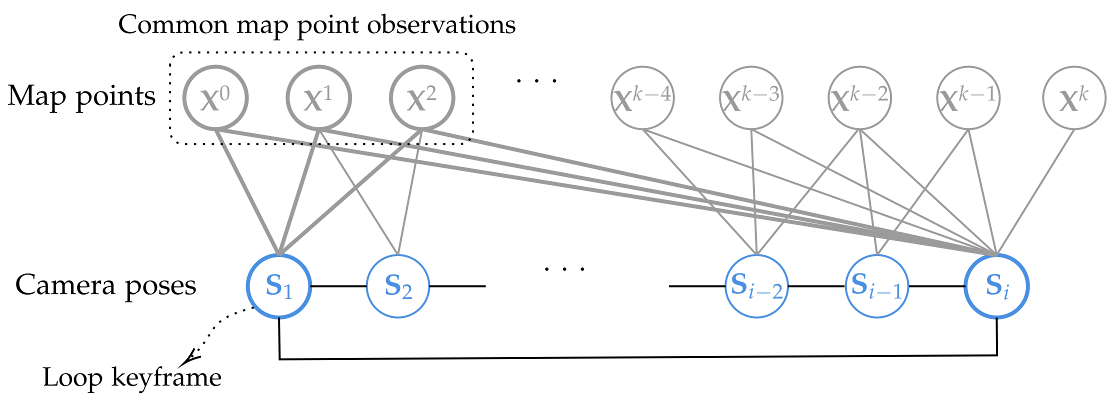
Figure 2.9.: Projection error: the difference between the feature location measured in the current keyframe $\mathbf{x}_i^k$ and the location projected from the map point into the current keyframe $\pi(\hat{\mathbf{S}}_{C_lC_i}, \mathbf{X}_l^k)$

are added as the black lines, and the relative pose from the current keyframe to the loop keyframe is also added as the loop measurement between two keyframes. LSE is applied to find the optimal keyframe poses that minimize the sum of the odometry and loop measurement errors.

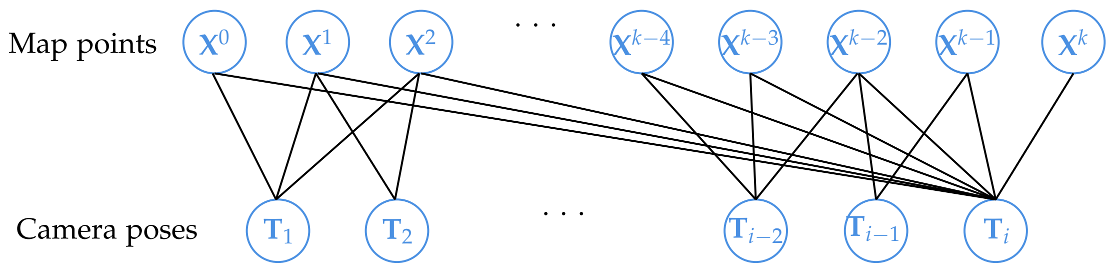
Furthermore, Global Bundle Adjustment (GBA) is run to obtain more accurate keyframe poses and map point coordinates. Fig. 2.10b shows the factor graph created for GBA in which all the keyframes and map points in the map database are set as the state variables (blue nodes). The feature point observations of the map points are added as the measurement factors between the keyframe and map point nodes (black lines). Global Bundle Adjustment (GBA) is used to find the optimal keyframe poses and map point coordinates that minimize the sum of the squared re-projection errors of the feature point measurements depicted in the graph.

2.4. Scale Ambiguity Problem of Monocular VSLAM

The absolute scale of camera positions and map point coordinates cannot be estimated only using a monocular camera. Fig. 2.11 shows this inherent scale ambiguity problem of monocular VSLAM. The distance b between the previous camera frame and current camera frame cannot be measured using a monocular camera, so the camera position and map point coordinates can be estimated only up-to-scale, i.e. the current camera frame could be located at any position on



(a) A factor graph created to optimize only the keyframe poses before running GBA. The keyframe poses are set as the state variables, and odometry measurements are added between them. The relative pose between the current keyframe and loop keyframe is also added as a measurement factor



(b) A factor graph created for GBA. All the keyframe poses map points are set as the state variables

Figure 2.10.: Factor graphs created for loop closing. Blue circular nodes are the state variables, and black circular nodes and lines are measurements. The gray nodes and lines are not involved in the processes

the blue line, and the map point could be located at any point on the red line.

A stereo camera can be used to resolve this problem since the distance between the left and right cameras of a rig is fixed and known, unlike a monocular camera. The absolute scale (depth) of the camera position map point coordinates can be estimated using the baseline length b of a stereo rig. However, the absolute scale can be inaccurately estimated, as the scale estimation accuracy is proportional to the squared distance from the camera to the map point and disproportional to the baseline length [Olson and Abi-Rached, 2010]. The stereo camera's baseline length cannot be extended significantly due to the hardware limitation, so the scale estimate can be very inaccurate when map points are located far from the camera.

Additionally, other types of sensors providing absolute measurements (e.g. Inertial Measure-

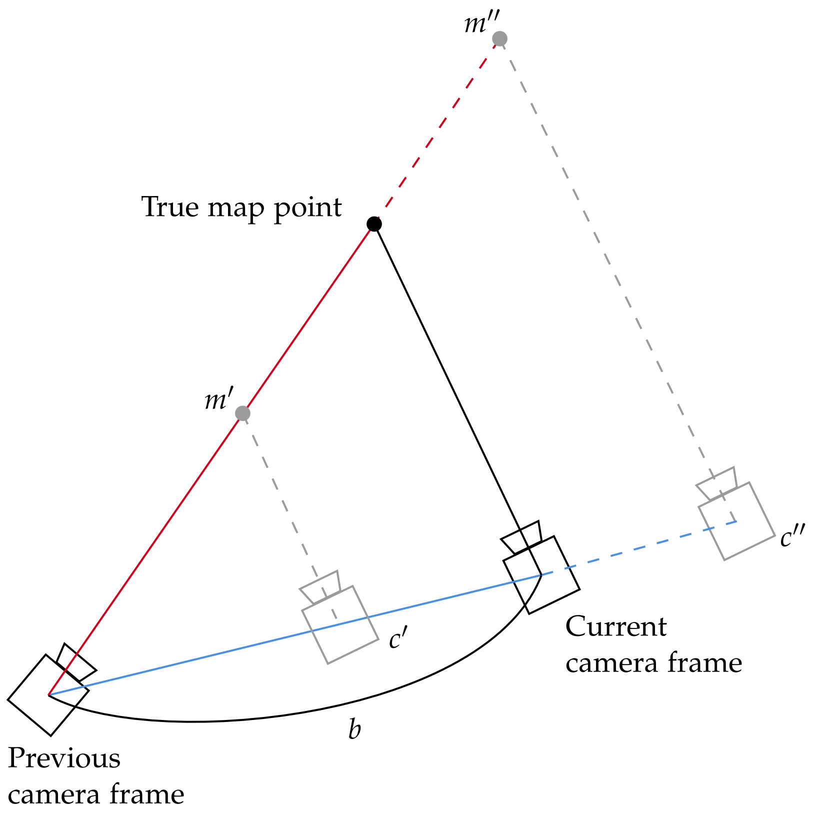


Figure 2.11.: Scale ambiguity problem of monocular VSLAM: the distance b between the previous and current cameras is unknown, so the current camera frame could be located at any position on the blue line, and the map point could be located at any point on the red line

ment Unit (IMU) and GNSS) can be also used to resolve the inherent problem of monocular VSLAM. The following chapter proposes the fusion of visual odometry and range measurements to accurately estimate the camera poses and map point coordinates without the inherent scale ambiguity.

3. VOR-SLAM: SLAM using Visual Odometry and Range Measurements

Robots can employ VSLAM for their self-localization and mapping when GNSS is unavailable or unreliable. When a monocular camera is exploited for VSLAM, the robot's poses and map points can be estimated only up-to-scale since no absolute measurements are available. A stereo camera with a fixed and known baseline length can be used to resolve this scale ambiguity problem, but due to its short baseline length, stereo VSLAM cannot accurately estimate the absolute scale when the robot observes distant objects. Alternatively, Visual-INertial System (VINS) can be used to solve the scale ambiguity problem as proposed in [Qin et al., 2018, Mur-Artal and Tardós, 2017b]. However, the estimation errors of camera poses and map points are accumulated over time without a bound, as SLAM using VINS is a dead-reckoning process. The loop closing technique can be applied to mitigate the estimation errors, but it requires the robot to travel back to the previous locations to detect a loop. Moreover, a big storage unit and large computing power are needed to fuse the map.

Data fusion of ranges and visual measurements is proposed to accurately estimate robot's poses and create a map with much fewer computational requirements. For instance, an EKF-based data fusion of visual, inertial, and UWB ranges measurements is proposed in [Benini et al., 2013], showing the cm-level positioning accuracy for indoor applications with quadcopters. In addition, tightly-coupled and graph-based data fusion methods are proposed in [Shi et al., 2018, Wang et al., 2017], and these algorithms are demonstrated with public image datasets and UWB ranges.

This chapter proposes a graph-based SLAM using visual odometry and range measurements (VOR-SLAM), which is the improved work from my previous research in [Lee et al., 2018a, Lee et al., 2018b]. Instead of using visual measurements, such as feature points, VOR-SLAM only exploits monocular visual odometry (6DoF) and range measurements (1DoF, scalar values) to estimate the robot's poses and map points, so it needs much less computing power compared to VSLAM using the loop closing technique. Moreover, range measurements can be obtained using available signals in many environments, without requiring complex additional infrastructure. For example, cellular networks, such as 5G links, can be used to acquire range measurements in urban areas, as reviewed in [Nikookar and Oonincx, 2016].

In the following section, the system architecture is first introduced, as well as the prediction models of visual odometry and range measurements, and then the data fusion and map point update methods are detailed. The system's estimation accuracy and computational requirement are analyzed with both public image datasets, and demonstrated with experimental data which is obtained in two different outdoor environments (football field and gravel pit).

3.1. System Overview and Measurement Prediction Models

As shown in Fig. 3.1, first, feature points are detected from raw images, and then associated (matched) in the front-end. In addition, ranges are obtained with the Time of Flight (ToF) measurements of the signals transmitted and received between the tag and the anchor point (two way ranging). Using the feature matches provided from the front-end, visual odometry (pose tracking and local mapping) estimates the camera poses, creates new map points, and optimizes the latest camera poses and map points by local bundle adjustments. To maintain the database size, only the keyframes' poses are stored, as well as the map points observed in the keyframes. Visual-range data fusion is executed only using the keyframes data and ranges (without map points) to mitigate the estimation error and resolve the scale ambiguity of monocular VSLAM. The map points coordinates are updated after the data fusion using the differences between the keyframe poses before and after the fusion. In this research project, the open source code of ORB-SLAM [Mur-Artal et al., 2015, Mur-Artal and Tardós, 2017a] is employed (without loop closing), but any other VSLAM methods can be used to obtain the odometry measurements.

Fig. 3.2 illustrates the system setup with the mathematical notations. A camera C^i and an onboard ranging-tag sensor T^i are mounted on the robot. The local frame is set as the VSLAM's reference frame, which coincides with the initial pose of the camera frame. The local frame's front direction is the Z-axis, and the direction from the left camera to the right camera is the X-axis. The global frame is defined as the local frame rotated 90° with respect to its X-direction because the local frame's XZ-plane is the horizontal plane (not conventional). The relative pose

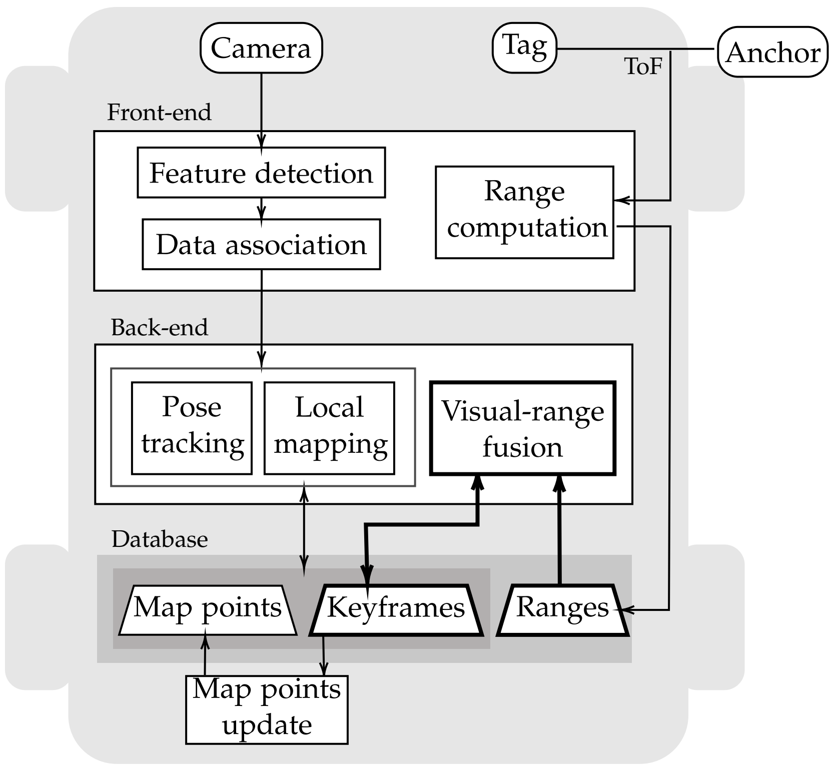


Figure 3.1.: The system overview of VOR-SLAM

from the local frame to the global frame $\mathbf{S}_{\mathrm{GL}}$ is

$$\mathbf{S}_{\mathrm{GL}} = \begin{bmatrix} \mathbf{R}_{\mathrm{GL}} & \mathbf{t}_{\mathrm{LG}} \\ \mathbf{0}_{1\times 3} & (s_{\mathrm{GL}})^{-1} \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{\mathrm{X}}(-90^\circ) & \mathbf{0}_{3\times 1} \\ \mathbf{0} & (s_{\mathrm{GL}})^{-1} \end{bmatrix}. \tag{3.21}$$

$\mathbf{R}_{\mathrm{GL}}$ and $\mathbf{t}_{\mathrm{LG}}$ are the rotation matrix and the translation vector from the local frame to the global frame, respectively. The scale difference from the local frame to the global frame is denoted as s_{GL}.

The 7DoF camera pose with respect to the global frame $\mathbf{S}^{t_i}_{\mathrm{GC}}$ is defined by multiplying the 6DoF camera local pose $\mathbf{T}^{t_i}_{\mathrm{LC}}$ and the 7DoF relative similarity matrix from the local to the global

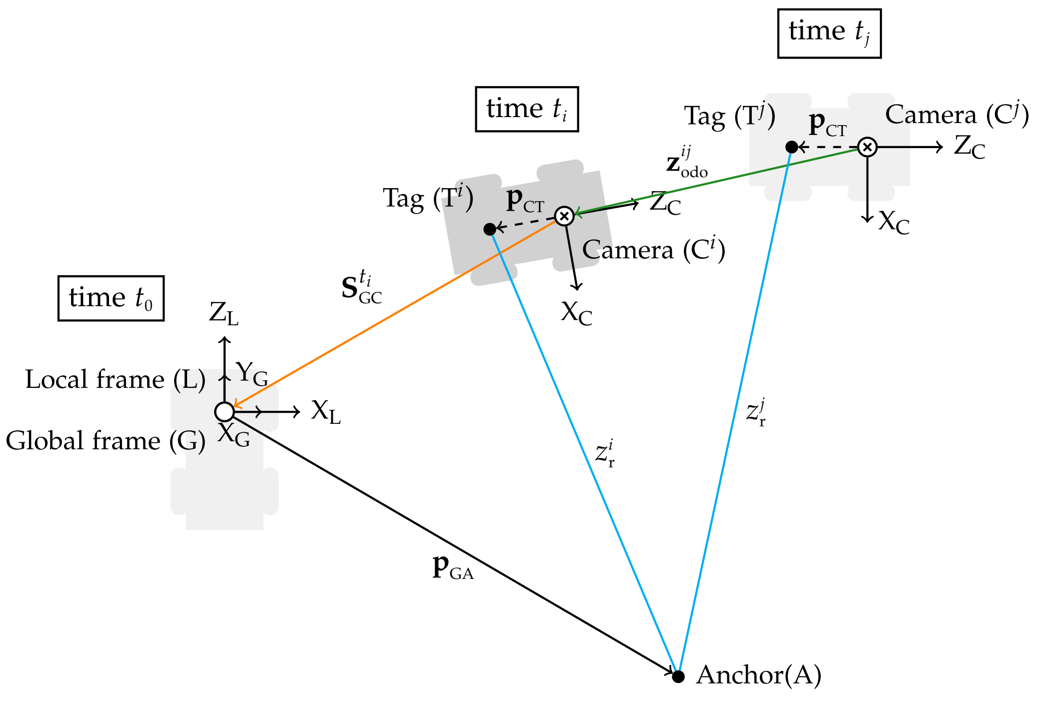


Figure 3.2.: A single rover with an onboard camera and ranging tag. The camera pose with respect to the global frame is denoted with $\mathbf{S}_{GC}^{t}$ (orange). The visual odometry measurement between t_i and t_j is denoted with $\mathbf{z}_{odo}^{ij}$ (green), and the range measurements at t_i is denoted as z_r^i (blue).

frame $\mathbf{S}_{GL}$:

$$\mathbf{S}_{GC}^{t_i} = \mathbf{S}_{GL}\mathbf{T}_{LC}^{t_i} = \begin{bmatrix} \mathbf{R}_{GL} & \mathbf{t}_{LG} \\ \mathbf{0}_{1\times3} & (s_{GL})^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{R}_{LC}^{t_i} & \mathbf{t}_{CL}^{t_i} \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix}$$
$$= \begin{bmatrix} \mathbf{R}_{GL}\mathbf{R}_{LC}^{t_i} & \mathbf{R}_{GL}\mathbf{t}_{CL}^{t_i} + \mathbf{t}_{LG} \\ \mathbf{0}_{1\times3} & (s_{GL}^{t_i})^{-1} \end{bmatrix}$$
$$= \begin{bmatrix} \mathbf{R}_{GC}^{t_i} & \mathbf{t}_{CG}^{t_i} \\ \mathbf{0}_{1\times3} & (s_{GC}^{t_i})^{-1} \end{bmatrix}. \tag{3.22}$$

The state variables of VOR-SLAM are the 7DoF keyframe poses at all timestamps:

$$\mathbf{\Theta} = \left[\mathbf{S}_{GC}^{t_1}, \mathbf{S}_{GC}^{t_2}, \ldots, \mathbf{S}_{GC}^{t_N}\right]. \tag{3.23}$$

The measurements of VOR-SLAM are modeled with a function $\mathbf{h}(\mathbf{\Theta})$ with zero mean Gaussian noise $\boldsymbol{\eta} \sim \mathcal{N}(0, \mathbf{\Sigma})$:

$$\mathbf{z} = \mathbf{h}(\mathbf{\Theta}) + \boldsymbol{\eta}.$$

First, range measurements between the onboard ranging-tag module and the anchor at t_i (blue in Fig. 3.2) is model as

$$h_{\mathrm{r}}^{i}(\mathbf{S}_{\mathrm{GC}}^{t_i}) = ||{}_{\mathrm{C}}\mathbf{p}_{\mathrm{TA}}^{t_i}|| = ||-\mathbf{p}_{\mathrm{CT}} + \mathbf{p}_{\mathrm{CG}}^{t_i} + {}_{\mathrm{C}}\mathbf{p}_{\mathrm{GA}}||, \tag{3.24}$$

where ${}_{\mathrm{C}}\mathbf{p}_{\mathrm{GA}}$ is the position vector from the global frame to the anchor with respect to the camera frame. In addition, ${}_{\mathrm{C}}\mathbf{p}_{\mathrm{TA}}^{t_i}$ is the first three elements of the homogeneous vector ${}_{\mathrm{C}}\tilde{\mathbf{p}}_{\mathrm{TA}}^{t_i}$ with respect to the camera frame with the scale factor $\mathrm{s}_{\mathrm{CG}}^{t_i}$:

$$\begin{aligned}
{}_{\mathrm{C}}\tilde{\mathbf{p}}_{\mathrm{TA}}^{t_i} = \begin{pmatrix} {}_{\mathrm{C}}\mathbf{p}_{\mathrm{TA}}^{t_i} \\ \left(\mathrm{s}_{\mathrm{CG}}^{t_i}\right)^{-1} \end{pmatrix} &= \begin{bmatrix} \mathbf{R}_{\mathrm{CG}}^{t_i} & \mathbf{t}_{\mathrm{GC}}^{t_i} \\ \mathbf{0} & \left(\mathrm{s}_{\mathrm{CG}}^{t_i}\right)^{-1} \end{bmatrix} \begin{pmatrix} \mathbf{p}_{\mathrm{GA}} \\ 1 \end{pmatrix} - \begin{pmatrix} \mathbf{p}_{\mathrm{CT}} \\ \left(\mathrm{s}_{\mathrm{CG}}^{t_i}\right)^{-1} \end{pmatrix} \\
&= \left(\mathbf{S}_{\mathrm{GC}}^{t_i}\right)^{-1} \tilde{\mathbf{p}}_{\mathrm{GA}} - \tilde{\mathbf{p}}_{\mathrm{CT}} \\
&= \tilde{\mathbf{p}}_{\mathrm{CA}}^{t} - \tilde{\mathbf{p}}_{\mathrm{CT}}.
\end{aligned} \tag{3.25}$$

The anchor position in the global frame $\mathbf{p}_{\mathrm{GA}}$ is assumed to be known to users, as well as the position vector from the camera to the ranging-tag $\mathbf{p}_{\mathrm{CT}}$. The ranging errors can be simply defined as the difference between the measurement z_{r}^{i} and the value computed with the model:

$$e_{\mathrm{r}}^{i}(\mathbf{S}_{\mathrm{GC}}^{t_i}, z_{\mathrm{rr}}^{i}) = h_{\mathrm{r}}^{i}(\mathbf{S}_{\mathrm{GC}}^{t_i}) - z_{\mathrm{r}}^{i}. \tag{3.26}$$

The odometry measurement between t_i and t_j (green in Fig. 3.2) is modeled as the 4×4 matrix that is the multiplication of the 6DoF camera poses with respect to the local frame at t_i and t_j:

$$\begin{aligned}
\mathbf{h}_{\mathrm{odo}}^{ij}(\mathbf{S}_{\mathrm{GC}}^{t_i}, \mathbf{S}_{\mathrm{GC}}^{t_j}) &= \left(\mathbf{S}_{\mathrm{GC}}^{t_i}\right)^{-1} \mathbf{S}_{\mathrm{GC}}^{t_j} \\
&= \left(\mathbf{S}_{\mathrm{GL}}\mathbf{T}_{\mathrm{LC}}^{t_i}\right)^{-1} \left(\mathbf{S}_{\mathrm{GL}}\mathbf{T}_{\mathrm{LC}}^{t_j}\right) \\
&= \left(\mathbf{T}_{\mathrm{LC}}^{t_i}\right)^{-1} \mathbf{T}_{\mathrm{LC}}^{t_j}.
\end{aligned} \tag{3.27}$$

The error between the the odometry measurement $\mathbf{z}_{\mathrm{odo}}^{ij}$ and the predicted one $\mathbf{h}_{\mathrm{odo}}^{ij}(\mathbf{S}_{\mathrm{GC}}^{t_i}, \mathbf{S}_{\mathrm{GC}}^{t_j})$ is described with the 4×4 error matrix:

$$\mathbf{E}_{\mathrm{odo}}^{ij}(\mathbf{S}_{\mathrm{GC}}^{t_i}, \mathbf{S}_{\mathrm{GC}}^{t_j}, \mathbf{z}_{\mathrm{vo}}^{ij}) = (\mathbf{z}_{\mathrm{vo}}^{ij})^{-1} \mathbf{h}_{\mathrm{vo}}^{ij}(\mathbf{S}_{\mathrm{GC}}^{t_i}, \mathbf{S}_{\mathrm{GC}}^{t_j}). \tag{3.28}$$

This error matrix can be to be converted to the 7×1 twist coordinates (vector) using the logarithmic relationship between the twist coordinates and Lie group matrices as described in Eq. (A.90):

$$\mathbf{e}^{ij}_{\mathrm{odo}}(\mathbf{S}^{t_i}_{\mathrm{GC}}, \mathbf{S}^{t_j}_{\mathrm{GC}}, \mathbf{z}^{ij}_{\mathrm{vo}}) = \ln\left((\mathbf{z}^{ij}_{\mathrm{vo}})^{-1}\, \mathbf{h}^{ij}_{\mathrm{vo}}(\mathbf{S}^{t_i}_{\mathrm{GC}}, \mathbf{S}^{t_j}_{\mathrm{GC}}) \right). \tag{3.29}$$

A prior knowledge can be described with the state variable of the first pose $\hat{\mathbf{S}}^{t_1}_{\mathrm{GC}}$, i.e. $\mathbf{h}_{\mathrm{pri}}(\mathbf{S}^{t_1}_{\mathrm{GC}}) = \hat{\mathbf{S}}^{t_1}_{\mathrm{GC}}$, and the error matrix of the prior knowledge can be defined in Sim(3) as

$$\mathbf{E}_{\mathrm{pri}}(\mathbf{S}^{t_1}_{\mathrm{GC}}) = (\mathbf{z}_{\mathrm{pri}})^{-1}\, \mathbf{h}_{\mathrm{pri}}(\mathbf{S}^{t_1}_{\mathrm{GC}}). \tag{3.30}$$

Similar to the odometry measurements, this error matrix can be converted to the 7×1 twist coordinates using logarithmic mapping, so the errors between the predicted values and the measurements can be described as 7×1 vectors:

$$\mathbf{e}_{\mathrm{pri}}(\mathbf{S}^{t_1}_{\mathrm{GC}}) = \ln\left((\mathbf{z}_{\mathrm{pri}})^{-1}\, \mathbf{h}_{\mathrm{pri}}(\mathbf{S}^{t_1}_{\mathrm{GC}}) \right). \tag{3.31}$$

3.2. Visual-Range Data Fusion and Map Points Update

Fig. 3.3 shows the factor graph created for the visual-range data fusion. The state variables $\mathbf{S}^{t_i}_{\mathrm{GC}}$ are illustrated as the circular nodes, and the conditional probability density of the states given the measurements (measurement factors) are depicted as the black dots between or at the nodes. For instance, the odometry factors ϕ^{ij}_{odo} are the likelihood of the state variables, $\mathbf{S}^{t_i}_{\mathrm{GC}}$ and $\mathbf{S}^{t_j}_{\mathrm{GC}}$, given the odometry measurements $\mathbf{z}^{ij}_{\mathrm{odo}}$, so these factors are depicted as the binary factors between the nodes. Assuming the zero-mean Gaussian noise with the covariance $\boldsymbol{\Sigma}_{\mathrm{vo}}$, the odometry factors can be defined as

$$\phi^{ij}_{\mathrm{odo}} = l(\mathbf{S}^{t_i}_{\mathrm{GC}}, \mathbf{S}^{t_j}_{\mathrm{GC}} | \mathbf{z}^{ij}_{\mathrm{vo}}) \propto \exp\{-\frac{1}{2} || \mathbf{e}^{ij}_{\mathrm{odo}}(\mathbf{S}^{t_i}_{\mathrm{GC}}, \mathbf{S}^{t_j}_{\mathrm{GC}}, \mathbf{z}^{ij}_{\mathrm{odo}}) ||^2_{\Sigma_{\mathrm{odo}}}\}. \tag{3.32}$$

The prior factor ϕ_{pri} is the conditional probability density of the initial camera pose $\mathbf{S}^{t_1}_{\mathrm{GC}}$, given the current best estimate of the initial pose, so it is depicted as the unary factor at the initial camera pose. With the zero-mean Gaussian noise assumption, the prior factor is defined as

$$\phi_{\mathrm{pri}} = P(\mathbf{S}^{t_1}_{\mathrm{GC}} | \mathbf{z}_{\mathrm{pri}}) \propto \exp\{-\frac{1}{2} || \mathbf{e}_{\mathrm{pri}}(\mathbf{S}^{t_1}_{\mathrm{GC}}, \mathbf{z}_{\mathrm{pri}}) ||^2_{\Sigma_{\mathrm{pri}}}\}. \tag{3.33}$$

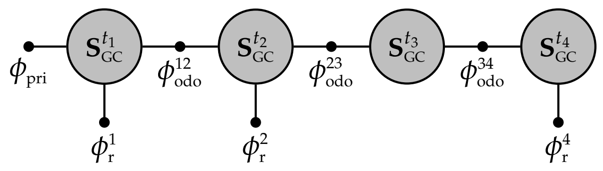

$\mathbf{S}_{\mathrm{GC}}^{t_1}$
$\mathbf{S}_{\mathrm{GC}}^{t_2}$
$\mathbf{S}_{\mathrm{GC}}^{t_3}$
$\mathbf{S}_{\mathrm{GC}}^{t_4}$
ϕ_{pri}
ϕ_{odo}^{12}
ϕ_{odo}^{23}
ϕ_{odo}^{34}
ϕ_{r}^{1}
ϕ_{r}^{2}
ϕ_{r}^{4}

Figure 3.3.: The factor graph including the state variables $\mathbf{S}_{\mathrm{GC}}^{t_i}$ (circular nodes). The prior factor ϕ_{pri} is depicted at the first keyframe, the odometry factors ϕ_{odo}^{ij} are illustrated between two circular nodes. Range factors ϕ_{r}^{i} are depicted at three keyframes which obtain range measurements

In addition, range factors ϕ_{r}^{i} are added at each node whenever the range measurement is available. This factor is defined as the likelihood of the state $\mathbf{S}_{\mathrm{GC}}^{t_i}$, given the range measurement z_{r}^{i}. Similar to the other measurement factors, the range factor is described as

$$\phi_{\mathrm{r}}^{i} = l(\mathbf{S}_{\mathrm{GC}}^{t_i} | z_{\mathrm{r}}^{i}) \propto \exp\{-\frac{1}{2} || e_{\mathrm{r}}^{i}(\mathbf{S}_{\mathrm{GC}}^{t_i}, z_{\mathrm{r}}^{i}) ||_{\Sigma_{\mathrm{r}}}^{2}\}, \tag{3.34}$$

with the zero-mean Gaussian noise assumption.

The state variables are estimated using the Maximum A Posteriori (MAP) approach with all the factors in the graph shown in Fig. 3.3:

$$\hat{\boldsymbol{\Theta}} = \underset{\boldsymbol{\Theta}}{\operatorname{argmax}} \, \phi_{\mathrm{pri}} \prod_{i,j} \phi_{\mathrm{odo}}^{ij} \prod_{l} \phi_{\mathrm{r}}^{l}. \tag{3.35}$$

Assuming constant a priori and zero-mean Gaussian noise in the measurements, this MAP is equivalent to the weighted Least Squares Estimation (LSE) that minimizes the sum of the squared measurement errors:

$$\begin{aligned}
\hat{\boldsymbol{\Theta}} = \underset{\boldsymbol{\Theta}}{\operatorname{argmin}} \, & \underbrace{\left(\mathbf{e}_{\mathrm{pri}}(\mathbf{S}_{\mathrm{GC}}^{t_1}, \mathbf{z}_{\mathrm{pri}})\right)^{\mathrm{T}} \boldsymbol{\Sigma}_{\mathrm{pri}}^{-1} \mathbf{e}_{\mathrm{pri}}(\mathbf{S}_{\mathrm{GC}}^{t_1}, \mathbf{z}_{\mathrm{pri}})}_{\text{a prior error}} \\
& \underbrace{+ \sum_{i,j} \left(\mathbf{e}_{\mathrm{odo}}^{ij}(\mathbf{S}_{\mathrm{GC}}^{t_i}, \mathbf{S}_{\mathrm{GC}}^{t_j}, \mathbf{z}_{\mathrm{odo}}^{ij})\right)^{\mathrm{T}} \boldsymbol{\Sigma}_{\mathrm{odo}}^{-1} \mathbf{e}_{\mathrm{odo}}^{ij}(\mathbf{S}_{\mathrm{GC}}^{t_i}, \mathbf{S}_{\mathrm{GC}}^{t_j}, \mathbf{z}_{\mathrm{odo}}^{ij})}_{\text{visual odometry measurement errors}} \\
& \underbrace{+ \sum_{l} \left(e_{\mathrm{r}}^{l}(\mathbf{S}_{\mathrm{GC}}^{t_l}, z_{\mathrm{r}}^{l})\right)^{\mathrm{T}} \boldsymbol{\Sigma}_{\mathrm{r}}^{-1} e_{\mathrm{r}}(\mathbf{S}_{\mathrm{GC}}^{t_l}, z_{\mathrm{r}}^{l})}_{\text{range measurement errors}} \\
\triangleq \underset{\boldsymbol{\Theta}}{\operatorname{argmin}} \mathcal{F}(\boldsymbol{\Theta}, \mathbf{Z}) &
\end{aligned} \tag{3.36}$$

This nonlinear LSE is solved using the Levenberg-Marquardt algorithm [Levenberg, 1944, Marquardt, 1963]. The algorithm iteratively estimates the camera poses in the vector format, i.e. the 7×1 twist coordinates

$$\Delta\hat{\boldsymbol{\theta}} = \left[\Delta\hat{\boldsymbol{\xi}}^{t_1}_{GC}, \ldots, \Delta\hat{\boldsymbol{\xi}}^{t_N}_{GC}\right], \tag{3.37}$$

by solving the following equation

$$(\mathbf{H} + d\mathbf{I})\Delta\hat{\boldsymbol{\theta}} = -\mathbf{b}. \tag{3.38}$$

When N is the total number of the keyframes, $\mathbf{H}$ is the $7 \times 7N$ Hessian matrix as

$$\mathbf{H} = \underbrace{\left(\mathbf{J}_{\mathbf{e}_{\text{pri}}}\right)^{\mathrm{T}} \boldsymbol{\Sigma}_{\text{pri}}\mathbf{J}_{\mathbf{e}_{\text{pri}}}}_{\text{a prior}} + \underbrace{\sum_{i,j} \left(\mathbf{J}^{ij}_{\mathbf{e}_{\text{odo}}}\right)^{\mathrm{T}} \boldsymbol{\Sigma}_{\text{odo}}\mathbf{J}^{ij}_{\mathbf{e}_{\text{odo}}}}_{\text{visual odometry}} + \underbrace{\sum_{l} \left(\mathbf{J}^{l}_{r}\right)^{\mathrm{T}} \boldsymbol{\Sigma}_{\text{r}}\mathbf{J}^{l}_{e_{\text{r}}}}_{\text{ranges}}. \tag{3.39}$$

In addition, $\mathbf{b}$ on the right hand side of Eq. (3.38) is the 7×1 vector as

$$\mathbf{b} = \underbrace{\left(\mathbf{J}_{\mathbf{e}_{\text{pri}}}\right)^{\mathrm{T}} \boldsymbol{\Sigma}_{\text{pri}}\mathbf{e}_{\text{pri}}}_{\text{a prior}} + \underbrace{\sum_{i,j} \left(\mathbf{J}^{ij}_{\mathbf{e}_{\text{odo}}}\right)^{\mathrm{T}} \boldsymbol{\Sigma}_{\text{odo}}\mathbf{e}^{ij}_{\text{odo}}}_{\text{visual odometry}} + \underbrace{\sum_{l} \left(\mathbf{J}^{l}_{r}\right)^{\mathrm{T}} \boldsymbol{\Sigma}_{\text{r}}e^{l}_{\text{r}}}_{\text{ranges}}. \tag{3.40}$$

The Jacobian of the odometry error function in Eq. (3.39) and Eq. (3.40) is the $7 \times 7N$ matrix

$$\mathbf{J}^{ij}_{\mathbf{e}_{\text{odo}}}(\boldsymbol{\theta}) = \left[\mathbf{0} \ldots \frac{\partial \mathbf{e}^{ij}_{\text{vo}}(\mathbf{S}^{t_i}_{\text{GC}}, \mathbf{S}^{t_j}_{\text{GC}}, \mathbf{z}^{ij}_{\text{vo}})}{\partial \boldsymbol{\xi}^{t_i}_{GC}}, \frac{\partial \mathbf{e}^{ij}_{\text{vo}}(\mathbf{S}^{t_i}_{\text{GC}}, \mathbf{S}^{t_j}_{\text{GC}}, \mathbf{z}^{ij}_{\text{vo}})}{\partial \boldsymbol{\xi}^{t_j}_{GC}} \ldots \mathbf{0}\right], \tag{3.41}$$

where the partial derivatives of the odometry error function of the pose at t_i and t_j are

$$\frac{\partial \mathbf{e}^{ij}_{\text{odo}}(\mathbf{S}^{t_i}_{\text{GC}}, \mathbf{S}^{t_j}_{\text{GC}}, \mathbf{z}^{ij}_{\text{odo}})}{\partial \boldsymbol{\xi}^{t_i}_{GC}} = -\text{Adj}\left(\left(\text{s}^{\mathbf{t_j}}_{\text{GC}}\right)^{-\mathbf{1}} \text{s}^{\mathbf{t_i}}_{\text{GC}}\right) \tag{3.42}$$

$$\frac{\partial \mathbf{e}^{ij}_{\text{odo}}(\mathbf{S}^{t_i}_{\text{GC}}, \mathbf{S}^{t_j}_{\text{GC}}, \mathbf{z}^{ij}_{\text{odo}})}{\partial \boldsymbol{\xi}^{t_j}_{GC}} = \mathbf{I}_7. \tag{3.43}$$

The adjoint $\text{Adj}\left(\left(\text{s}^{\mathbf{t_j}}_{\text{GC}}\right)^{-\mathbf{1}} \text{s}^{\mathbf{t_i}}_{\text{GC}}\right)$ can be computed using (A.114).

The partial derivative of the prior error function is the $7 \times 7N$ matrix:

$$\mathbf{J}_{\mathbf{e}_{\text{pri}}}(\boldsymbol{\theta}) = \left[\frac{\partial \mathbf{e}_{\text{pri}}(\mathbf{S}^{t_1}_{\text{GC}})}{\partial \boldsymbol{\xi}^{t_1}_{GC}}, \ldots, \mathbf{0}\right], \tag{3.44}$$

where the partial derivative is

$$\frac{\partial \mathbf{e}_{\text{pri}}(\mathbf{S}_{\text{GC}}^{t_1})}{\partial \boldsymbol{\xi}_{GC}^{t_1})} = \mathbf{I}_7. \tag{3.45}$$

The Jacobian of the ranging error function in Eq. (3.39) and Eq. (3.40) is the $1 \times 7N$ matrix

$$\mathbf{J}_{e_{\text{r}}}^{l}(\boldsymbol{\theta}) = \left[\mathbf{0} \ldots \frac{\partial e_{\text{r}}^{i}(\mathbf{S}_{\text{GC}}^{t_i}, z_{\text{r}}^{i})}{\partial \boldsymbol{\xi}_{GC}^{t_i}} \ldots \mathbf{0}\right]. \tag{3.46}$$

The partial derivative of the error function is

$$\begin{aligned}\frac{\partial e_{\text{r}}^{i}(\mathbf{S}_{\text{GC}}^{t_i}, h_{\text{r}}^{i})}{\partial \boldsymbol{\xi}_{GC}^{t_i}} &= \frac{\partial h_{\text{r}}^{i}(\mathbf{S}_{\text{GC}}^{t_i})}{\partial \boldsymbol{\xi}_{GC}^{t_i}} \\ &= \frac{\partial ||{}_{\text{C}}\mathbf{p}_{\text{TA}}^{t_i}||}{\partial \boldsymbol{\xi}_{GC}^{t_i}} = \frac{\partial ||{}_{\text{C}}\mathbf{p}_{\text{TA}}^{t_i}||}{\partial {}_{\text{C}}\mathbf{p}_{\text{TA}}^{t_i}} \frac{\partial {}_{\text{C}}\mathbf{p}_{\text{TA}}^{t_i}}{\partial {}_{\text{C}}\tilde{\mathbf{p}}_{\text{TA}}^{t_i}} \frac{\partial {}_{\text{C}}\tilde{\mathbf{p}}_{\text{TA}}^{t_i}}{\partial \boldsymbol{\xi}_{GC}^{t_i}},\end{aligned} \tag{3.47}$$

where

$$\frac{\partial ||{}_{\text{C}}\mathbf{p}_{\text{TA}}^{t_i}||}{\partial {}_{\text{C}}\mathbf{p}_{\text{TA}}^{t_i}} = \frac{1}{||{}_{\text{C}}\mathbf{p}_{\text{TA}}^{t_i}||} \left({}_{\text{C}}\mathbf{p}_{\text{TA}}^{t_i}\right)^{\text{T}} \tag{3.48}$$

$$\frac{\partial {}_{\text{C}}\mathbf{p}_{\text{TA}}^{t_i}}{\partial {}_{\text{C}}\tilde{\mathbf{p}}_{\text{TA}}^{t_i}} = \begin{bmatrix}\mathbf{I}_3 & \mathbf{0}_{3\times1}\end{bmatrix} \tag{3.49}$$

$$\begin{aligned}\frac{\partial {}_{\text{C}}\tilde{\mathbf{p}}_{\text{TA}}^{t_i}}{\partial \boldsymbol{\xi}_{GC}^{t_i}} = \frac{\partial}{\partial \boldsymbol{\xi}_{GC}^{t_i}} \left(\tilde{\mathbf{p}}_{\text{CA}}^{t_i} - \tilde{\mathbf{p}}_{\text{CT}}\right) &= \text{H}[\tilde{\mathbf{p}}_{\text{CA}}^{t_i}] - \begin{bmatrix}\mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & 1\end{bmatrix} \\ &= \begin{bmatrix}[\mathbf{p}_{\text{CA}}^{t_i}]_{\times} & -\text{s}_{\text{GC}}^{t_i}\mathbf{I}_3 & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \text{s}_{\text{GC}}^{t_i}\end{bmatrix} - \begin{bmatrix}\mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & 1\end{bmatrix} \\ &= \begin{bmatrix}[\mathbf{p}_{\text{CA}}^{t_i}]_{\times} & -\text{s}_{\text{GC}}^{t_i}\mathbf{I}_3 & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \text{s}_{\text{GC}}^{t_i} - 1\end{bmatrix}.\end{aligned} \tag{3.50}$$

By substituting Eq. (3.48), Eq. (3.49), and Eq. (3.50) to Eq. (3.47), the partial derivative can be re-written as

$$\begin{aligned}\frac{\partial e_{\text{r}}^{i}(\mathbf{S}_{\text{GC}}^{t_i}, \mathbf{z}_{\text{r}}^{i})}{\partial \boldsymbol{\xi}_{GC}^{t_i}} &= \begin{bmatrix}\frac{\partial h_{\text{r}}^{i}(\mathbf{S}_{\text{GC}}^{t_i})}{\partial \boldsymbol{\omega}} & \frac{\partial h_{\text{r}}^{i}(\mathbf{S}_{\text{GC}}^{t_i})}{\partial \boldsymbol{\nu}} & \frac{\partial h_{\text{r}}^{i}(\mathbf{S}_{\text{GC}}^{t_i})}{\partial \lambda}\end{bmatrix} \\ &= \frac{1}{||{}_{\text{C}}\mathbf{p}_{\text{TA}}^{t_i}||} \begin{bmatrix}({}_{\text{C}}\mathbf{p}_{\text{TA}}^{t_i})^{\text{T}}[\mathbf{p}_{\text{CA}}^{t_i}]_{\times} & -\text{s}_{\text{GC}}^{t_i}({}_{\text{C}}\mathbf{p}_{\text{TA}}^{t_i})^{\text{T}} & 0\end{bmatrix}.\end{aligned} \tag{3.51}$$

The optimal twist coordinates $\Delta\hat{\boldsymbol{\theta}}$ estimated using Eq. (3.38) are converted to the matrices in the Lie algebra domain sim(3), $\left[\Delta\hat{\boldsymbol{\Xi}}_{GC}^{t_1}, \ldots, \Delta\hat{\boldsymbol{\Xi}}_{GC}^{t_N}\right]$, and then each matrix $\hat{\mathbf{S}}_{\text{GC}}^{t_i}$ is updated using

(A.123):

$$\hat{\mathbf{S}}_{\mathrm{GC}}^{t_i} \leftarrow \mathbf{S}_{\mathrm{GC}}^{t_i} \oplus \Delta\hat{\mathbf{S}}_{\mathrm{GC}}^{t_i} = \mathbf{S}_{\mathrm{GC}}^{t_i} \exp \Delta\hat{\boldsymbol{\Xi}}_{GC}^{t_i}. \tag{3.52}$$

After the LSE, the map points coordinates are updated using the differences between keyframe poses before and after the LSE. First, the keyframe poses with respect to the global frame $\hat{\mathbf{S}}_{\mathrm{GC}}^{t_i}$ are converted to the local frame $\hat{\mathbf{S}}_{\mathrm{LC}}^{t_i}$ as

$$\begin{aligned}
\hat{\mathbf{S}}_{\mathrm{LC}}^{t_i} &= \left(\mathbf{S}_{\mathrm{GL}}\right)^{-1} \hat{\mathbf{S}}_{\mathrm{GC}}^{t_i} \\
&= \begin{bmatrix} \left(\mathbf{R}_{\mathrm{GL}}\right)^{\mathrm{T}} & -\mathrm{s}_{\mathrm{GL}}\left(\mathbf{R}_{\mathrm{GL}}\right)^{\mathrm{T}}\mathbf{t}_{\mathrm{LG}} \\ \mathbf{0}_{1\times 3} & \mathrm{s}_{\mathrm{GL}} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{R}}_{\mathrm{GC}}^{t_i} & \hat{\mathbf{t}}_{\mathrm{CG}}^{t_i} \\ \mathbf{0}_{1\times 3} & \left(\hat{\mathrm{s}}_{\mathrm{GC}}^{t}\right)^{-1} \end{bmatrix} \\
&= \begin{bmatrix} \left(\mathbf{R}_{\mathrm{GL}}\right)^{\mathrm{T}}\hat{\mathbf{R}}_{\mathrm{GC}}^{t_i} & -\left(\mathbf{R}_{\mathrm{GL}}\right)^{\mathrm{T}}\hat{\mathbf{t}}_{\mathrm{CG}}^{t_i} - \left(\hat{\mathrm{s}}_{\mathrm{GC}}^{t_i}\right)^{-1} \mathrm{s}_{\mathrm{GL}}\left(\mathbf{R}_{\mathrm{GL}}\right)^{\mathrm{T}}\mathbf{t}_{\mathrm{LG}} \\ \mathbf{0}_{1\times 3} & \left(\hat{\mathrm{s}}_{\mathrm{GC}}^{t_i}\right)^{-1} \mathrm{s}_{\mathrm{GL}} \end{bmatrix} \\
&= \begin{bmatrix} \hat{\mathbf{R}}_{\mathrm{LC}}^{t_i} & \hat{\mathbf{t}}_{\mathrm{CL}}^{t_i} \\ \mathbf{0}_{1\times 3} & \left(\hat{\mathrm{s}}_{\mathrm{LC}}^{t_i}\right)^{-1} \end{bmatrix}.
\end{aligned} \tag{3.53}$$

Then, 6DoF matrices $\mathbf{T}_{\mathrm{LC}}^{t_i}$ in SE(3) are defined with the translation vector $\hat{\mathbf{t}}_{\mathrm{LC}}^{t_i}$ scaled with $\mathrm{s}_{\mathrm{GC}}^{t_i}$:

$$\hat{\mathbf{T}}_{\mathrm{LC}}^{t_i} = \begin{bmatrix} \hat{\mathbf{R}}_{\mathrm{LC}}^{t_i} & \hat{\mathrm{s}}_{\mathrm{LC}}^{t_i}\hat{\mathbf{t}}_{\mathrm{CL}}^{t_i} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix}.$$

Using the keyframe pose after the data fusion $\hat{\mathbf{T}}_{\mathrm{LC}}^{t_i}$ and before the fusion $\mathbf{T}_{\mathrm{LC}}^{t_i}$, the map points coordinates in the local reference frame $\mathbf{X}_{\mathrm{L}}^{i}$ are updated as

$$\hat{\hat{\mathbf{X}}}_{\mathrm{L}}^{i} = \hat{\mathbf{T}}_{\mathrm{LC}}^{t_i} \left(\mathbf{T}_{\mathrm{LC}}^{t_i}\right)^{-1} \begin{pmatrix} \mathbf{X}_{\mathrm{L}}^{i} \\ 1 \end{pmatrix} = \begin{pmatrix} \hat{\mathbf{X}}_{\mathrm{L}}^{i} \\ \left(\hat{\mathrm{s}}_{\mathrm{LC}}^{t_i}\right)^{-1} \end{pmatrix} \tag{3.54}$$

Lastly, the scale $\hat{\mathrm{s}}_{\mathrm{LC}}^{t_i}$ is multiplied with the 3×1 vector $\hat{\mathbf{X}}_{\mathrm{L}}^{i}$ to convert the homogeneous vector to 3D coordinates, so the updated 3×1 map point coordinates are $\hat{\mathrm{s}}_{\mathrm{LC}}^{t_i}\hat{\mathbf{X}}_{\mathrm{L}}^{i}$.

3.3. System Analysis with a Public Image Dataset

In this section, the sensitivity of the proposed VOR-SLAM is evaluated with the Karlsruhe Institute of Technology and Toyota Technological Institute (KITTI) odometry benchmark [Geiger et al., 2012]. This dataset's images were obtained using a front-facing stereo camera on a car in a

Figure 3.4.: A sample image of the sequence 07 of the KITTI dataset (KITTI-07)

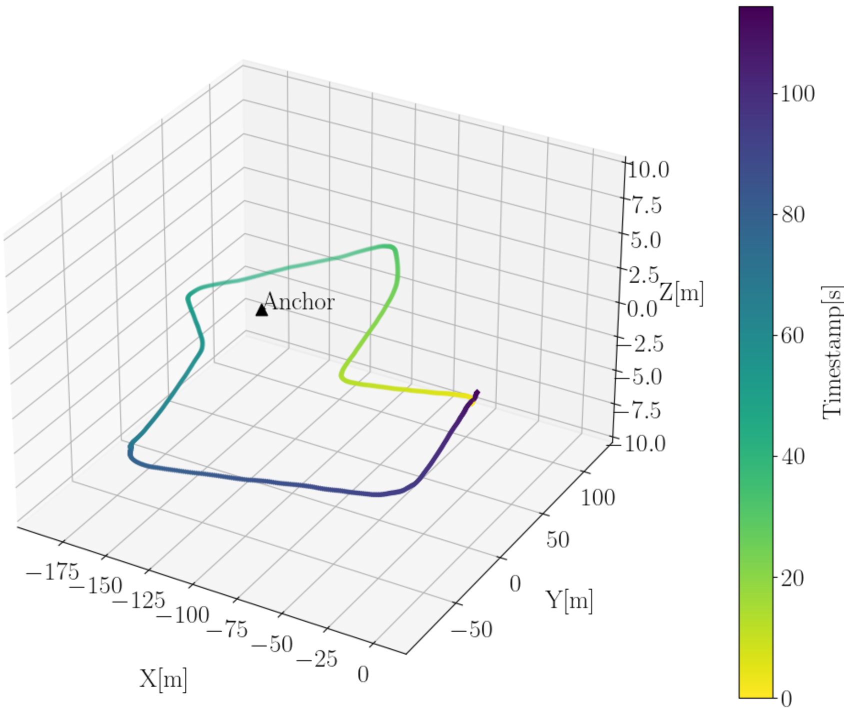


Figure 3.5.: The ground truth trajectory and the virtual anchor (KITTI-07)

Table 3.1.: The summary of the KITTI-07

Dataset	Total time [s]	Total length [m]	Dimension (X[m] × Y[m] × Z[m])
KITTI-07	114.33	694.70	191.45 × 209.35 × 4.86

residential area of Karlsruhe, Germany, and the ground truth poses (positions and orientation) of each image frame were obtained using GPS/INS. Sequence 07 (KITTI-07) was used to evaluate VOR-SLAM. A sample image of this sequence is shown in Fig. 3.4, and the total traveling time, distance, and dimension are summarized in Table 3.1. Since the KITTI dataset does not include ranges to a reference point, range measurements were synthetically generated using the ground truth trajectory and an anchor virtually set at $[-120, 5, 2]^{\mathrm{T}}$m, as illustrated in Fig. 3.5.

Additionally, the radial, tangential, and normal directions are defined as depicted in Fig. 3.6.

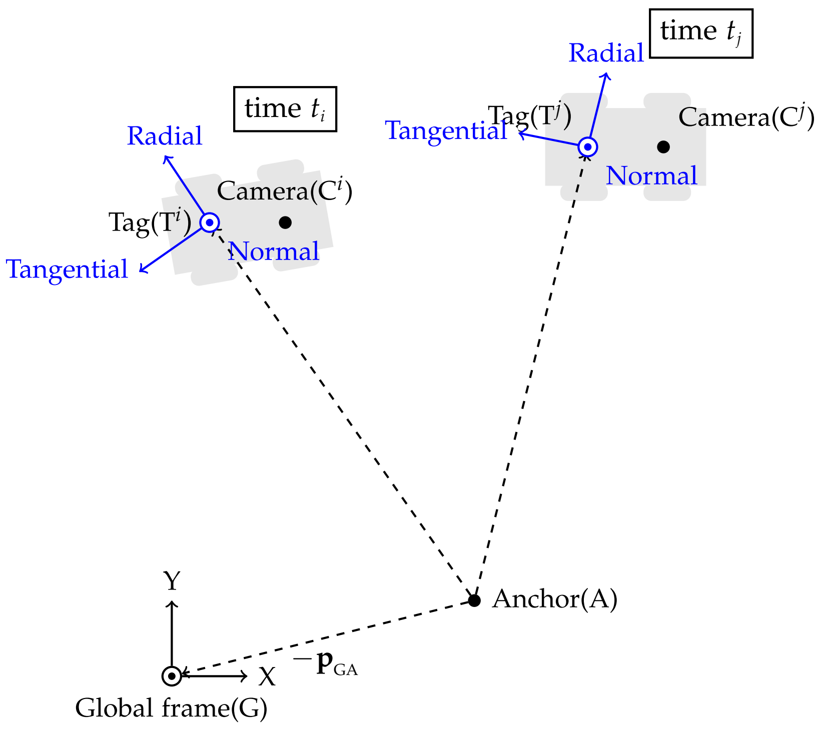


Figure 3.6.: The radial, tangential, and normal directions at t_i and t_j.

As can be seen in this figure, the radial direction is defined as the unit vector from the anchor to the tag at each timestamp:

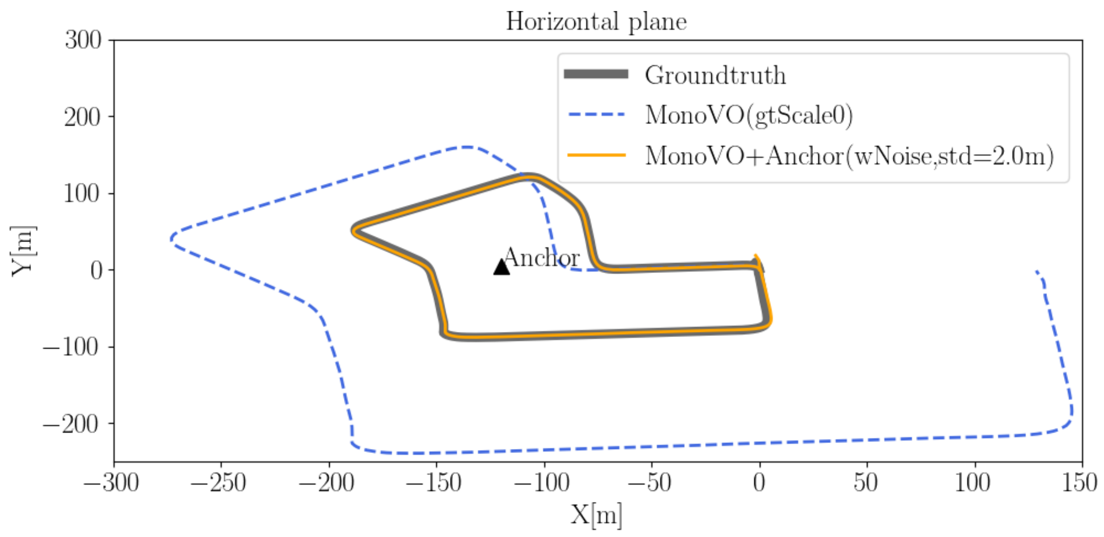
$$\mathbf{u}^t_{\text{rad}} = \frac{{}_{\text{G}}\mathbf{p}^t_{\text{TA}}}{||{}_{\text{G}}\mathbf{p}^t_{\text{TA}}||}.$$

The normal direction vector is defined as the normal vector of the plane which is determined with the radial unit vector and the anchor position vector in the global frame:

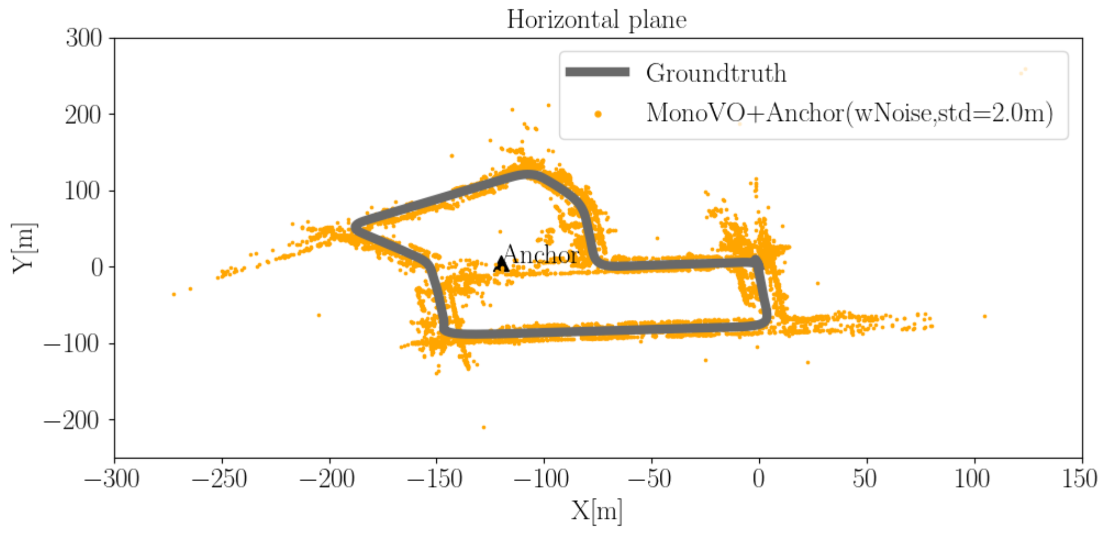
$$\mathbf{u}^t_{\text{nor}} = \frac{\mathbf{u}^t_{\text{rad}} \times \mathbf{p}_{\text{GA}}}{||\mathbf{p}_{\text{GA}}||}.$$

Lastly, the tangential direction is defined with the right-hand rule, i.e. $\mathbf{u}^t_{\tan} = \mathbf{u}^t_{\text{nor}} \times \mathbf{u}^t_{\text{rad}}$.

System sensitivity to ranging errors First, VOR-SLAM is analyzed in terms of the system sensitivity to ranging noise. For this analysis, 100×5 range sequences were generated with zero-mean Gaussian noise, using five different standard deviations, $[0.1\text{m}, 0.5\text{m}, 1.0\text{m}, 1.5\text{m}, 2.0\text{m}]$, (i.e. 100 range sequences were generated with each standard deviation). Fig. 3.7a shows the horizontal trajectory estimated with VOR-SLAM using monocular visual odometry (VO) measurements and one of the range sequences including Gaussian noise with the standard deviation of 2m (MonoVO+Anchor(wNoise,std=2m), orange). In addition, this figure shows



(a) The horizontal trajectory estimated with monocular VO and scaled with the first keyframe's ground truth scale (blue), and the trajectory estimated with VOR-SLAM (orange)



(b) The ground truth trajectory (gray), and the map points in the horizontal plane estimated with VOR-SLAM (orange)

Figure 3.7.: The horizontal trajectories estimated using monocular VO and VOR-SLAM, and the map points in the horizontal plane estimated with VOR-SLAM (KITTI-07)

the ground truth trajectory (GroundTruth, gray) and the trajectory estimated using monocular VO and scaled with the first keyframe's ground truth value (MonoVO(gtScale0), blue). This is necessary because the absolute scale is ambiguous when only a monocular camera is used to estimate camera poses. As shown in Fig. 3.7a, the positions estimated using MonoVO(gtScale0) drift over traveling time, although the trajectory estimate is scaled with the ground truth value. This drift is efficiently corrected using range measurements between the car and only one anchor even when the standard deviation of the range measurement noise is 2m. In addition, Fig. 3.8 shows the positioning errors in the radial, tangential, and normal directions (which are

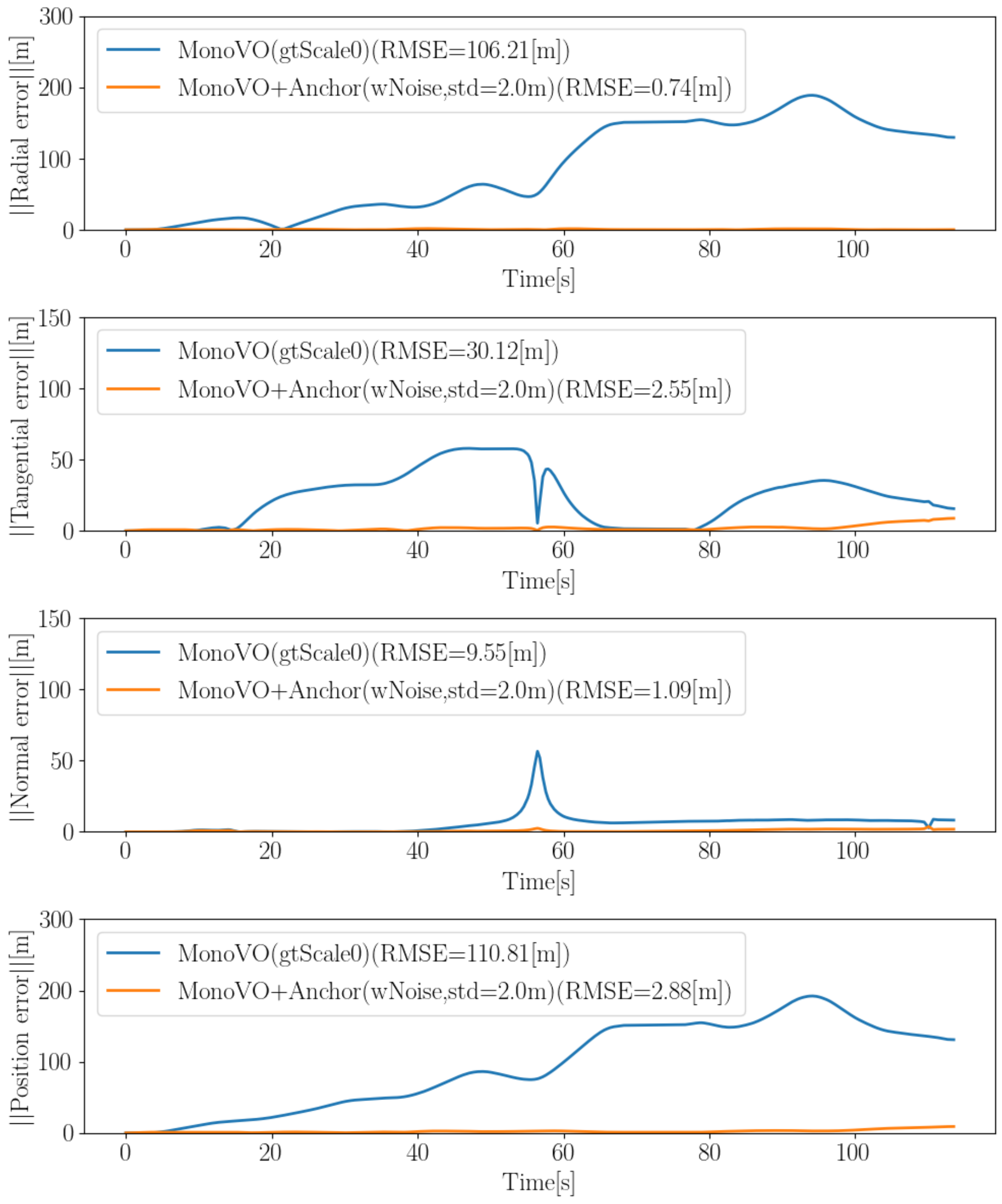
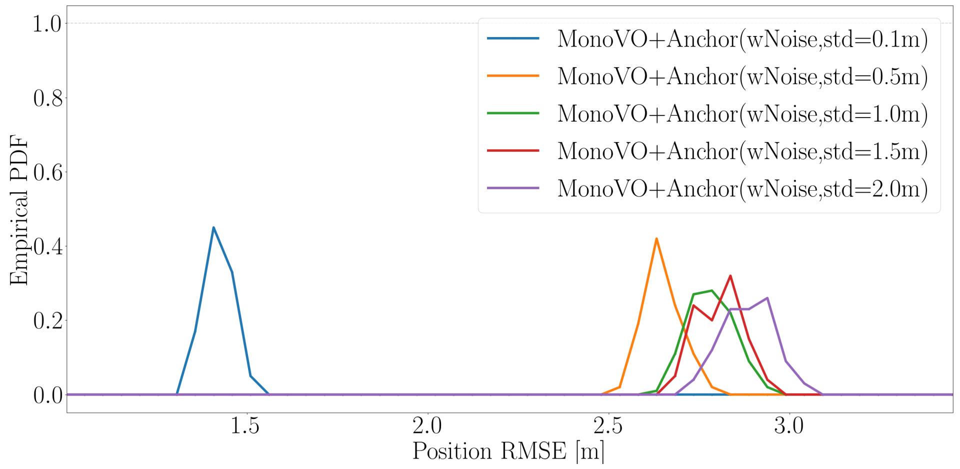
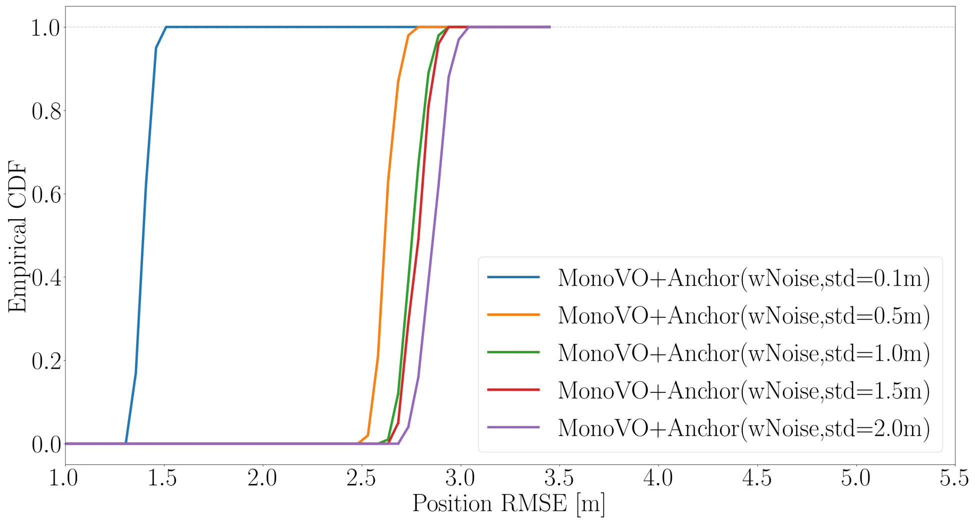


Figure 3.8.: The positioning errors in the radial, tangential, and normal directions of the positions estimated using monocular VO and scaled with the first keyframe's ground truth scale (MonoVO(gtScale0), blue), and using VOR-SLAM with ranges including Gaussian noise of 2m (MonoVO+Anchor(wNoise,std=2m), orange)

defined in Fig. 3.6) of the positions estimated using monocular VO and VOR-SLAM, as well as the magnitude of the positioning error. As can be seen in this figure, VOR-SLAM significantly reduces the positioning error in all directions. Particularly, it is most effective in reducing the radial direction error, showing 99.30% error mitigation compared to MonoVO(gtScale0). Fig. 3.7b illustrates the map points estimated with VOR-SLAM. The ground truth of the map



(a) The PDF of the position RMSE values

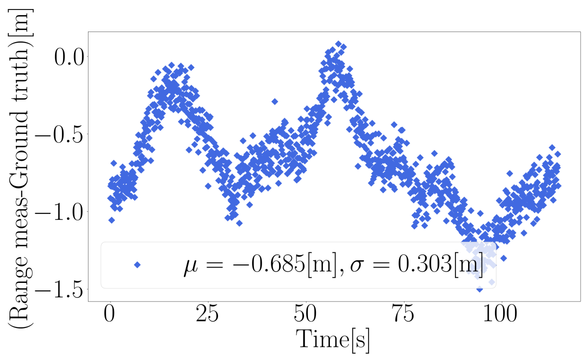


(b) The CDF of the position RMSE values

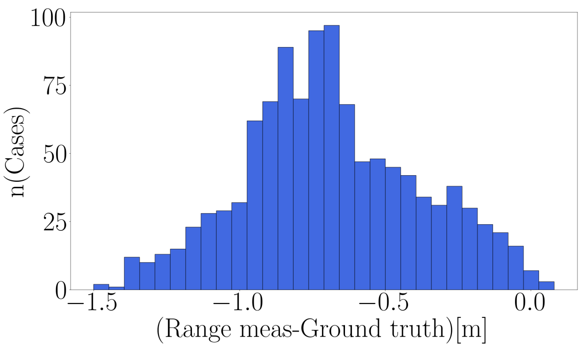
Figure 3.9.: The PDF and CDF of the position RMSE values obtained using VOR-SLAM with visual odometry measurements and range measurements including Gaussian random noise with the standard deviation of $[0.1\text{m}, 0.5\text{m}, 1.0\text{m}, 1.5\text{m}, 2.0\text{m}]$

points are unavailable, so the estimates cannot be compared to the ground truth directly. However, KITTI-07 images were obtained using a front-facing camera mounted on a car in a residential area, so map points should be created along the ground truth trajectory without scale ambiguity. As as result, the map points depicted in Fig. 3.7b can be considered accurately estimated.

The position estimation results obtained using VOR-SLAM with all 100×5 range sequences are illustrated in Fig. 3.9 as the empirical Probability Density Function (PDF) and Cumulative Distribution Function (CDF). For example, the blue line in Fig. 3.9a shows the PDF of 100 RMSE values obtained using VOR-SLAM with ranges including Gaussian noise with a standard deviation of 0.1m, while the blue line in Fig. 3.9b is the CDF of these RMSE values. When the iterative



(a) The range differences between the measurements and ground truth over time



(b) The histogram of the differences between the range measurements and ground truth

Figure 3.10.: The range differences between the ground truth and measurements including both bias and Gaussian noise

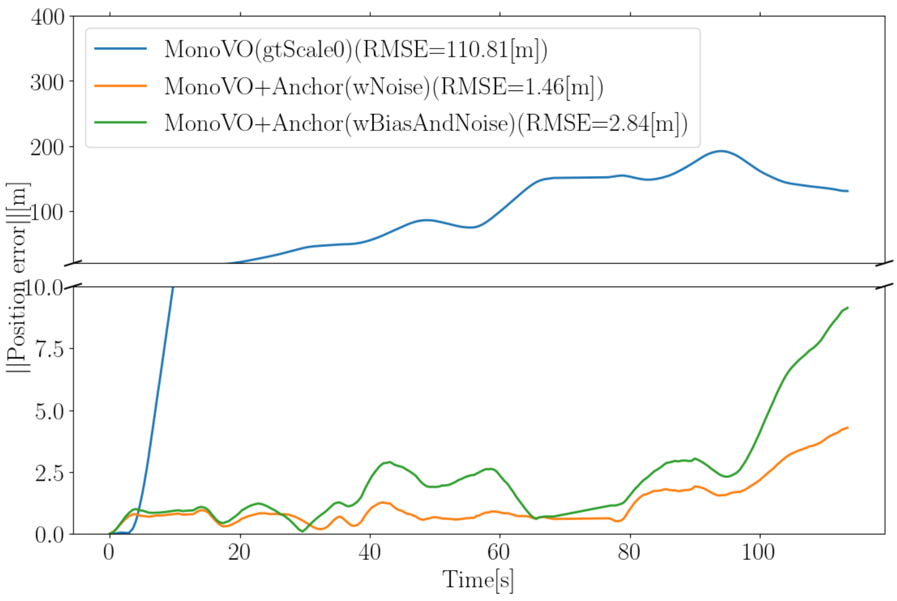
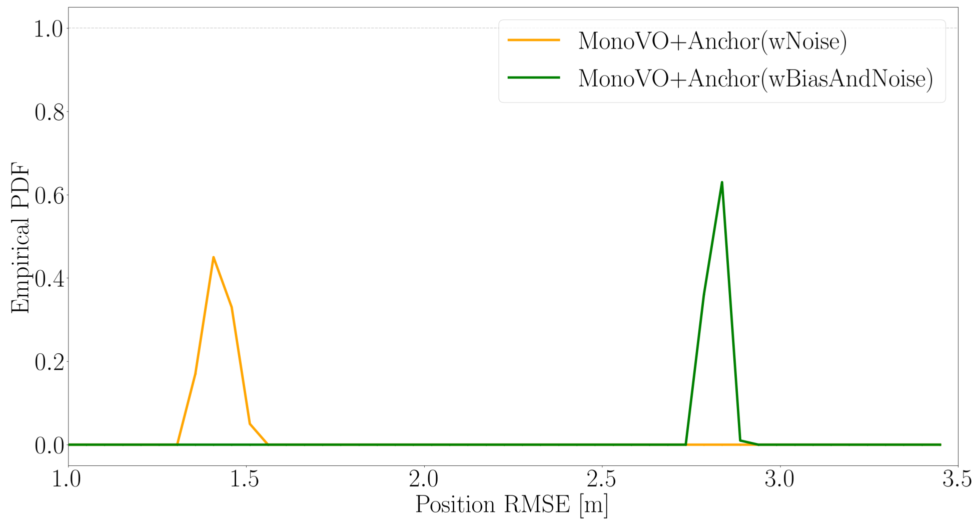


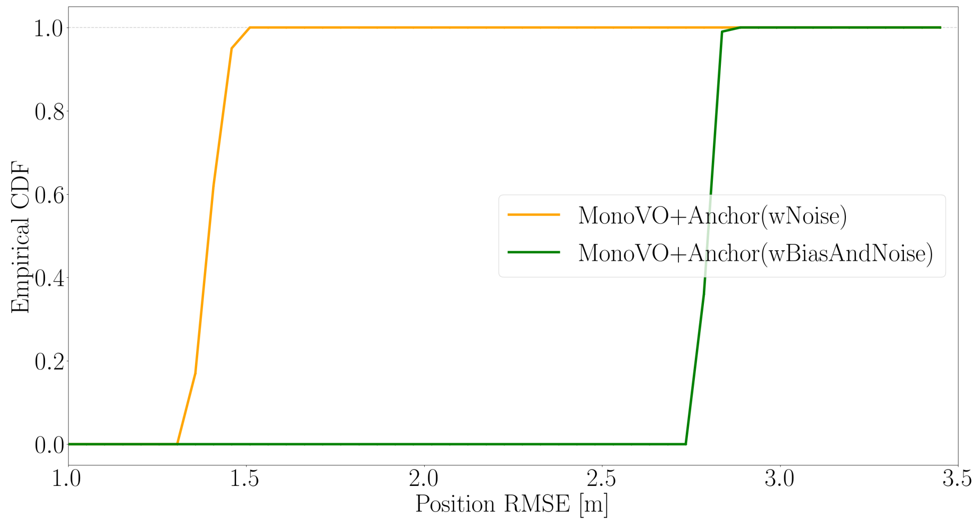
Figure 3.11.: The errors of positions estimated using monocular VO (blue), VOR-SLAM with ranges including only Gaussian noise (orange), and VOR-SLAM with ranges that are biased and include Gaussian noise (green)

LSE was executed with the cost function defined in Eq. (3.36), the covariance matrix of range measurements was set as the square of each standard deviation, $[0.1\text{m}, 0.5\text{m}, 1.0\text{m}, 1.5\text{m}, 2.0\text{m}]$. Both PDF and CDF show that the positioning is most accurate when the standard deviation of ranging noise is lowest (0.1m).

The system sensitivity to ranging bias is evaluated using range measurements including both bias and Gaussian noise. For this system evaluation, the bias was modeled as the combination of the systemic bias (deterministic values) and multipath (random values), using real UWB-based range measurements. The details of this ranging error model is described in



(a) The PDF of the position RMSE values



(b) The CDF of the position RMSE values

Figure 3.12.: The PDF and CDF of the 100 position RMSE values obtained with VOR-SLAM using ranges with Gaussian random noise (orange) and with both bias and ranges (green)

Appendix B. 100 range measurement sequences were generated including bias and Gaussian noise with the standard deviation of 0.1m. The differences between one of these measurement sequences and the ground truth ranges are depicted in Fig. 3.10. Fig. 3.11 shows the errors of the positions estimated using VOR-SLAM with one of the range sequences including only the Gaussian noise (MonoVO+Anchor(wNoise), orange), and with both bias and noise (MonoVO+Anchor(wBiasAndNoise), green). In addition, the positioning errors of monocular VO without the data fusion of range measurements are indicated in blue. As shown in this figure, the positions estimated using VOR-SLAM are more inaccurate than MonoVO+Anchor(wNoise) when range measurements are biased, but VOR-SLAM can still significantly reduce the positioning error of monocular VO even when range measurements are biased. The RMSE values of positions estimated using VOR-SLAM with all the 100 range measurements are illustrated in Fig. 3.12 as empirical PDF and CDF. This figure shows that consistent results can be obtained from all the ranging sequences.

System sensitivity to ranging availability In practice, the onboard ranging module cannot always communicate with the anchor point due to e.g. Non-Line of Sight (NLoS). To analyze the effect of the range measurement availability on the positioning accuracy and computational loads, the ranges were generated with maximum distance limitations, $[50\text{m}, 80\text{m}, 90\text{m}, 100\text{m}, 120\text{m}, 150\text{m}]$. If the true distance between the onboard ranging sensor and anchor is longer than the maximum distance, ranging is unavailable between the car and anchor point. After removing the ranges greater than the maximum values, $[50\text{m}, 80\text{m}, 90\text{m}, 100\text{m}, 120\text{m}, 150\text{m}]$, the remaining ranges are $[13.77\%, 31.16\%, 43.12\%, 60.14\%, 82.61\%, 100\%]$ of the entire traveling time, e.g. when the maximum range is set to 100m, ranges are available 60.14% of the traveling time. With each maximum distance setting, 30 range sequences including bias and noise were generated using the ranging error model described in Appendix B. Fig. 3.13 illustrates the average of the position RMSE in blue, and the average of the processing time required to compute the visual-range fusion in gray. As shown in this figure, the processing time required for the sensor fusion remains less 0.8ms for all cases, and does not increase substantially when more ranges are

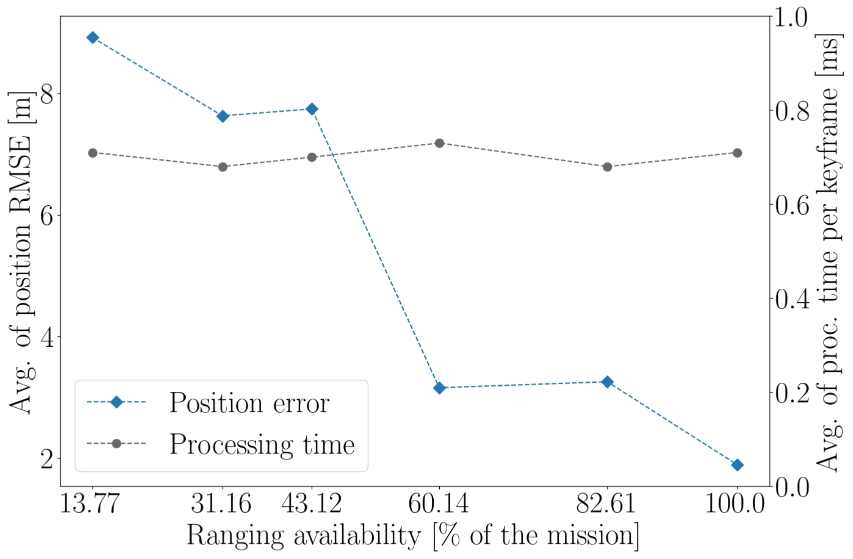


Figure 3.13.: The RMSE of the positions estimated using VOR-SLAM with the different range measurement availability (blue), and the processing time of VOR-SLAM (gray)

included in the fusion. In addition, VOR-SLAM can still significantly reduce the positioning error in the visual odometry (the RMSE of MonoVO(gtScale0)= 99.77m) even when the ranges are only available 10.72% of the frames. In addition, this figure shows that the positioning error increases significantly when the ranges are available less than 50% of the mission duration. Thus, it is crucial to acquire the ranges more than 50% of the traveling time to obtain accurate positioning with VOR-SLAM.

Comparison of VOR-SLAM and loop closures Fig. 3.14 illustrated the trajectory estimated using VOR-SLAM with ranges that include both bias and noise and available 60.14% of the traveling time (MonoVO+Anchor(wBiasAndNoise, maxRange=100m), and the trajectory estimated with the visual odometry and the loop closures (MonoVO+LC(gtScale0)). As shown in this figure, the positioning errors of the monocular visual odometry are accurately corrected with loop closures, but note that this trajectory is artificially scaled with the first keyframe's ground truth scale, but in the real world, this trajectory is still up-to-scale. Moreover, MonoVO+LC(gtScale0) shows less accurate positioning compared to MonoVO+Anchor(wBiasAndNoise, maxRange=100m) even after correcting the scale, and VOR-SLAM is conducted with the ranges only available 60.14% of the entire mission.

The scale estimation accuracy of each system is more clearly shown in Fig. 3.15. The scale

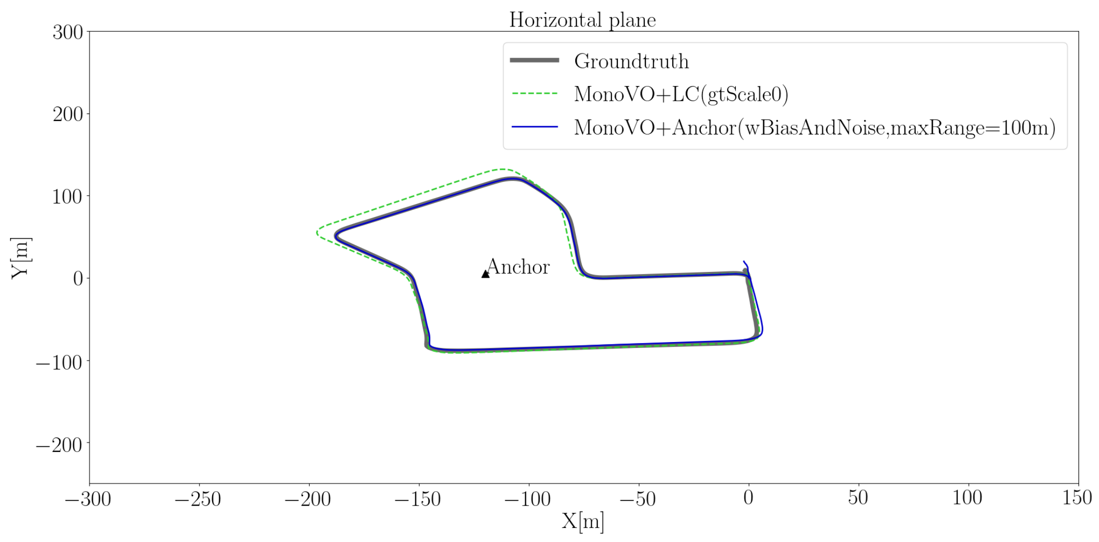


Figure 3.14.: The horizontal trajectory estimated using loop closures and scaled with the first keyframe's ground truth scale (green), and the trajectory estimated using VOR-SLAM with ranges including both bias and noise and available at 60.14% of the keyframes (blue)

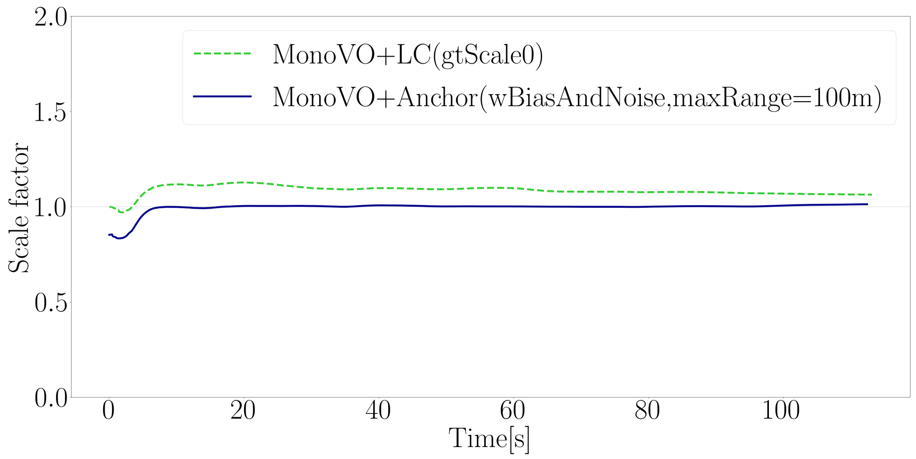


Figure 3.15.: The scale factors of the positions estimated using loop closures (green) and VOR-SLAM with ranges including both bias and noise and available at 60.14% of the keyframes (blue)

factor of the Y-axis of this figure is defined as

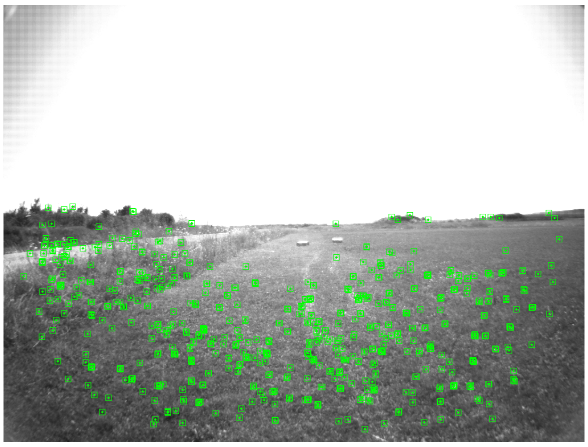
$$\text{Scale factor at } t_k = \frac{\sum_{i=0}^{i=k}\left(\mathbf{p}_{\text{est}}^{t_i} - \mathbf{p}_{\text{est}}^{t_{i-1}}\right)}{\sum_{i=0}^{i=k}\left(\mathbf{p}_{\text{gt}}^{t_i} - \mathbf{p}_{\text{gt}}^{t_{i-1}}\right)}, \tag{3.55}$$

where $\mathbf{p}_{\text{est}}^{t_i}$ and $\mathbf{p}_{\text{gt}}^{t_i}$ denotes the position estimates and the ground truth positions, respectively. If a system estimates the absolute scale perfectly, the scale factor should be 1. Loop closures can only correct the scale and positioning errors of the keyframes that observe the common map points as the loop keyframe. Thus, the absolute scale of MonoVO+LC(gtScale0) cannot be perfectly corrected over time, whereas the ambiguous scale of MonoVO is accurately estimated over time, using VOR-SLAM.

3.4. System Demonstration with Real-World Experiments

To test the proposed VOR-SLAM, outdoor experiments were collaboratively carried out with the Institute of Communications and Navigation, Germany Aerospace Center (DLR-KN). For this experiments, the rover system is developed by DLR-KN with the sensor package of the Institute for Communications and Navigation, Technical University of Munich (TUM-NAV), as shown in Fig. 3.17. As shown in Fig. 3.17a, a front-looking camera (Bumblebee2, pointgrey, 20fps) and a UWB ranging onboard module (TREK1000, Decawave/Qorvo) are mounted on the dynamic rover. To obtain the ground truth poses, GNSS Real-Time Kinematic (RTK) and Inertial Navigation Systems (INS) are used. An UWB ranging sensor on another rover staying at a fixed location is used as the reference point for ranging, as can be seen in Fig. 3.17b. With these ranging sensors, ranges between two modules can be measured every ~ 0.3s. For the data communication, a ground station is built with a Wi-Fi router and GNSS antenna as shown in Fig. 3.17c. The ground truth poses are estimated from RTK/INS with respect to the East-North-Up (ENU) frame whose origin is at the ground station's GNSS antenna.

Using this platform, real images and range measurements are obtained in two different outdoor environments: a football field of DLR, Oberpfaffenhofen, Germany, and a gravel pit in Planegg, Germany. All the images, ranges, and the ground truth poses are recorded using the Robot Operating System (ROS) as ".bag" formatted files. Fig. 3.16 shows the sample images of the datasets with the features detected in the images, and Table 3.2 presents the two datasets' total traveling time, length, and the dimension of the trajectories.


(a) Sample image of the DLR football field dataset

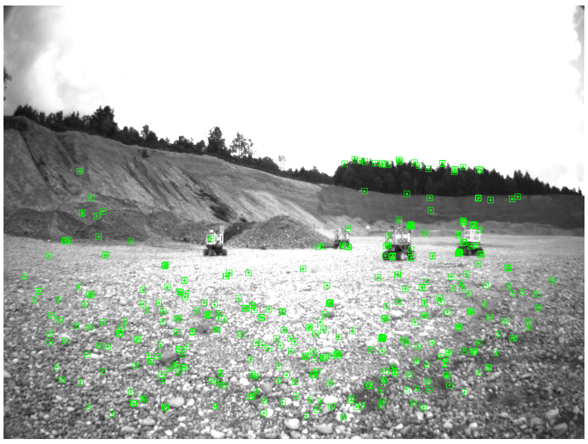
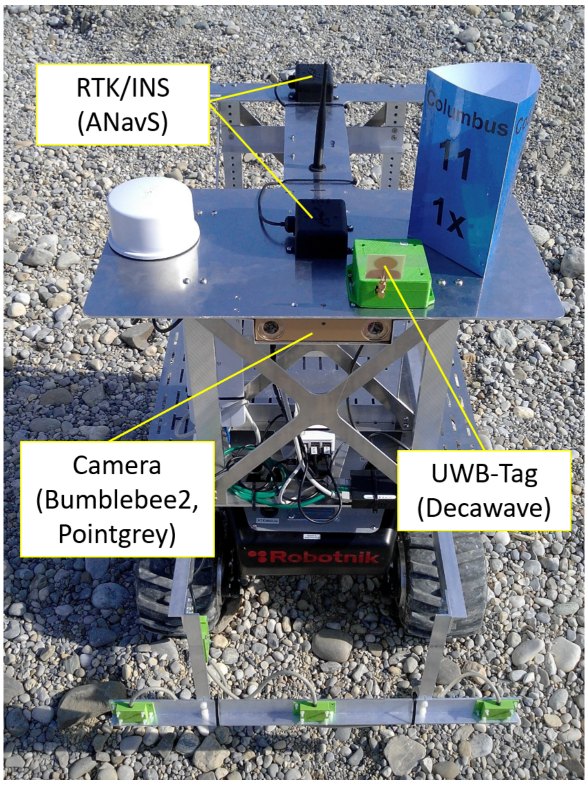

(b) Sample image of the gravel pit dataset

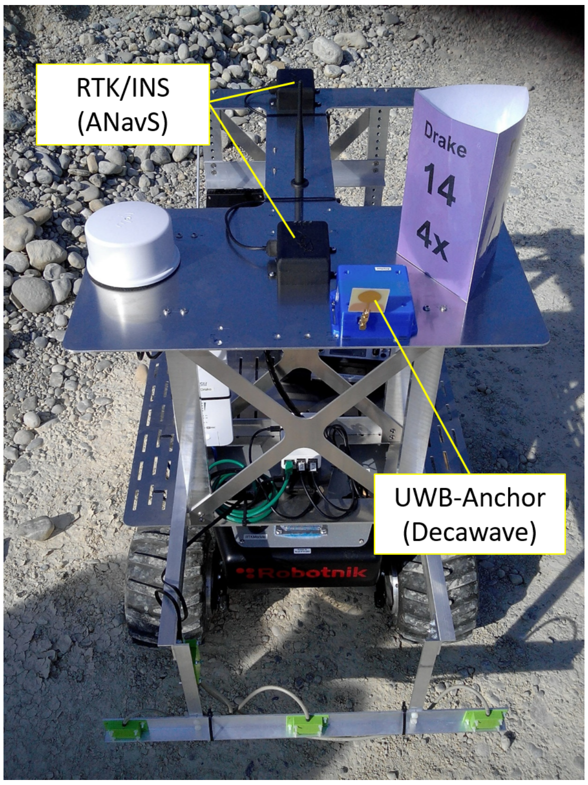
Figure 3.16.: Sample images of the DLR football field and gravel pit datasets with the features detected in the images

Table 3.2.: Summary of the DLR football field and gravel pit datasets

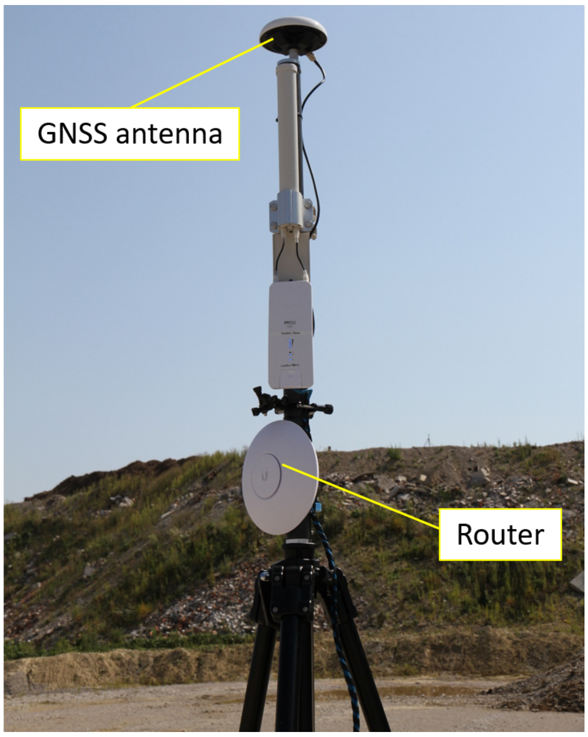
Dataset	Total time [s]	Total length [m]	Dimension (X[m] × Y[m] × Z[m])
football field	494.01	105.61	$25.54 \times 26.25 \times 0.18$
Gravel pit	457.17	38.98	$8.05 \times 9.29 \times 0.17$



(a) A dynamic rover with a camera, onboard UWB ranging sensor (tag), and multi-sensor modules for RTK/INS

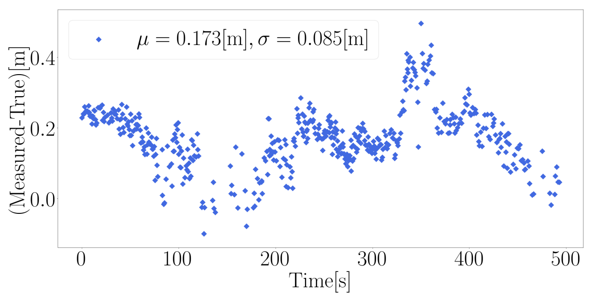


(b) A static rover with a static UWB ranging sensor (anchor) and multi-sensor modules for RTK/INS

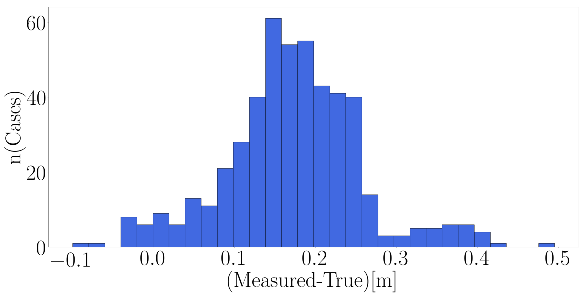


(c) A ground station for the communication system

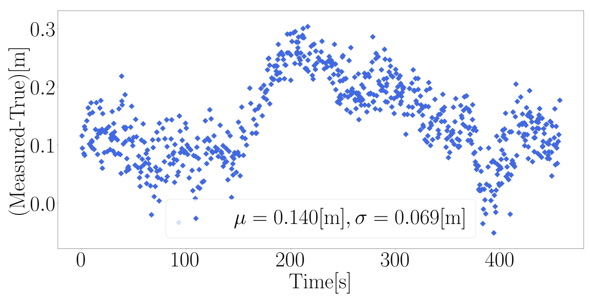
Figure 3.17.: The rover system developed by DLR-KN with the sensor package of TUM-NAV



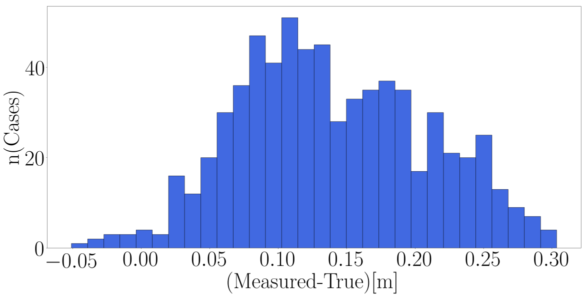
(a) The range differences over time (football field)



(b) The histogram of the range differences (football field)



(c) The range differences over time (gravel pit)

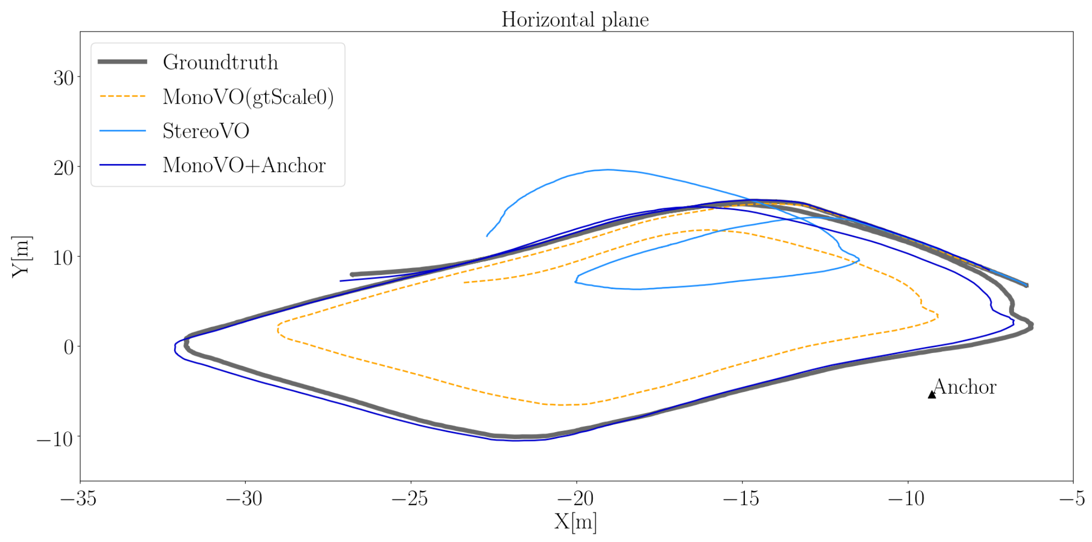


(d) The histogram of the range differences (gravel pit)

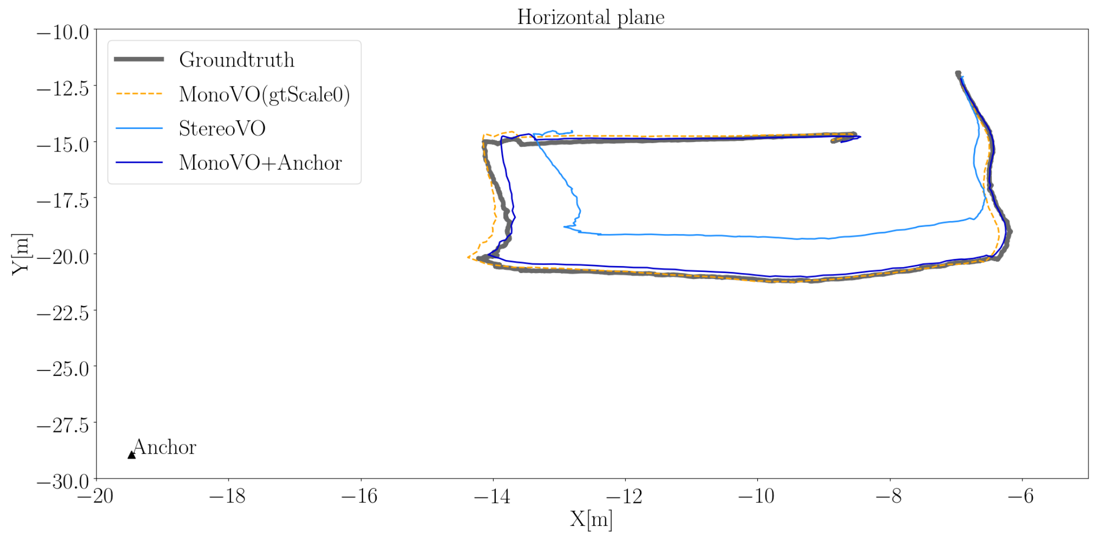
Figure 3.18.: The differences between real range measurements obtained using the UWB sensors and the ground truth ranges computed with the ground truth trajectories

The difference of the real UWB range measurements and ground truth values computed with the ground truth trajectories are shown in Fig. 3.18. Unlike the assumption of modeling in Section 3.2, real ranges are biased and not Gaussian distributed. This bias could be induced by a systematic error due to the changes of the Signal-to-Noise Ratio (SNR) and multipath [Ltd, 2014]. In Appendix B, the systematic bias is statistically modeled as a function of the angles between the LoS vector between two ranging sensors and the tag/anchor directions, and the true distances between two ranging modules. Additionally, the multipath bias is modeled as a Gaussian random walk. In Section 3.3 and Section 5.3, the synthetic range measurements of KITTI-07 are generated using this ranging error.

Accuracy of the position and absolute scale estimations Fig. 3.19 shows the trajectories estimated using monocular VO and scaled with the first keyframe's true scale (MonoVO(gtScale0), orange), using stereo visual odometry (StereoVO, sky blue), and using VOR-SLAM with visual odometry and range measurements (MonoVO+Anchor, dark blue). As shown in this figure, the rover's positions are most accurately estimated using VOR-SLAM without scale ambiguity. For the both cases, StereoVO shows the worst positioning and scale estimation accuracy because the depth of the map points could not be accurately estimated using the stereo camera with the



(a) The horizontal trajectories estimated with the DLR football field dataset

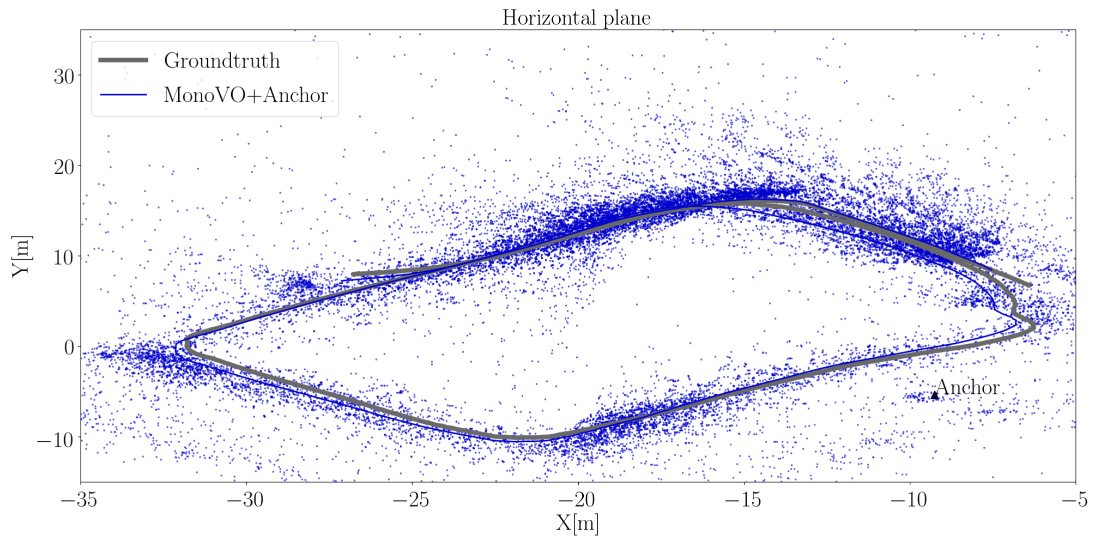


(b) The horizontal trajectories estimated with the gravel pit dataset

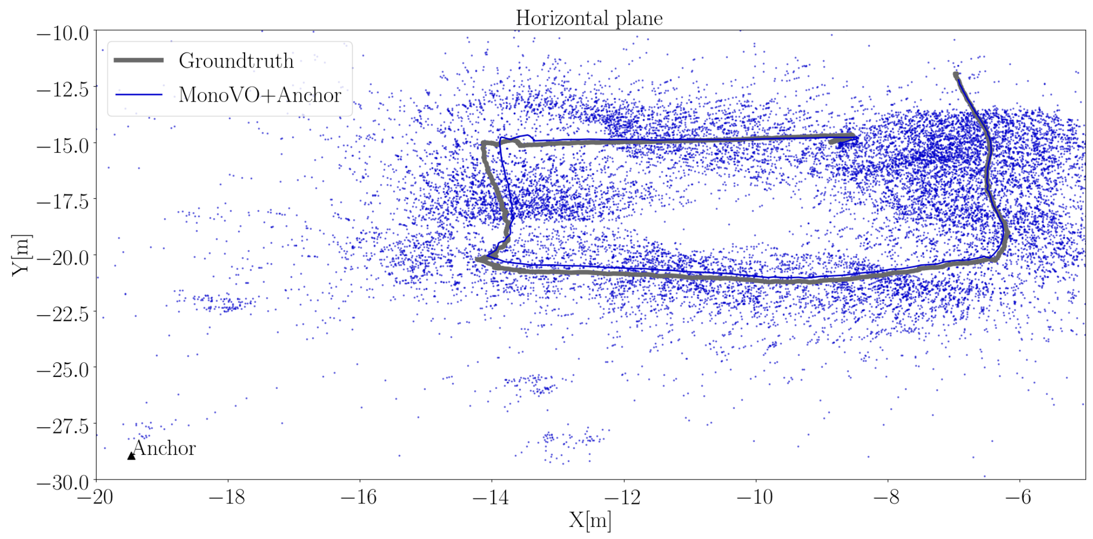
Figure 3.19.: The horizontal trajectories estimated using monocular VO and scaled with the first keyframe's ground truth scale (MonoVO(gtScale0), orange), stereo VO (StereoVO, sky blue), and VOR-SLAM (MonoVO+Anchor, dark blue)

short baseline length (12cm). Additionally, the trajectories estimated with MonoVO(gtScale0) have the similar shapes of the ground truth trajectories for both datasets, but the trajectories are still less accurate than MonoVO+Anchor, even after scaling the positions with the first keyframe's ground truth scale. Moreover, note that the absolute scale of the positions estimated using MonoVO is ambiguous in the real world.

The map points estimated using VOR-SLAM are shown in Fig. 3.20. Although no ground truth is available for the map points, considering that the images are obtained with a rover with a front-looking camera, the map points are expected to be estimated along the ground



(a) The map point estimated with the DLR football field dataset

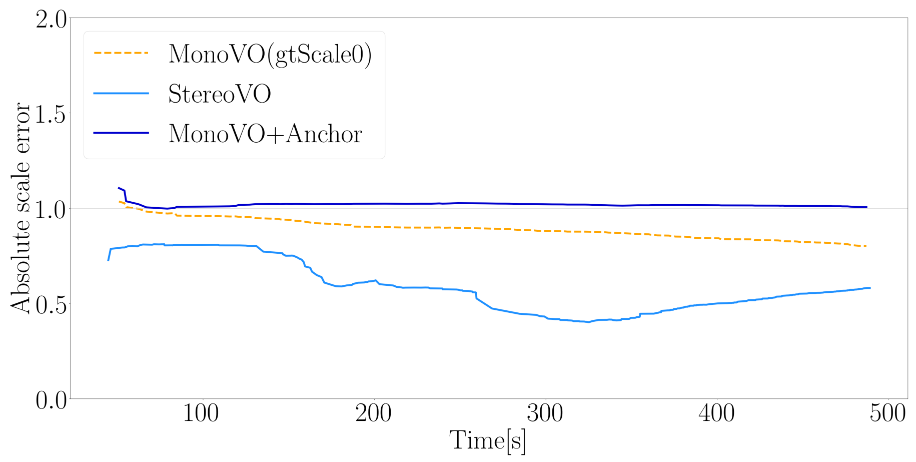


(b) The map point estimated with the gravel pit dataset

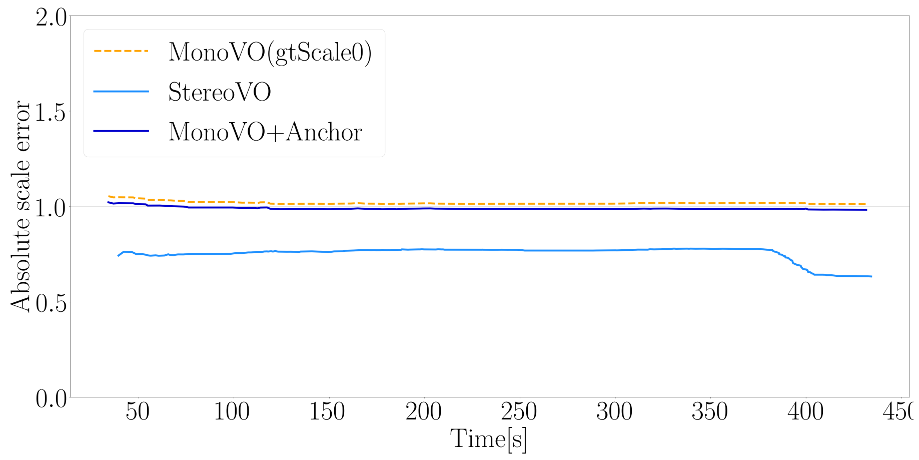
Figure 3.20.: The map points in the horizontal plane estimated using VOR-SLAM

truth trajectory. In this sense, Fig. 3.20a and Fig. 3.20b shows that the map point coordinates are also accurately estimated with the two datasets.

The scale estimation accuracy is evaluated with the scale factors defined in Eq. (3.55)). As can be seen in Fig. 3.21, StereoVO provides the most inaccurate scale estimation because of the stereo camera's short baseline length. In addition, the absolute scale drifts over time, when the trajectories are estimated using MonoVO and scaled with the ground truth scale of the first keyframe (MonoVO(gtScale0)). The scale drifts are relatively smaller for the gravel pit dataset, compared to the DLR football field dataset. However, with MonoVO(gtScale0), the scale estimation is still more inaccurate than MonoVO+Anchor.



(a) The scale factors of the DLR football field dataset



(b) The scale factors of the gravel pit dataset

Figure 3.21.: The scale factors over time, obtained with monocular VO (orange), with stereo VO (sky blue), and with VOR-SLAM (dark blue)

Comparison of VOR-SLAM and loop closures Fig. 3.22 shows the horizontal trajectory estimated with the DLR football field dataset, using monocular VO and scaled with the first keyframe's ground truth scale (MonoVO(gtScale0), orange), using monocular VO with loop closures and scaled with the first keyframe's ground truth scale (MonoVO+LC(gtScale0), green), and using VOR-SLAM (MonoVO+Anchor, dark blue). The rover travels back to the place previously visited in the DLR football field dataset, so the loop closing technique can be applied to mitigate the positioning error. Thus, the trajectory estimated using MonoVO+LC(gtScale0) seems to be the most accurate in Fig. 3.22 and Fig. 3.23. However, note that the absolute scale is still ambiguous for the MonoVO+LC(gtScale0) setup.

This scale ambiguity problem is effectively solved by VOR-SLAM with much less computing power than loop closures. The processing time per keyframe of loop closing and VOR-SLAM is measured using a laptop with Intel® Core™ i7-7700HQ CPU @ 2.80GHz $\times$ 8, Ubuntu 20.04.2

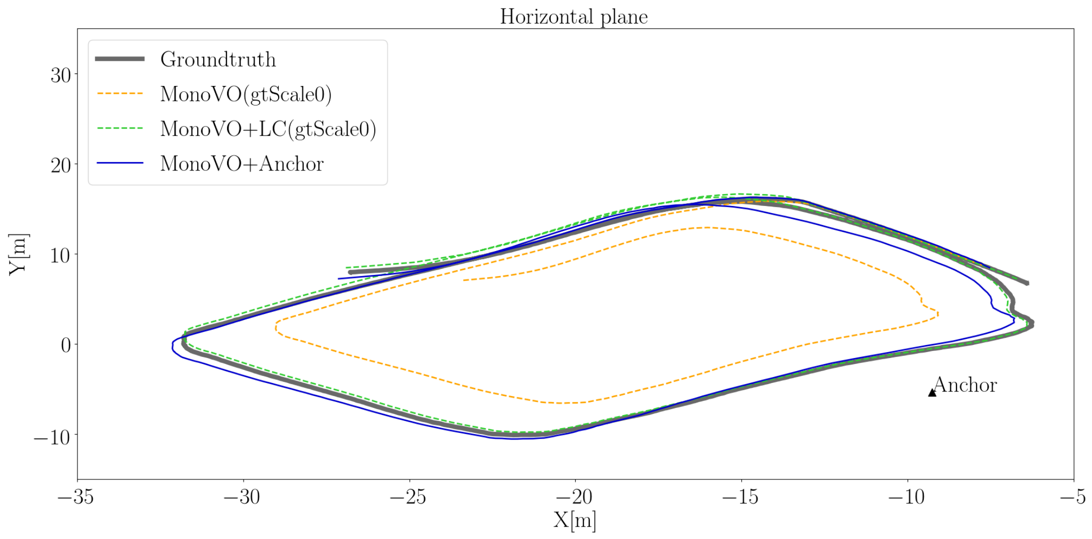


Figure 3.22.: The horizontal trajectories estimated with the DLR football field dataset, using monocular VO and scaled with the first keyframe's ground truth scale (orange), using monocular SLAM with loop closing and scaled with the first keyframe's ground truth scale (green), and using VOR-SLAM (dark blue)

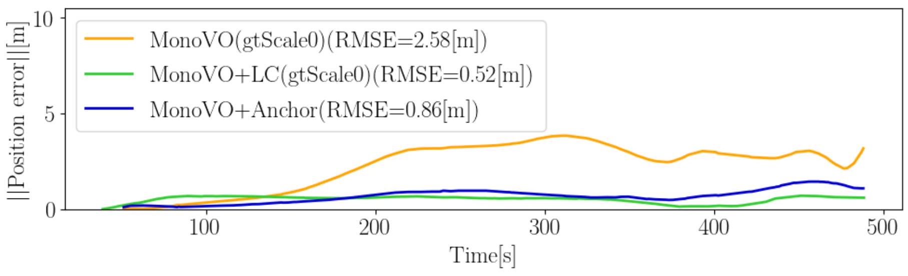


Figure 3.23.: The magnitude of the positioning errors with the DLR football field dataset, using monocular VO and scaled with the first keyframe's ground truth scale (orange), using monocular SLAM with loop closing and scaled with the first keyframe's ground truth scale (green), and using VOR-SLAM (dark blue)

LTS, and presented in Table 3.3. This table shows that the VOR-SLAM's optimization process (GraphOpt) takes $\sim$ 96% less time than the loop closing's optimization process (CorrectLoop).

Furthermore, the loop closing can successfully mitigate the errors only when a loop is detected. If no loop is detected during the mission as the gravel pit dataset, the onboard computer only consumes battery power without mitigating the positioning errors. Thus, VOR-SLAM is more useful than visual SLAM with loop closures when a robot has to operate with a non-powerful processing unit and limited battery power (e.g. micro or small drones), and it cannot go back to the previous places to detect a loop.

Lastly, the trajectories are estimated using monocular SLAM with loop closing and ranging fusion. The trajectory estimated with this combined system (MonoVO+LC+Anchor) is shown

Table 3.3.: Processing time per keyframe [ms]

	Loop closing		VOR-SLAM	
Dataset	DetectLoop	CorrectLoop	CheckFusion	GraphOpt
football field	0.088	14.730	-	0.565

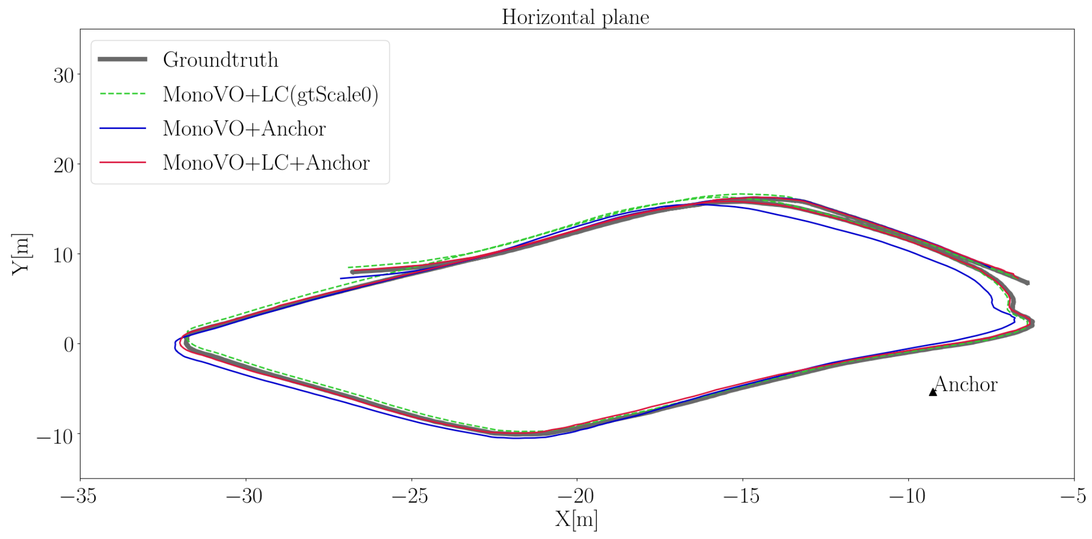


Figure 3.24.: The horizontal trajectories estimated with the DLR football field dataset, using monocular SLAM with loop closing and scaled with the first keyframe's ground truth scale (green), using VOR-SLAM (dark blue), and using monocular SLAM with loop closing and range measurements (red)

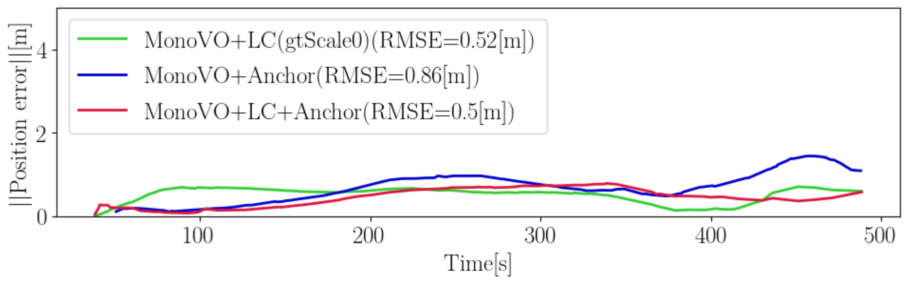


Figure 3.25.: The magnitude of the positioning errors with the DLR football field dataset, using monocular SLAM with loop closing and scaled with the first keyframe's ground truth scale (green), using VOR-SLAM (dark blue), and using monocular SLAM with loop closing and range measurements (red)

in Fig. 3.24 with the trajectories estimated using MonoVO+LC(gtScale0) and MonoVO+Anchor. Fig. 3.25 illustrates the magnitude of positioning error obtained with MonoVO+LC(gtScale0), MonoVO+Anchor, and MonoVO+LC+Anchor. As can be seen in these figures, the position RMSE is reduced to 0.5m using MonoVO+LC+Anchor, without scaling with the first keyframe's ground truth scale. This result shows that the positioning error is accurately corrected with the loop closing and the ambiguous absolute scale is properly estimated with range measurements.

4. Collaborative VSLAM - A Review

VSLAM is widely applied to estimate robots' egomotion and to map unknown environments when GNSS is unavailable or unreliable. When multiple robots cooperatively carry out given tasks, the relative poses between the robots need to be accurately estimated. To accurately estimate the relative poses, the loop closing technique can be applied using visual features commonly observed by different robots (inter-agent loop), as in [Alam and Dempster, 2013, Saeedi et al., 2016, Zou et al., 2019, Rizk et al., 2019, Chung et al., 2018]. Fig. 4.1 shows the system overview of loop closures using inter-agent map point matches. First, each robot transmits its keyframes and local map points to the processing unit. For example, Centralized Collaborative Monocular SLAM (CCM-SLAM) [Schmuck and Chli, 2019] collects the 2D feature positions in the images and their feature descriptors in keyframe messages, which are then transmitted from each agent to the processing unit. In addition, map point messages are created with the 3D coordinates of map points and the information of the keyframes, in which the map points are observed. In the map databases of different agents, the processing unit searches for inter-agent loops between different local map points (inter-agent place recognition). When enough number inter-agent loops are detected, the relative poses between the local reference frames of different agents can be estimated. Using the relative poses estimates, the local keyframes and map points are merged into one global map data (frame alignment). After aligning different reference frames, the estimation errors of all agents' keyframe poses and map point coordinates can be mitigated by pose graph optimization and Global Bundle Adjustments (GBAs) using inter-agent loop measurements (multi-agent map optimization). The keyframe poses and map point coordinates updated by the multi-agent map optimization need to be transmitted back to each agent, so the agents can exploit the up-to-date estimates for their local visual odometry processes.

As illustrated in Fig. 4.1 and Fig. 4.2a, an additional processing unit can execute all the computationally demanding processes, such as inter-agent place recognition and multi-map data optimization. A server computer [Schmuck and Chli, 2019] or a cloud server [Riazuelo

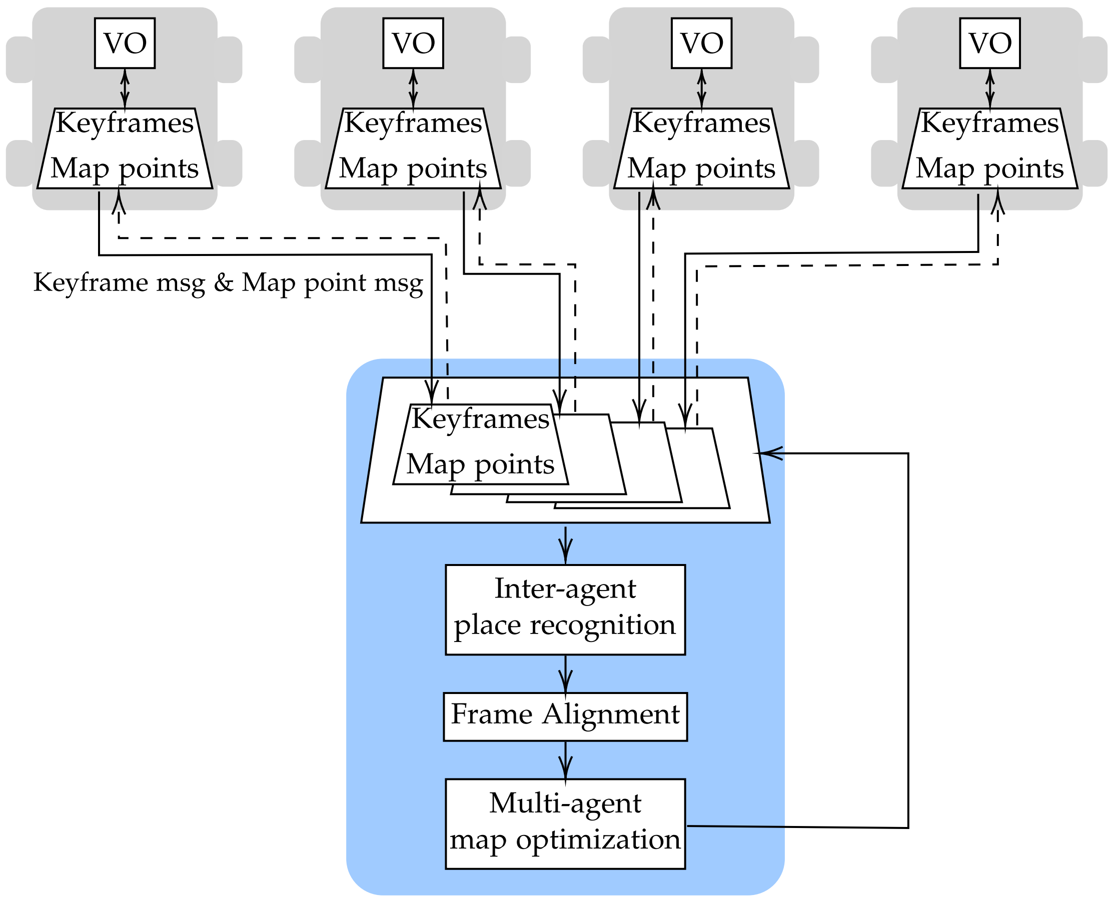
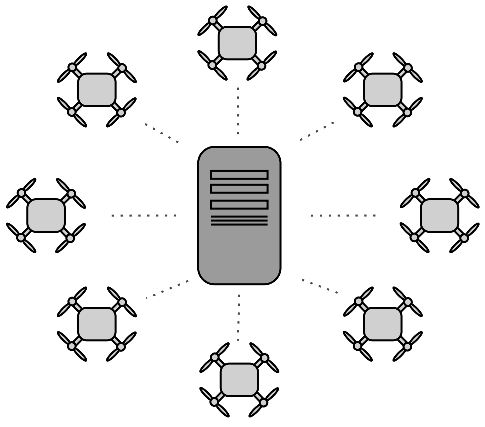


Figure 4.1.: The system overview of cooperative VSLAM with inter-agent loop closures using map points matches that are commonly observed by different agents

et al., 2014, Mohanarajah et al., 2015] can be used as the processing unit. When the central unit processes all the demanding tasks, each agent needs to execute only visual odometry. In addition, robots can discard the keyframes and map points that are unnecessary to locally execute visual odometry since the entire history of the keyframes and map points are stored on the server. Thus, onboard computers can be prevented from experiencing memory overloads. Furthermore, the global map database can be easily managed since all the data is processed at one place (server).

However, data communication networks can be easily overloaded since all the agents transmit their local data to the server. The data size of local maps can be extremely large within a short period as the agents can observe lots of map points at each timestamp. This becomes particularly problematic when many agents need to simultaneously operate in a swarm robotic system. Furthermore, agents' movement is significantly constrained because they must remain within a range that still enables communication to the central server.

As proposed in Decentralized Data Fusion-Smoothing and Mapping (DDF-SAM) [Cunning-


(a) Centralized system architecture

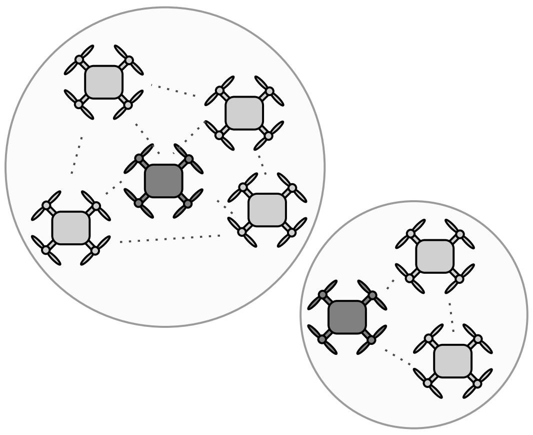

(b) Decentralized system architecture

Figure 4.2.: Centralized and decentralized system architectures of cooperative VSLAM

ham et al., 2010, Cunningham et al., 2013] and Decentralized SLAM (dSLAM) [Cieslewski et al., 2018], the computationally demanding tasks of inter-agent loop closing can be carried out in a decentralized manner, instead of using a central server. In this decentralized system architecture, one of the robots conducts inter-agent loop closures using the local map data transmitted from neighboring robots, as seen in Fig. 4.2b. Since only the neighboring robots' map data is processed, inter-agent loop closing and global map fusion require much less computing power than centralized systems. In addition, robots can explore with much fewer constraints because they do not always need to be connected to the central server. However, the global map database's consistency cannot be easily maintained if it is processed in a decentralized manner. Data synchronizations and global map fusion between different neighborhoods become more complicated and challenging than a centralized system since map data is processed on multiple local servers.

Instead of inter-agent loops, swarm robots can use other measurements to accurately estimate the relative poses and create a global map. For example, robots can mitigate the estimation errors when they observe a checkerboard (or other patterns known to users) on the other robots [Richardson et al., 2013, Kim et al., 2010] or when they observe the other robots directly [Zhou and Roumeliotis, 2006]. However, this chapter focuses on reviewing collaborative VSLAM with inter-agent loop closing since it is the most widely studied and tested method at the system level. In the following sections, each system process (shown in Fig. 4.1) is detailed. First, the inter-agent image classification and place recognition method are explained, followed by the inter-agent map point matching and frame alignment methods. Then, multi-agent map fusion and optimization strategies are reviewed. Finally, the communication requirements of the methods reviewed are evaluated. The evaluation results show that collaborative VSLAM using inter-agent loop closures is impractical when swarm robots explore areas where the communication capabilities are significantly limited, e.g. in an extraterrestrial area or deep sea.

4.1. Inter-Agent Place Recognition

For collaborative VSLAM with inter-agent loop closures, map points matches need to be searched between different robots' map databases. Fig. 4.3 shows the overview of an efficient inter-agent loop searching method with the Bag of Words (BoW) approach (introduced in Fig. 2.5). First, each image's visual vector is computed using the feature descriptors of the image and visual vocabularies. By examining the distances of different visual vectors, images can be classified into the subgroups, i.e. if the distance between two images' visual vectors is smaller than the criteria, they are classified in the same class. Feature points are compared only between the images that are in the same class, using their descriptors. This process requires much less computing power compared to brute-force loop searches using all the images without a prior image classification.

To minimize the onboard computers' computational loads, each robot only extracts ORB features and descriptors [Rublee et al., 2011], and transmits them with keyframe messages to the central server, in CCM-SLAM [Schmuck and Chli, 2017, Schmuck and Chli, 2019]. In addition, the robots transmit map point messages (including e.g. the 3D coordinates of the local map points) to the server. The server converts all the images to visual vectors using the BoW approach with the feature descriptors, as proposed in [Gálvez-López and Tardos, 2012]. Since CCM-SLAM transmits full feature descriptors from multiple agents to the server, the communication network system can be easily overloaded. To avoid this problem, the keyframe and map point messages are divided into two types: new and update messages. The constant information of the messages (e.g. feature descriptors) is only sent once in the new keyframe or map point messages from the agents to the server. When a keyframe or map point only needs to be updated (i.e. when the keyframe or map point was sent before), the constant information is discarded from the update messages.

For cooperative VSLAM in a decentralized manner, dSLAM [Cieslewski et al., 2018] proposes to compute the visual vectors on each robot's onboard computer, and transmits them to the main robot in the robot's neighborhood group. To convert the images to the visual vectors, the neutral Network Vector of Locally Aggregated Descriptors (NetVLAD) [Arandjelovic et al.,

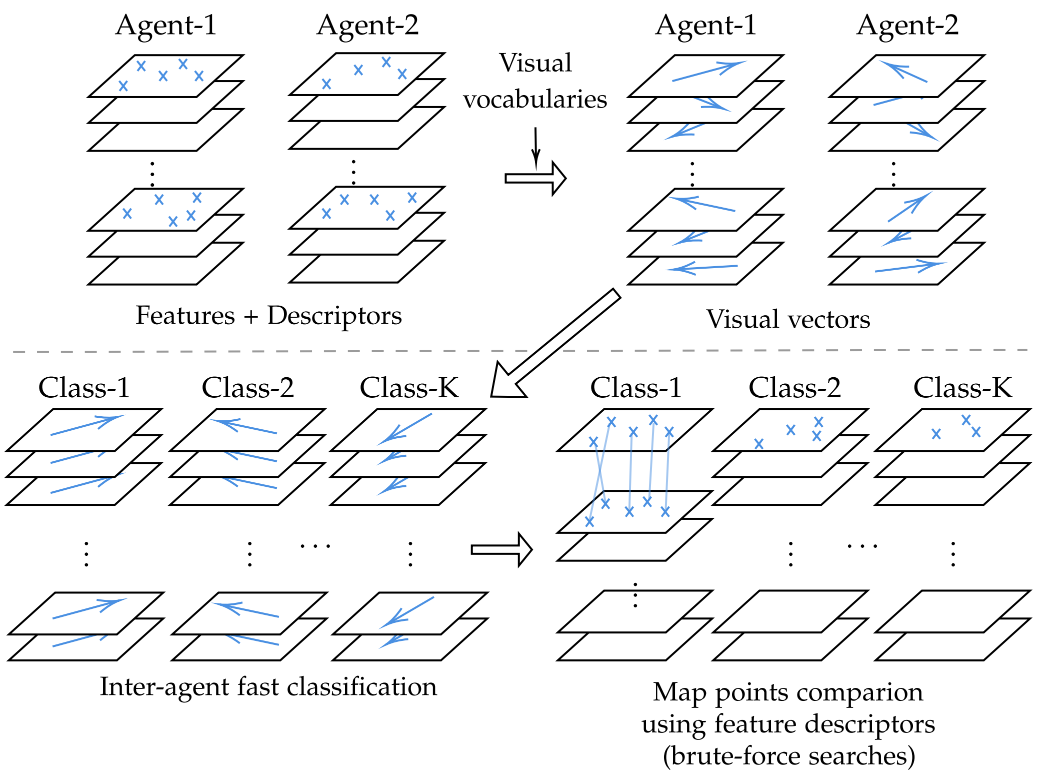


Figure 4.3.: The overview of an efficient inter-agent loop detection method using inter-agent image fast classification

2016] is used. Using NetVLAD with K clusters, an image with the D-dimensional feature descriptors (e.g. $D = 32$ for an ORB descriptor) is converted to a $(D \times K) \times 1$ visual vector. Note that the dimension of the visual vectors is dependent on the feature descriptors' dimension and the number of the clusters, and is not increased with the number of the features in the image.

After classifying the images, the map point matches are searched between the images that belongs to the same class. To reduce the computing power for this search, features' visual identifiers [Tardioli et al., 2015] can be exploited, instead of using the feature descriptors. Visual identifier is a 2×1 pointer of visual vocabularies, and it contains the identification numbers of the visual words associated to descriptors. Since similar descriptors are associated with the same visual word, the same visual identifier is allocated to these descriptors. Thus, visual identifiers can be used for feature matching. Using visual identifiers, feature points can be compared with much less computing power than the feature comparison using descriptors. However, feature matching can be more inaccurate than the feature matching with descriptors.

4.2. Local Frame Alignment and Multi-Agent Map Fusion

Each robot uses its own initial camera pose as the local reference frame, when it estimates the keyframe poses and map points using VSLAM. Thus, the relative poses between the local reference frames need to be estimated before merging the local map databases transmitted from different robots. The relative pose between two robots can be estimated as a 7DoF similarity matrix by the closed form solution, Horn's method [Horn, 1987], using map point matches commonly observed by two robots. Alternatively, dSLAM [Cieslewski et al., 2018] estimates the relative pose between two local reference frames, L_1 and L_2, by searching the relative rotation matrix $\hat{\mathbf{R}}_{\mathrm{L_1L_2}}$, translation vector $\hat{\mathbf{t}}_{\mathrm{L_1L_2}}$, and scale $\hat{\mathrm{s}}_{\mathrm{L_1L_2}}$ that minimize the sum of the squared errors of map point coordinates with respect to L_1 (i.e. $\mathbf{X}^i_{\mathrm{L_1}}$) and the coordinates converted from the matching map point with respect to L_2 to L_1 using the relative pose (i.e. $\mathrm{s}_{\mathrm{L_1L_2}}(\mathbf{R}_{\mathrm{L_1L_2}}\mathbf{X}^i_{\mathrm{L_2}} + \mathbf{t}_{\mathrm{L_2L_1}})$), as shown in Fig. 4.4:

$$\hat{\mathbf{R}}_{\mathrm{L_1L_2}}, \hat{\mathbf{t}}_{\mathrm{L_2L_1}}, \hat{\mathrm{s}}_{\mathrm{L_1L_2}} = \underset{\mathbf{R}_{\mathrm{L_1L_2}}, \mathbf{t}_{\mathrm{L_2L_1}}, \mathrm{s}_{\mathrm{L_1L_2}}}{\operatorname{argmin}} \sum_i ||\mathbf{X}^i_{\mathrm{L_1}} - \mathrm{s}_{\mathrm{L_1L_2}}(\mathbf{R}_{\mathrm{L_1L_2}}\mathbf{X}^i_{\mathrm{L_2}} + \mathbf{t}_{\mathrm{L_2L_1}})||^2. \tag{4.56}$$

After successfully estimating the relative pose between the local reference frames, a pose graph is created including the keyframe poses as the state variables. The keyframe poses are defined as 7DoF similarity matrices to consider the scale difference between different agents:

$$\boldsymbol{\Theta} = \{\mathbf{S}^{t_i}_{\mathrm{L_kC_k}} | \forall \text{ agent ID } k, \text{ and } \forall \text{ timestamp } t_i\}.$$

These keyframe poses are depicted as the circular nodes in Fig. 4.5a. As illustrated with the black dashed lines, visual odometry measurements connect the circular nodes. Additionally, the relative pose estimate between two reference frames is added as a measurement between the keyframes of different agents (blue line). The optimal keyframe poses can be estimated, by searching the poses that minimize the sum of the squared error of the visual odometry measurements and loop measurements (all the measurements errors shown in the pose graph), as proposed in CCM-SLAM [Schmuck and Chli, 2017, Schmuck and Chli, 2019].

After the pose graph optimization, the 3D coordinates of map points $\tilde{\mathbf{X}}^i_{\mathrm{L_k}}$ are updated using the differences between the keyframe poses before the optimization $\mathbf{T}^{t_i}_{\mathrm{L_kC_k}}$ and after the optimization $\hat{\mathbf{S}}^{t_i}_{\mathrm{L_kC_k}}$:

$$\hat{\tilde{\mathbf{X}}}^j_{\mathrm{L_k}} = \left(\hat{\mathbf{S}}^{t_i}_{\mathrm{L_kC_k}} \left(\mathbf{T}^{t_i}_{\mathrm{L_kC_k}}\right)^{-1}\right) \tilde{\mathbf{X}}^j_{\mathrm{L_k}}$$

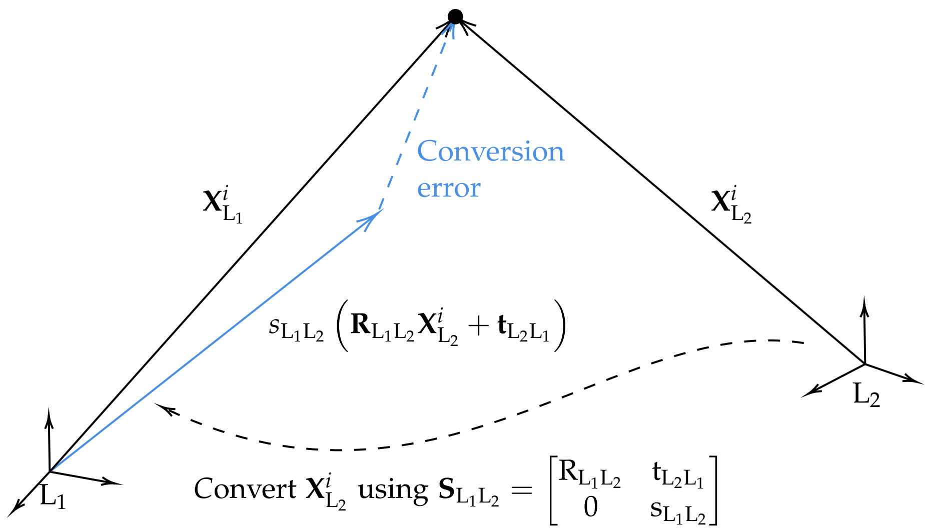
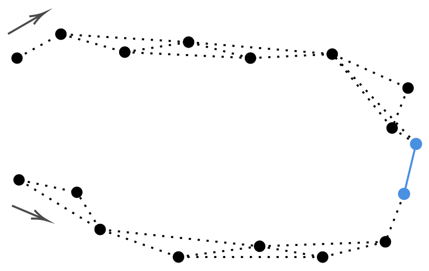
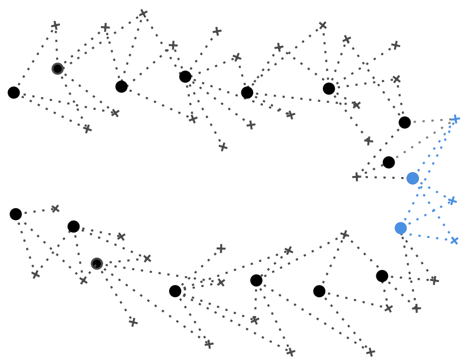

$\mathbf{X}^i_{L_1}$
Conversion error
$\mathbf{X}^i_{L_2}$
$s_{L_1L_2}\left(\mathbf{R}_{L_1L_2}\mathbf{X}^i_{L_2} + \mathbf{t}_{L_2L_1}\right)$
L_1
L_2
Convert $\mathbf{X}^i_{L_2}$ using $\mathbf{S}_{L_1L_2} = \begin{bmatrix} R_{L_1L_2} & t_{L_2L_1} \\ 0 & s_{L_1L_2} \end{bmatrix}$

Figure 4.4.: The error between the map point coordinates with respect to L_1 ($\mathbf{X}^i_{L_1}$) and the coordinates converted from the matching map point with respect to the L_2 using the relative similarity pose ($s_{L_1L_2}(\mathbf{R}_{L_1L_2}\mathbf{X}^i_{L_2} + \mathbf{t}_{L_2L_1})$)



(a) Pose graph with two agents' local poses (black circles) with the odometry measurements (black dashed lines) and loop measurement (blue solid line)



(b) Graph with two agents' local poses (black circles) and map points (black crosses)

Figure 4.5.: Graphs for the multi-agent map fusion and optimization

Since $\hat{\tilde{\mathbf{X}}}^j_L$ are 4×1 homogeneous vectors including the scale factor $\hat{s}^{t_i}_{L_kC_k}$ in the fourth element, they are converted to 3×1 vectors $\hat{\mathbf{X}}^j_L$, by multiplying the first three elements with the scale factor $\hat{s}^{t_i}_{L_kC_k}$.

To achieve better accuracy of the keyframe poses and map points, GBAs can be executed using a factor graph-based method after updating the local poses and map points separately. As illustrated in Fig. 4.5b, the factor graph includes both the 7DoF keyframe poses (black circular nodes) and the map points (black crosses nodes) as the state variables. The feature observations of the map points in the keyframe images connect the keyframes and map points nodes (black dashed lines). Between the keyframes and map points nodes of different agents, inter-agent loops (the map points commonly observed by the both agents) are added as illustrated with the blue dashed lines in Fig. 4.5b. Using this factor graph, the keyframe poses and map point coordinates can be optimized, by searching them minimizing the summation of the squared re-projection errors of all the feature observations added into the graph.

After executing GBAs, the processing unit needs to transmit the updated data back to each agent regularly. Each agent can estimate its egomition using VSLAM, but local estimation errors are accumulated over time, if it does not regularly receive the feedback from the processing unit. Moreover, the local data saved on the robots and global map data stored at the processing unit becomes inconsistent when the data is transmitted only from the robots to the processing unit. This data inconsistency could induce a significant problem for the further map fusion, so the processing unit should transmit the up-to-date data back to the robots regularly. Moreover, the Optimistic Concurrency Control (OCC) approach can be applied to main the data consistency, as proposed in [Cieslewski et al., 2015, Gadd and Newman, 2016].

4.3. Analysis of the Communication Requirement

In this section, two representative VSLAM methods with inter-agent loop closures are analyzed in terms of their communication and computational requirements. First, CCM-SLAM [Schmuck and Chli, 2019] with a centralized system architecture transmits both the keyframe and map point messages to the server with a fixed data rate. To avoid network overload, a fixed number of the keyframes are included in the keyframe messages, and the map point messages include only the map points observed in these keyframes. To further reduce the data size, the messages are sorted into two groups: new" and update messages. The data that is not changed in the map fusion (e.g. feature points' 2D locations and their descriptors) is only included in the new messages once, then discarded when the keyframes or map points only need to be updated later.

To implement inter-agent loop closing as a decentralized manner, dSLAM [Cieslewski et al., 2018] was devised to further reduce the communication loads. For example, instead of transmitting the full descriptors of the feature points, each robot computes the images' visual vectors locally, and then only transmits the visual vectors and visual identifiers to the neighboring robots.

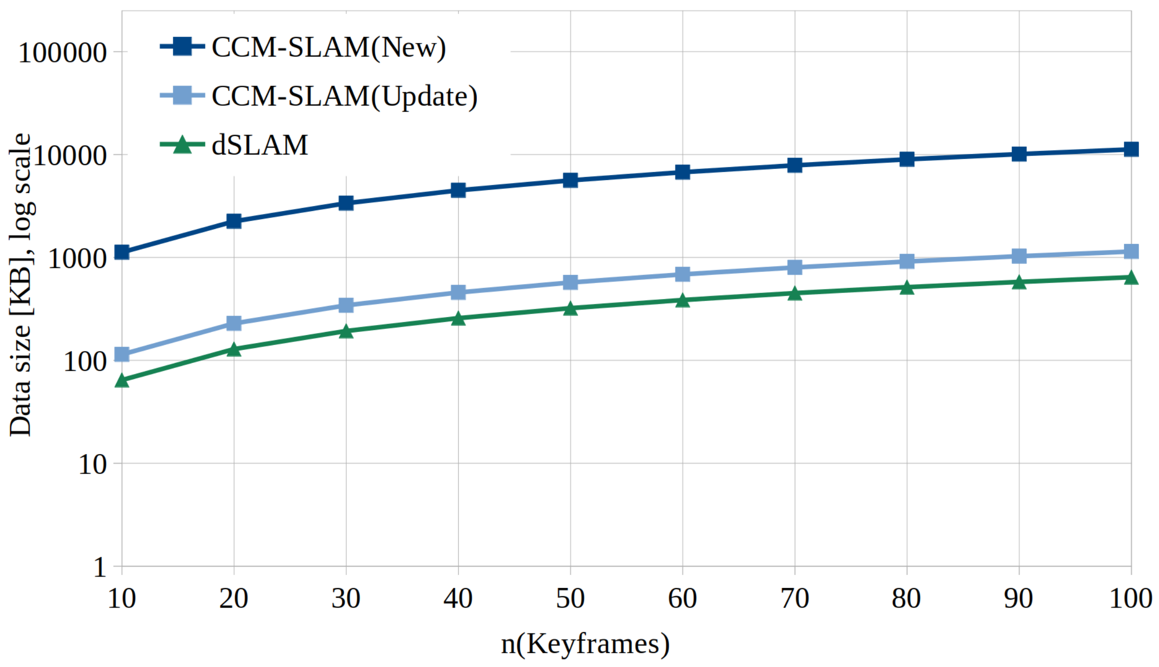


Figure 4.6.: The size of data transmitted with various numbers of keyframes (on a log scale)

To compared the communication loads of CCM-SLAM and dSLAM, Fig. 4.6 is illustrated. This figure shows the size of the data transmitted with the various numbers of keyframes, on a log scale. To compute the data size, the followings are assumed:

- 1000 map points are observed in each keyframe
- The total number of the map points is 1000 × (The total number of the keyframes) × 0.3; Only 30% are identical map points

The second one is assumed because in practice, the same map points are observed in consecutive keyframes, so not all the 1000 × (The total number of the keyframes) can be identical map points. As shown in Fig. 4.6, CCM-SLAM requires the largest communication loads when robots transmit new keyframe and map point messages (CCM-SLAM(New), dark blue). By discarding the data that does not change, CCM-SLAM (CCM-SLAM(Update), sky blue) requires to transmit much reduced data compared to the CCM-SLAM(New). As illustrated with the green line, dSLAM requires to send the smallest data to execute multi-map fusion and optimization (44% reduced data from CCM-SLAM(Update)).

However, dSLAM could be non-applicable for swarm robotic systems operating in areas where the communication and computation capabilities are highly limited, such as deep sea and extraterrestrial environments. In addition, both CCM-SLAM and dSLAM cannot be used when different robots cannot observe common map points (inter-agent loops). Improving the previous works of The Institute for Communications and Navigation, Technical University of Munich (TUM-NAV) [Zhu, 2019, Lee et al., 2020a], collaborative VOR-SLAM is proposed in the next chapter. Using this method, swarm robots' poses and map points can be accurately estimated without searching for a inter-agent loop. Furthermore, it requires much smaller communication and computational capabilities compared to cooperative VSLAM using inter-agent loop closures.

5. CoVOR-SLAM: Cooperative VOR-SLAM for Multi-Robot Systems

Collaborative VSLAM using the map points commonly observed by multiple robots (inter-agent loop) is a useful tool to localize swarm robots and create a global map of an unknown environment. However, robots need to exchange sizeable visual data (e.g. feature descriptors) to detect inter-agent loops. Moreover, substantial computing power is required to fuse multiple agents' map databases. Thus, it is challenging to run inter-agent loop closures when the computational and communication capabilities are highly limited.

Inter-agent ranges can be used to run collaborative SLAM in such environments. Instead of searching loops between the agents, inter-agent range measurements are used to mitigate the errors of the keyframe poses and map point coordinates. Thus, this approach requires much fewer communication loads compared to inter-agent loop closures. In addition, range measurements can be obtained using the communication links between the robots, without installing additional onboard hardware. Alternatively, low-cost ranging sensors, such as UWB sensors, can be also used to obtain range measurements between robots. In [Paull et al., 2015], a data fusion of odometry and inter-agent ranges was proposed for submarines exploring underwater where adequate computational and communication capabilities are unavailable. A similar algorithm was tested in [Strader et al., 2016] with UAVs formation-flying in 2D. In [Xu et al., 2020], visual, inertial, and inter-agent range measurements were fused to cooperatively estimate multiple UAVs' 3D poses, and this algorithm was experimentally evaluated with five UAVs and UWB ranging sensors. Ziegler et al. [Ziegler et al., 2021] proposed a similar algorithm, and tested it in a simulation with 49 drones.

This chapter proposes cooperative VSLAM using visual odometry and range measurements for swarm robotic systems (CoVOR-SLAM). Fig. 5.1 shows the system overview of CoVOR-SLAM using the two-agent setup. Each robot estimates the local poses and map points using VO or VSLAM with respect to its local reference frame coinciding with the initial camera pose. The local poses and map points are stored in the keyframes and map points, respectively. In

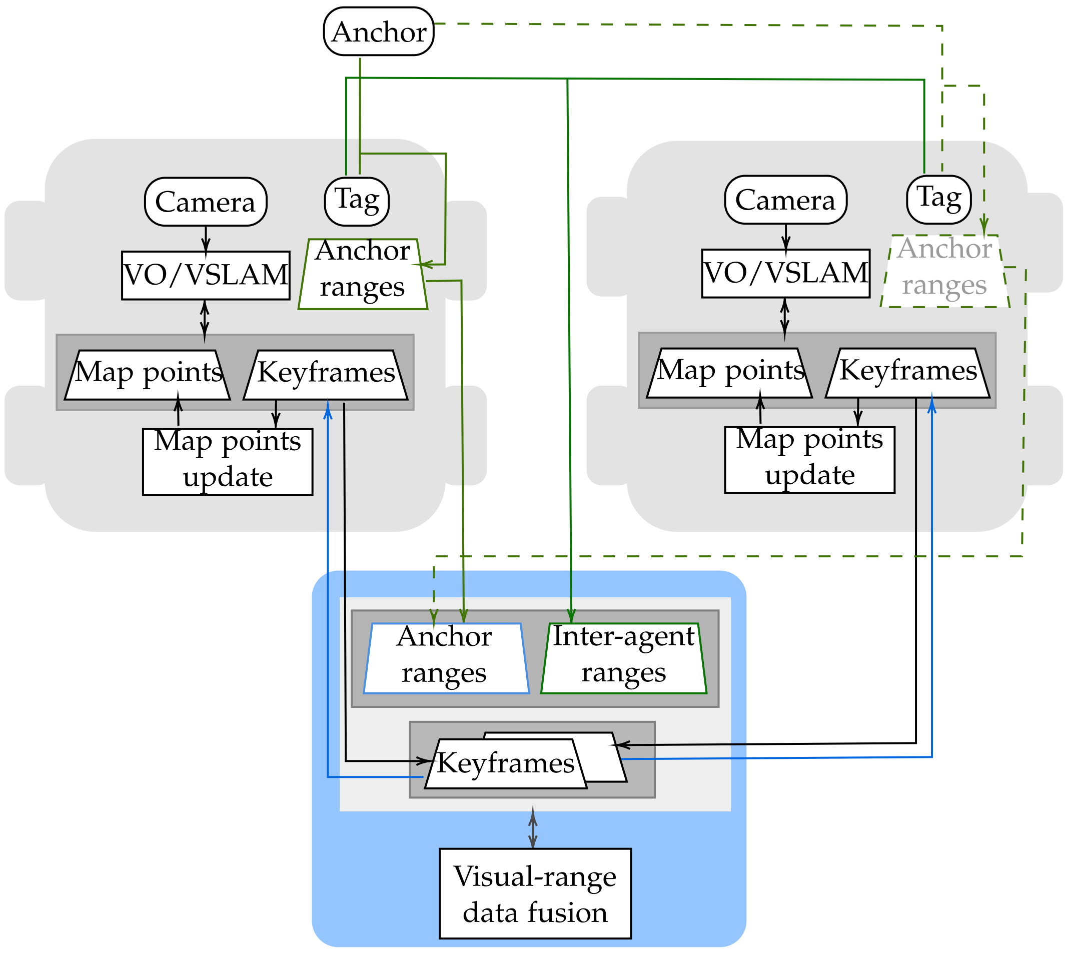


Figure 5.1.: The system overview of CoVOR-SLAM with the two-agent setup

addition to inter-agent ranges between the onboard ranging sensors (tag), all available ranges between robots and a reference point (anchor) are obtained, whenever robots can communicate with the anchor point. Since each robot uses its own reference frame to estimate the local poses and map points, the local reference frames need to be aligned before merging multiple robots' map databases. Inter-agent map points matches can be exploited to estimate the relative pose between two different reference frames, as described in Section 4.2. Thus, map points messages need to be transmitted to the processing unit until the reference frames are successfully aligned. Note that map point messages are transmitted only at the beginning of a mission to align the different local reference frames. Afterward, robots only need to transmit available range measurements (scalar values) and the keyframe messages that include the local pose estimates, while robots running collaborative VSLAM with inter-agent loop closures need to exchange map point messages during the entire mission. After aligning the local reference frames, the server can fuse multiple agents' visual odometry and range measurements to reduce their poses estimation error. The server sends the updated keyframe poses back to each robot (data feedback), and then map points are locally updated, using the differences of the keyframe poses before and after the data fusion. In Fig. 5.1, CoVOR-SLAM is shown with a centralized system architecture for convenience, but it can be structured in a decentralized manner, i.e. the blue part can be run using one of the onboard computers.

In the following sections, the system setup is first introduced, as well as the measurement prediction models. Then, the multi-agent data fusion approach is explained, followed by the data feedback and local map data update methods. The proposed system is evaluated using two different application scenarios with public image datasets and range measurements that are synthetically generated using the statistical error model. Lastly, the communication load of CoVOR-SLAM is compared to two state-of-the-art collaborative VSLAM methods using inter-agent loop closures.

5.1. System Setup and Measurement Models

Fig. 5.2 shows the system setup with three agents. Note that the three-agent setup is used only for convenience, and this setup can be extend to N-agent cases ($N > 3$). The local reference frame of the k-th robot is denoted with $\mathrm{L_k}$, and the global frame G is chosen to be a rotated version of $\mathrm{L_1}$ (agent-1's local reference frame). Since the $\mathrm{L_1}$'s XZ-plane is the horizontal plane (not conventional), $\mathrm{L_1}$ is rotated 90deg with respect to the X-axis to make the XY-plane as the horizontal plane. The relative pose from the local frame $\mathrm{L_1}$ to the global frame G is

$$\mathbf{S}_{\mathrm{GL_1}} = \begin{bmatrix} \mathbf{R}_{\mathrm{X}}(-90^\circ) & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & (\mathrm{s}_{\mathrm{GL_1}})^{-1} \end{bmatrix}, \tag{5.57}$$

including the scale parameter $\mathrm{s}_{\mathrm{GL_1}}$.

Assuming that the relative poses between the local reference frames are successfully estimated, the camera local poses $\mathbf{T}^t_{\mathrm{L_kC_k}}$ estimated using VSLAM are converted to the 7DoF similarity matrices with respect to the global frame $\mathbf{S}^t_{\mathrm{GC_k}}$. For example, the agent-2's local poses $\mathbf{T}^t_{\mathrm{L_2C_1}}$ are converted to the similarity matrices with respect to the global frame as $\mathbf{S}^t_{\mathrm{GC_2}} = \mathbf{S}_{\mathrm{GL_1}}\mathbf{S}_{\mathrm{L_1L_2}}\mathbf{T}^t_{\mathrm{L_2C_2}}$. All the camera poses converted to the global frame are defined as the system's state variables:

$$\mathbf{\Theta} = \{\{\mathbf{S}^{t_0}_{\mathrm{GC_k}}, \dots, \mathbf{S}^{t_{N_k}}_{\mathrm{GC_k}}\}, \forall k \in \{1, 2, \dots, K\}\}, \tag{5.58}$$

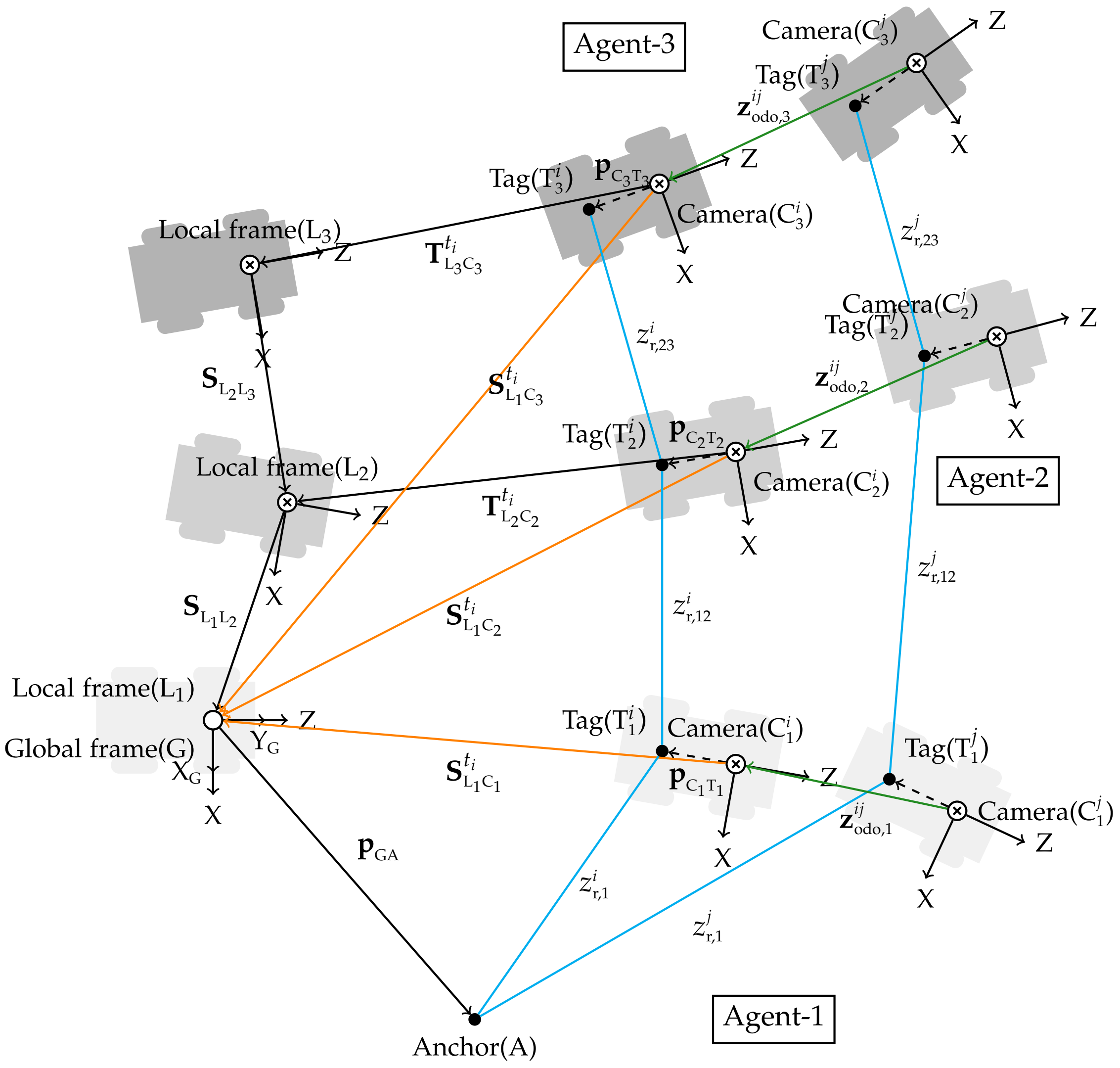


Figure 5.2.: The system setup of CoVOR-SLAM with three agents. The state variables are all agents' camera poses with respect to the global frame (orange). Visual odometry are green and range measurements (both agent-to-agent and agent-to-anchor) are blue

where K is the total number of the agents (e.g. $K = 3$ for the three-agent setup). These state variables are the orange lines in Fig. 5.2.

For CoVOR-SLAM, a prior knowledge, visual odometry measurements, and agent-to-anchor range measurements are exploited as the single-agent cases, and the same prediction models derived in Section 3.1 are used for these measurements. In addition, range measurements between the agent-k and agent-k' are used for CoVOR-SLAM, and they are modeled as the magnitude of the position vector from the agent-k to the agent-k' (or from the agent-k' to the agent-k):

$$
\begin{aligned}
h^{i}_{\mathrm{r},kk'}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_i}_{\mathrm{GC_{k'}}}) &= ||{}_{\mathrm{C_k}}\mathbf{p}^{t_i}_{\mathrm{T_kT_{k'}}}|| && (5.59)\\
&= ||{}_{\mathrm{C_{k'}}}\mathbf{p}^{t_i}_{\mathrm{T_{k'}T_k}}||. && (5.60)
\end{aligned}
$$

The position vector from the agent-k to the agent-k' defined in the camera frame of the agent-k (${}_{\mathrm{C_k}}\mathbf{p}^{t_i}_{\mathrm{T_kT_{k'}}}$) can be expanded as

$$
\begin{aligned}
{}_{\mathrm{C_k}}\mathbf{p}^{t_i}_{\mathrm{T_kT_{k'}}} &= {}_{\mathrm{C_k}}\mathbf{p}^{t_i}_{\mathrm{GT_{k'}}} - {}_{\mathrm{C_k}}\mathbf{p}^{t_i}_{\mathrm{GT_k}} \qquad (5.61)\\
&= \left({}_{\mathrm{C_k}}\mathbf{p}^{t_i}_{\mathrm{GC_{k'}}} + {}_{\mathrm{C_k}}\mathbf{p}_{\mathrm{C_{k'}T_{k'}}}\right) - \left({}_{\mathrm{C_k}}\mathbf{p}^{t_i}_{\mathrm{GC_k}} + \mathbf{p}_{\mathrm{C_kT_k}}\right)\\
&= \mathbf{R}^{t_i}_{\mathrm{C_kG}}\mathbf{p}^{t_i}_{\mathrm{GC_{k'}}} + \mathbf{R}^{t_i}_{\mathrm{C_kG}}\left(\mathbf{R}^{t_i}_{\mathrm{GC_{k'}}}\mathbf{p}_{\mathrm{C_{k'}T_{k'}}}\right) - \mathbf{R}^{t_i}_{\mathrm{C_kG}}\mathbf{p}^{t_i}_{\mathrm{GC_k}} - \mathbf{p}_{\mathrm{C_kT_k}}\\
&= \mathbf{R}^{t_i}_{\mathrm{C_kG}}\left(\mathbf{p}^{t_i}_{\mathrm{GC_{k'}}} + \mathbf{R}^{t_i}_{\mathrm{GC_{k'}}}\mathbf{p}_{\mathrm{C_{k'}T_{k'}}}\right) + \mathbf{t}^{t_i}_{\mathrm{GC_k}} - \mathbf{p}_{\mathrm{C_kT_k}},
\end{aligned}
$$

where $\mathbf{t}^{t_i}_{\mathrm{GC_k}} = -\mathbf{R}^{t_i}_{\mathrm{C_kG}}\mathbf{p}^{t_i}_{\mathrm{GC_k}}$. This vector can be described as a homogeneous vector including the scale parameters as

$$
\begin{aligned}
{}_{\mathrm{C_k}}\tilde{\mathbf{p}}^{t_i}_{\mathrm{T_kT_{k'}}} = \begin{pmatrix} {}_{\mathrm{C_k}}\mathbf{p}^{t_i}_{\mathrm{T_kT_{k'}}} \\ (s^{t_i}_{\mathrm{C_kG}})^{-1} \end{pmatrix} &= \begin{bmatrix} \mathbf{R}^{t_i}_{\mathrm{C_kG}} & \mathbf{t}^{t_i}_{\mathrm{GC_k}} \\ \mathbf{0}_{1\times 3} & (s^{t_i}_{\mathrm{C_kG}})^{-1} \end{bmatrix} \begin{pmatrix} \mathbf{p}^{t_i}_{\mathrm{GC_{k'}}} + \mathbf{R}^{t_i}_{\mathrm{GC_{k'}}}\mathbf{p}_{\mathrm{C_{k'}T_{k'}}} \\ 1 \end{pmatrix} - \begin{pmatrix} \mathbf{p}_{\mathrm{C_kT_k}} \\ (s^{t_i}_{\mathrm{C_kG}})^{-1} \end{pmatrix} \qquad (5.62)\\
&= \left(\mathbf{S}^{t_i}_{\mathrm{GC_k}}\right)^{-1}\tilde{\mathbf{p}}^{t_i}_{\mathrm{GT_{k'}}} - \tilde{\mathbf{p}}_{\mathrm{C_kT_k}}\\
&= \begin{pmatrix} \mathbf{p}^{t_i}_{\mathrm{C_kT_{k'}}} \\ (s^{t_i}_{\mathrm{C_kG}})^{-1} \end{pmatrix} - \begin{pmatrix} \mathbf{p}_{\mathrm{C_kT_k}} \\ (s^{t_i}_{\mathrm{C_kG}})^{-1} \end{pmatrix},
\end{aligned}
$$

where $\left(\mathbf{S}^{t_i}_{\mathrm{GC_k}}\right)^{-1} = \begin{bmatrix} \mathbf{R}^{t_i}_{\mathrm{C_kG}} & \mathbf{t}^{t_i}_{\mathrm{GC_k}} \\ \mathbf{0}_{1\times 3} & (s^{t_i}_{\mathrm{C_kG}})^{-1} \end{bmatrix}$.

The position vector from the agent-k' to the agent-k defined in the camera frame of the agent-k' (${}_{\mathrm{C_{k'}}}\mathbf{p}^{t_i}_{\mathrm{T_{k'}T_k}}$) can be expanded in a similar manner:

$$
\begin{aligned}
{}_{\mathrm{C_{k'}}}\mathbf{p}^{t_i}_{\mathrm{T_{k'}T_k}} &= {}_{\mathrm{C_{k'}}}\mathbf{p}^{t_i}_{\mathrm{GT_k}} - {}_{\mathrm{C_{k'}}}\mathbf{p}^{t_k}_{\mathrm{GT_{k'}}} \qquad (5.63)\\
&= \mathbf{R}^{t_i}_{\mathrm{C_kG}}\left(\mathbf{p}^{t_i}_{\mathrm{GC_k}} + \mathbf{R}^{t_i}_{\mathrm{GC_k}}\mathbf{p}_{\mathrm{C_kT_k}}\right) + \mathbf{t}^{t_i}_{\mathrm{GC_{k'}}} + \mathbf{p}_{\mathrm{C_{k'}T_{k'}}},
\end{aligned}
$$

and expressed as a homogeneous vector as

$$_{\mathrm{C_{k'}}}\tilde{\mathbf{p}}^{t_i}_{\mathrm{T_{k'}T_k}} = \begin{pmatrix} _{\mathrm{C_{k'}}}\mathbf{p}^{t_i}_{\mathrm{T_{k'}T_k}} \\ (s^{t_i}_{\mathrm{C_{k'}G}})^{-1} \end{pmatrix} = \begin{pmatrix} \mathbf{p}^{t_i}_{\mathrm{C_{k'}T_k}} \\ (s^{t_i}_{\mathrm{C_{k'}G}})^{-1} \end{pmatrix} - \begin{pmatrix} \mathbf{p}_{\mathrm{C_{k'}T_{k'}}} \\ (s^{t_i}_{\mathrm{C_{k'}G}})^{-1} \end{pmatrix}. \tag{5.64}$$

Since inter-agent range measurements are scalar values, the measurement errors are simply the differences of the values computed with the prediction model $h^i_{\mathrm{r},kk'}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_i}_{\mathrm{GC_{k'}}})$ and the measurements $z^i_{\mathrm{r},kk'}$:

$$e^i_{\mathrm{r},kk'}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_i}_{\mathrm{GC_{k'}}}, z^i_{\mathrm{r},kk'}) = h^i_{\mathrm{r},kk'}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_i}_{\mathrm{GC_{k'}}}) - z^i_{\mathrm{r},kk'}. \tag{5.65}$$

5.2. Multi-Agent Data Fusion and Data Feedback

A factor graph is created to fuse visual odometry and range measurements transmitted from multiple robots, as shown in Fig. 5.3. In this graph, the camera poses with respect to the global frame (state variables) are depicted as circular nodes. Anchor range factors $\phi^i_{\mathrm{r},k}$ are added when agents can obtain range measurements to them and an anchor. With a Gaussian noise assumption, a range factor can be described as the likelihood of the camera pose $\mathbf{S}^{t_i}_{\mathrm{GC_k}}$ given the range measurement $z^i_{\mathrm{r},k}$:

$$\phi^i_{\mathrm{r},k} = l(\mathbf{S}^{t_i}_{\mathrm{GC_k}}; z^i_{\mathrm{r},k}) \propto \exp\left(-\frac{1}{2}||e^i_{\mathrm{r},k}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, z^i_{\mathrm{r},k})||^2_{\Sigma_{\mathrm{r},k}}\right), \tag{5.66}$$

where the error function $e^i_{\mathrm{r},k}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, z^i_{\mathrm{r},k})$ is defined as Eq. (3.26).

In addition, inter-agent range factors $\phi^i_{\mathrm{r},kk'}$ are added between the agents. This factor is equivalent to the likelihood of the two camera poses $\mathbf{S}^{t_i}_{\mathrm{GC_k}}$ and $\mathbf{S}^{t_i}_{\mathrm{GC_{k'}}}$ given the inter-agent range measurement $z^i_{\mathrm{r},kk'}$. With a Gaussian noise assumption,

$$\phi^i_{\mathrm{r},kk'} = l(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_i}_{\mathrm{GC_{k'}}}; z^i_{\mathrm{r},kk'}) \propto \exp\{-\frac{1}{2}||e^i_{\mathrm{r},kk'}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_i}_{\mathrm{GC_{k'}}}, z^i_{\mathrm{r},kk'})||^2_{\Sigma_{\mathrm{r},kk'}}\}, \tag{5.67}$$

where the error function $e^i_{\mathrm{r},kk'}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_i}_{\mathrm{GC_{k'}}}, z^i_{\mathrm{r},kk'})$ is defined in Eq. (5.65).

Between the two consecutive poses of each agent, the likelihood of the states given the odometry measurements is added as an odometry factor $\phi^{ij}_{\mathrm{odo},k}$

$$\phi^{ij}_{\mathrm{odo},k} = l(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_j}_{\mathrm{GC_k}}; \mathbf{z}^{ij}_{\mathrm{odo},k}) \propto \exp\left(-\frac{1}{2}||\mathbf{e}^{ij}_{\mathrm{odo},k}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_j}_{\mathrm{GC_k}}, \mathbf{z}^{ij}_{\mathrm{odo},k})||^2_{\Sigma_{\mathrm{odo},k}}\right), \tag{5.68}$$

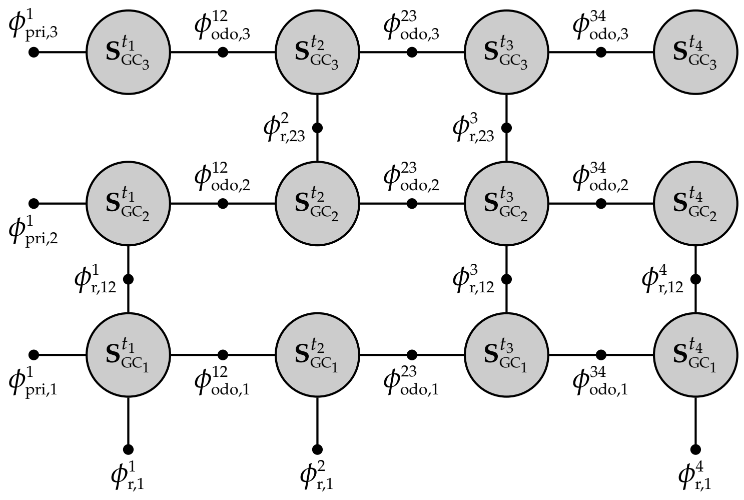

$\phi^1_{\text{pri},3}$
$\phi^{12}_{\text{odo},3}$
$\phi^{23}_{\text{odo},3}$
$\phi^{34}_{\text{odo},3}$
$\mathbf{S}^{t_1}_{\text{GC}_3}$
$\mathbf{S}^{t_2}_{\text{GC}_3}$
$\mathbf{S}^{t_3}_{\text{GC}_3}$
$\mathbf{S}^{t_4}_{\text{GC}_3}$
$\phi^2_{\text{r},23}$
$\phi^3_{\text{r},23}$
$\phi^{12}_{\text{odo},2}$
$\phi^{23}_{\text{odo},2}$
$\phi^{34}_{\text{odo},2}$
$\mathbf{S}^{t_1}_{\text{GC}_2}$
$\mathbf{S}^{t_2}_{\text{GC}_2}$
$\mathbf{S}^{t_3}_{\text{GC}_2}$
$\mathbf{S}^{t_4}_{\text{GC}_2}$
$\phi^1_{\text{pri},2}$
$\phi^1_{\text{r},12}$
$\phi^3_{\text{r},12}$
$\phi^4_{\text{r},12}$
$\mathbf{S}^{t_1}_{\text{GC}_1}$
$\mathbf{S}^{t_2}_{\text{GC}_1}$
$\mathbf{S}^{t_3}_{\text{GC}_1}$
$\mathbf{S}^{t_4}_{\text{GC}_1}$
$\phi^1_{\text{pri},1}$
$\phi^{12}_{\text{odo},1}$
$\phi^{23}_{\text{odo},1}$
$\phi^{34}_{\text{odo},1}$
$\phi^1_{\text{r},1}$
$\phi^2_{\text{r},1}$
$\phi^4_{\text{r},1}$

Figure 5.3.: Factor graph created for fusing visual odometry and range measurements of multiple agents (created with the three-agent setup)

.

where the error function $\mathbf{e}^{ij}_{\text{odo},k}(\mathbf{S}^{t_i}_{\text{GC}_\text{k}}, \mathbf{S}^{t_j}_{\text{GC}_\text{k}}, \mathbf{z}^{ij}_{\text{odo},k})$ is defined as Eq. (3.29).

Lastly, the conditional probability of the first pose given its prior knowledge is added as a prior factor $\phi_{\text{pri},k}$ at each agent's first pose:

$$\phi_{\text{pri},k} = P(\mathbf{S}^{t_1}_{\text{GC}_\text{k}} | \mathbf{z}_{\text{pri},k}) \propto \exp\left(-\frac{1}{2}||\mathbf{e}_{\text{pri},k}(\mathbf{S}^{t_1}_{\text{GC}_\text{k}}, \mathbf{z}_{\text{pri},k})||^2_{\Sigma_{\text{pri},k}}\right). \tag{5.69}$$

The error function $\mathbf{e}_{\text{pri},k}(\mathbf{S}^{t_1}_{\text{GC}_\text{k}}, \mathbf{z}_{\text{pri},k})$ is defined in Eq. (3.31).

A Maximum A Posteriori (MAP) problem is formulated using the factor graph as

$$\hat{\mathbf{\Theta}} = \underset{\mathbf{\Theta}}{\text{argmax}} \prod_k \phi_{\text{pri},k} \prod_k \prod_{i,j} \phi^{ij}_{\text{odo},k} \prod_k \prod_i \phi^i_{\text{r},k} \prod_{k,k'} \prod_i \phi^i_{\text{r},kk'}. \tag{5.70}$$

When all the measurements have Gaussian noise, this MAP problem is equivalent to the LSE problem that minimizes the sum of the squared measurement errors:

$$\begin{aligned}
\hat{\mathbf{\Theta}} = \underset{\mathbf{\Theta}}{\operatorname{argmin}} & \underbrace{\sum_k \left(\mathbf{e}_{\mathrm{pri},k}(\mathbf{S}^{t_0}_{\mathrm{GC_k}}, \mathbf{z}_{\mathrm{pri},k})\right)^{\mathrm{T}} \mathbf{\Sigma}^{-1}_{\mathrm{pri},k} \mathbf{e}_{\mathrm{pri},k}(\mathbf{S}^{t_0}_{\mathrm{GC_k}}, \mathbf{z}_{\mathrm{pri},k})}_{\text{a prior error}} \\
& + \underbrace{\sum_k \sum_{i,j} \left(\mathbf{e}_{\mathrm{odo},k}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_j}_{\mathrm{GC_k}}, \mathbf{z}^{ij}_{\mathrm{odo},k})\right)^{\mathrm{T}} \mathbf{\Sigma}^{-1}_{\mathrm{odo},k} \mathbf{e}_{\mathrm{odo},k}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_j}_{\mathrm{GC_k}}, \mathbf{z}^{ij}_{\mathrm{odo},k})}_{\text{visual odometry measurement errors}} \\
& + \underbrace{\sum_k \sum_i \left(e^i_{\mathrm{r},k}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, z^i_{\mathrm{r},k})\right)^{\mathrm{T}} \mathbf{\Sigma}^{-1}_{\mathrm{r},k} e_{\mathrm{r},k}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, z^i_{\mathrm{r},k})}_{\text{agent-to-anchor range measurement errors}} \\
& + \underbrace{\sum_{k,k'} \sum_i \left(e^i_{\mathrm{r},kk'}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_i}_{\mathrm{GC_{k'}}}, z^i_{\mathrm{r},kk'})\right)^{\mathrm{T}} \mathbf{\Sigma}^{-1}_{\mathrm{r},kk'} e^i_{\mathrm{r},kk'}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_i}_{\mathrm{GC_{k'}}}, z^i_{\mathrm{r},kk'})}_{\text{inter-agent range measurement errors}} \\
\triangleq & \underset{\mathbf{\Theta}}{\operatorname{argmin}} \mathcal{F}(\mathbf{\Theta}, \mathbf{Z}). \qquad (5.71)
\end{aligned}$$

The Levenberg-Marquardt algorithm [Levenberg, 1944, Marquardt, 1963] is used to solve this LSE problem. This algorithm iteratively estimates the optimal camera poses as the twist coordinates (vector format)

$$\Delta\hat{\boldsymbol{\theta}} = \{\{\hat{\boldsymbol{\xi}}^{t_0}_{GC_k}, \dots, \hat{\boldsymbol{\xi}}^{t_f}_{GC_k}\}, \forall k \in \{1, 2, \dots, K\}\} \qquad (5.72)$$

by solving the equation

$$(\mathbf{H} + d\mathbf{I})\Delta\hat{\boldsymbol{\theta}} = -\mathbf{b}, \qquad (5.73)$$

In this equation, $\mathbf{H}$ denotes the $7 \times \sum_k 7N_k$ Hessian matrix:

$$\begin{aligned}
\mathbf{H} &= \sum_k \mathbf{H}_{\mathrm{pri,k}} + \sum_k \sum_{i,j} \mathbf{H}^{ij}_{\mathrm{odo,k}} + \sum_k \sum_i \mathbf{H}^i_{\mathrm{r,k}} + \sum_{k,k'} \sum_i \mathbf{H}^i_{\mathrm{r,kk'}} \qquad (5.74) \\
&= \underbrace{\sum_k \left(\mathbf{J}_{\mathbf{e}_{\mathrm{pri},k}}\right)^{\mathrm{T}} \mathbf{\Sigma}_{\mathrm{pri},k} \mathbf{J}_{\mathbf{e}_{\mathrm{pri},k}}}_{\text{a prior}} + \underbrace{\sum_k \sum_{i,j} \left(\mathbf{J}_{\mathbf{e}^{ij}_{\mathrm{odo},k}}\right)^{\mathrm{T}} \mathbf{\Sigma}_{\mathrm{odo},k} \mathbf{J}_{\mathbf{e}^{ij}_{\mathrm{odo},k}}}_{\text{visual odometry}} \\
&+ \underbrace{\sum_k \sum_i \left(\mathbf{J}_{e^i_{\mathrm{r},k}}\right)^{\mathrm{T}} \mathbf{\Sigma}_{\mathrm{r},k} \mathbf{J}_{e^i_{\mathrm{r},k}}}_{\text{agent-to-anchor ranges}} + \underbrace{\sum_{k,k'} \sum_i \left(\mathbf{J}_{e^i_{\mathrm{r},kk'}}\right)^{\mathrm{T}} \mathbf{\Sigma}_{\mathrm{r},kk'} \mathbf{J}_{e^i_{\mathrm{r},kk'}}}_{\text{inter-agent ranges}},
\end{aligned}$$

where N_k is the total number of the agent-k's keyframe poses. $\mathbf{b}$ on the right hand side is a

$\sum_k 7N_k \times 1$ vector:

$$\begin{aligned}\mathbf{b} &= \sum_k \mathbf{b}_{\mathrm{pri,k}} + \sum_k \sum_{i,j} \mathbf{b}^{ij}_{\mathrm{odo,k}} + \sum_k \sum_i \mathbf{b}^i_{\mathrm{r,k}} + \sum_{k,k'} \sum_i \mathbf{b}^i_{\mathrm{r,kk'}} \\ &= \underbrace{\sum_k \sum_i \left(\mathbf{J}_{\mathbf{e}_{\mathrm{pri},k}}\right)^{\mathrm{T}} \mathbf{\Sigma}_{\mathrm{pri},k} \mathbf{e}_{\mathrm{pri},k}}_{\text{a prior}} + \underbrace{\sum_k \sum_{i,j} \left(\mathbf{J}_{\mathbf{e}^{ij}_{\mathrm{odo},k}}\right)^{\mathrm{T}} \mathbf{\Sigma}_{\mathrm{odo},k} \mathbf{e}^{ij}_{\mathrm{odo},k}}_{\text{visual odometry}} \\ &+ \underbrace{\sum_k \sum_i \left(\mathbf{J}_{e^i_{\mathrm{r},k}}\right)^{\mathrm{T}} \mathbf{\Sigma}_{\mathrm{r},k} e^i_{\mathrm{r},k}}_{\text{agent-to-anchor ranges}} + \underbrace{\sum_{k,k'} \sum_i \left(\mathbf{J}_{e^i_{\mathrm{r},kk'}}\right)^{\mathrm{T}} \mathbf{\Sigma}_{\mathrm{r},kk'} e^i_{\mathrm{r},kk'}}_{\text{inter-agent ranges}} .\end{aligned} \tag{5.75}$$

The Jacobian of the prior error function in Eq. (5.74) and Eq. (5.75) is a $7 \times \sum_{k=1}^{K} 7N_k$ matrix:

$$\mathbf{J}_{\mathbf{e}_{\mathrm{pri},k}}(\boldsymbol{\theta}) = \left[\begin{array}{c|c|c|c|c} \underbrace{\mathbf{0}_{7\times N_1}}_{\text{agent-1}} & \cdots & \underbrace{\frac{\partial \mathbf{e}_{\mathrm{pri},k}(\mathbf{S}^{t_1}_{\mathrm{GC_k}}, \mathbf{z}_{\mathrm{pri},k})}{\partial \boldsymbol{\xi}^{t_1}_{GC_k}}, \ldots, \mathbf{0}}_{\text{agent-}k} & \cdots & \underbrace{\mathbf{0}_{7\times N_K}}_{\text{agent-}K} \end{array}\right], \tag{5.76}$$

where the partial derivative $\frac{\partial \mathbf{e}_{\mathrm{pri},k}(\mathbf{S}^{t_0}_{\mathrm{GC_k}}, \mathbf{z}_{\mathrm{pri},k})}{\partial \xi^{t_0}_{GC_k}}$ is computed with Eq. (3.45). As shown in this equation, only the first component of the agent-k's part is non-zero, and all the elements of the Jacobian are zeros.

The Jacobian of the odometry error is a $7 \times \sum_{k=1}^{K} 7N_k$ matrix as

$$\begin{aligned}&\mathbf{J}^{ij}_{\mathbf{e}_{\mathrm{odo},k}}(\boldsymbol{\theta}) \\ &= \left[\begin{array}{c|c|c|c|c} \underbrace{\mathbf{0}_{7\times N_1}}_{\text{agent-1}} & \cdots & \underbrace{\mathbf{0}, \ldots, \frac{\partial \mathbf{e}^{ij}_{\mathrm{odo},k}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_j}_{\mathrm{GC_k}}, \mathbf{z}^{ij}_{\mathrm{odo},k})}{\partial \boldsymbol{\xi}^{t_i}_{GC_k}}, \frac{\partial \mathbf{e}^{ij}_{\mathrm{odo},k}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_j}_{\mathrm{GC_k}}, \mathbf{z}^{ij}_{\mathrm{odo},k})}{\partial \boldsymbol{\xi}^{t_j}_{GC_k}}, \ldots, \mathbf{0}}_{\text{agent-}k} & \cdots & \underbrace{\mathbf{0}_{7\times N_K}}_{\text{agent-}K} \end{array}\right],\end{aligned} \tag{5.77}$$

whose elements are all zeros, except for the i-th and $(i+1)$-th components of the agent-k's part. The two partial derivatives $\frac{\partial \mathbf{e}^{ij}_{\mathrm{odo},k}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_j}_{\mathrm{GC_k}}, \mathbf{z}^{ij}_{\mathrm{odo},k})}{\partial \xi^{t_i}_{GC_k}}$ and $\frac{\partial \mathbf{e}^{ij}_{\mathrm{odo},k}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_j}_{\mathrm{GC_k}}, \mathbf{z}^{ij}_{\mathrm{odo},k})}{\partial \xi^{t_j}_{GC_k}}$ are derived in Eq. (3.42) and Eq. (3.43), respectively.

When the agent-k obtains a range measurement at t_i, the Jacobian of the ranging error function needs to be computed as

$$\mathbf{J}^{i}_{e_{\mathrm{r},k}}(\boldsymbol{\theta}) = \left[\begin{array}{c|c|c|c|c} \underbrace{\mathbf{0}_{1\times N_1}}_{\text{agent-1}} & \cdots & \underbrace{\mathbf{0},\ldots,\frac{\partial e^{i}_{\mathrm{r},k}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, z^{i}_{\mathrm{r},k})}{\partial \boldsymbol{\xi}^{t_i}_{GC_k}},\ldots,\mathbf{0}}_{\text{agent-}k} & \cdots & \underbrace{\mathbf{0}_{1\times N_K}}_{\text{agent-}K} \end{array}\right], \tag{5.78}$$

which is a $1 \times \sum_{k=1}^{K} 7N_k$ matrix. The partial derivative $\frac{\partial e^{i}_{\mathrm{r},k}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, z^{i}_{\mathrm{r},k})}{\partial \boldsymbol{\xi}^{t_i}_{GC_k}}$ is derived in Eq. (3.51). As shown in the equation, all the elements of the Jacobian are zeros, except for the i-th component of the agent-k's part.

Additionally, when a range measurement is available between the agent-k and agent-k' at t_i, the $1 \times \sum_{k=1}^{K} 7N_k$ Jacobian matrix of the inter-agent ranging error function needs to be computed as

$$\mathbf{J}^{i}_{\mathbf{e}_{\mathrm{r,kk'}}}(\boldsymbol{\Theta}) = \left[\begin{array}{c|c|c|c|c} \mathbf{0} & \underbrace{\mathbf{0},\ldots,\frac{\partial e^{i}_{\mathrm{r},kk'}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_i}_{\mathrm{GC_{k'}}}, z^{i}_{\mathrm{r},kk'})}{\partial \boldsymbol{\xi}^{t_i}_{GC_k}},\ldots,\mathbf{0}}_{\text{agent-}k} & \cdots & \underbrace{\mathbf{0},\ldots,\frac{\partial e^{i}_{\mathrm{r},kk'}(\mathbf{S}^{t_i}_{\mathrm{GC_1}}, \mathbf{S}^{t_i}_{\mathrm{GC_2}}, z^{i}_{\mathrm{r},kk'})}{\partial \boldsymbol{\xi}^{t_i}_{GC_{k'}}},\ldots,\mathbf{0}}_{\text{agent-}k'} & \mathbf{0} \end{array}\right]. \tag{5.79}$$

In this Jacobian matrix, only the i-th components of the agent-k and agent-k' are non-zeros. The partial derivative $\frac{\partial e^{i}_{\mathrm{r},kk'}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_i}_{\mathrm{GC_{k'}}}, z^{i}_{\mathrm{r},kk'})}{\partial \boldsymbol{\xi}^{t_i}_{GC_k}}$ of the agent-k is computed as

$$\begin{aligned}\frac{\partial e^{i}_{\mathrm{r},kk'}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_i}_{\mathrm{GC_{k'}}}, z^{i}_{\mathrm{r},kk'})}{\partial \boldsymbol{\xi}^{t_i}_{GC_k}} &= \frac{\partial h^{i}_{\mathrm{r},kk'}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_i}_{\mathrm{GC_{k'}}})}{\partial \boldsymbol{\xi}^{t_i}_{GC_k}} \\ &= \frac{\partial \|{}_{\mathrm{C_k}}\mathbf{p}^{t_i}_{\mathrm{T_kT_{k'}}}\|}{\partial \boldsymbol{\xi}^{t_i}_{GC_k}} = \frac{\partial \|{}_{\mathrm{C_k}}\mathbf{p}^{t_i}_{\mathrm{T_kT_{k'}}}\|}{\partial {}_{\mathrm{C_k}}\mathbf{p}^{t_i}_{\mathrm{T_kT_{k'}}}} \frac{\partial {}_{\mathrm{C_k}}\mathbf{p}^{t_i}_{\mathrm{T_kT_{k'}}}}{\partial {}_{\mathrm{C_k}}\tilde{\mathbf{p}}^{t_i}_{\mathrm{T_kT_{k'}}}} \frac{\partial {}_{\mathrm{C_k}}\tilde{\mathbf{p}}^{t_i}_{\mathrm{T_kT_{k'}}}}{\partial \boldsymbol{\xi}^{t_i}_{GC_k}},\end{aligned} \tag{5.80}$$

where each part of the right hand side can be expanded as

$$\frac{\partial \|{}_{\mathrm{C_k}}\mathbf{p}^{t_i}_{\mathrm{T_kT_{k'}}}\|}{\partial {}_{\mathrm{C_k}}\mathbf{p}^{t_i}_{\mathrm{T_kT_{k'}}}} = \frac{1}{\|{}_{\mathrm{C_k}}\mathbf{p}^{t_i}_{\mathrm{T_kT_{k'}}}\|}\left({}_{\mathrm{C_k}}\mathbf{p}^{t_i}_{\mathrm{T_kT_{k'}}}\right)^{\mathrm{T}} \tag{5.81}$$

$$\frac{\partial {}_{\mathrm{C_k}}\mathbf{p}^{t_i}_{\mathrm{T_kT_{k'}}}}{\partial {}_{\mathrm{C_k}}\tilde{\mathbf{p}}^{t_i}_{\mathrm{T_kT_{k'}}}} = \begin{bmatrix}\mathbf{I}_3 & \mathbf{0}_{3\times1}\end{bmatrix} \tag{5.82}$$

$$\frac{\partial {}_{\mathrm{C_k}}\tilde{\mathbf{p}}^{t_i}_{\mathrm{T_kT_{k'}}}}{\partial \boldsymbol{\xi}^{t_i}_{GC_k}} = \frac{\partial}{\partial \boldsymbol{\xi}^{t_i}_{GT_k}}\left(\left(\mathbf{S}^{t_i}_{\mathrm{GC_k}}\right)^{-1}\tilde{\mathbf{p}}^{t_i}_{\mathrm{GC_{k'}}} - \tilde{\mathbf{p}}_{\mathrm{C_kT_k}}\right)$$

$$
\begin{aligned}
&= \mathrm{H}[\tilde{\mathbf{p}}^{t_i}_{\mathrm{C_k T_{k'}}}] - \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & 1 \end{bmatrix} \\
&= \begin{bmatrix} [\mathbf{p}^{t_i}_{\mathrm{C_k T_{k'}}}]_\times & -s^{t_i}_{\mathrm{GC_k}}\mathbf{I}_3 & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & s^{t_i}_{\mathrm{GC_k}} \end{bmatrix} - \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & 1 \end{bmatrix} \\
&= \begin{bmatrix} [\mathbf{p}^{t_i}_{\mathrm{C_k T_{k'}}}]_\times & -s^{t_i}_{\mathrm{GC_k}}\mathbf{I}_3 & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & s^{t_i}_{\mathrm{GC_k}} - 1 \end{bmatrix}.
\end{aligned} \tag{5.83}
$$

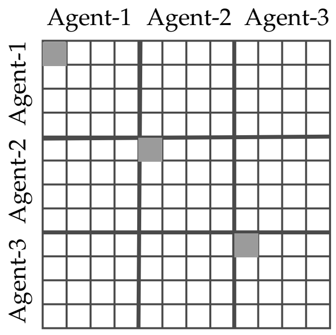
By substituting these results into Eq. (5.80), the partial derivative can be finally computed as

$$
\begin{aligned}
\frac{\partial e^i_{\mathrm{r},kk'}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_i}_{\mathrm{GC_{k'}}}, z^i_{\mathrm{r},kk'})}{\partial \boldsymbol{\xi}^{t_i}_{GC_k}} &= \begin{bmatrix} \frac{\partial h^i_{\mathrm{r},kk'}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_i}_{\mathrm{GC_{k'}}})}{\partial \omega_1} & \frac{\partial h^i_{\mathrm{r},kk'}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_i}_{\mathrm{GC_{k'}}})}{\partial v_k} & \frac{\partial h^i_{\mathrm{r},kk'}(\mathbf{S}^{t_i}_{\mathrm{GC_k}}, \mathbf{S}^{t_i}_{\mathrm{GC_{k'}}})}{\partial \lambda_k} \end{bmatrix} \\
&= \frac{1}{||{}_{\mathrm{C_k}}\mathbf{p}^{t_i}_{\mathrm{T_k T_{k'}}}||} \begin{bmatrix} ({}_{\mathrm{C_k}}\mathbf{p}^{t_i}_{\mathrm{T_k T_{k'}}})^{\mathrm{T}}[\mathbf{p}^{t_i}_{\mathrm{C_k T_{k'}}}]_\times & -s^{t_i}_{\mathrm{GC_k}}({}_{\mathrm{C_k}}\mathbf{p}^{t_i}_{\mathrm{T_k T_{k'}}})^{\mathrm{T}} & 0 \end{bmatrix}
\end{aligned} \tag{5.84}
$$

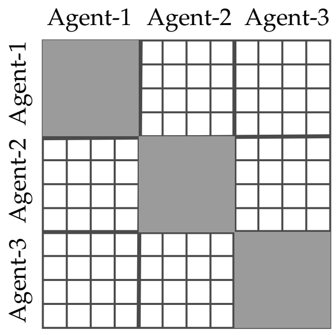
Similarly, the partial derivatives $\frac{\partial e^i_{\mathrm{r},12}(\mathbf{S}^{t_i}_{\mathrm{GC_1}}, \mathbf{S}^{t_i}_{\mathrm{GC_2}}, z^i_{\mathrm{r},12})}{\partial \xi^{t_i}_{GC_2}}$ of the agent-k' can be calculated as

$$
\begin{aligned}
\frac{\partial e^i_{\mathrm{r},12}(\mathbf{S}^{t_i}_{\mathrm{GC_1}}, \mathbf{S}^{t_i}_{\mathrm{GC_2}}, z^i_{\mathrm{r},12})}{\partial \boldsymbol{\xi}^{t_i}_{GC_2}} &= \begin{bmatrix} \frac{\partial h^i_{\mathrm{r},12}(\mathbf{S}^{t_i}_{\mathrm{GC_1}}, \mathbf{S}^{t_i}_{\mathrm{GC_2}})}{\partial \omega_2} & \frac{\partial h^i_{\mathrm{r},12}(\mathbf{S}^{t_i}_{\mathrm{GC_1}}, \mathbf{S}^{t_i}_{\mathrm{GC_2}})}{\partial v_2} & \frac{\partial h^i_{\mathrm{r},12}(\mathbf{S}^{t_i}_{\mathrm{GC_1}}, \mathbf{S}^{t_i}_{\mathrm{GC_2}})}{\partial \lambda_2} \end{bmatrix} \\
&= \frac{1}{||{}_{\mathrm{C_2}}\mathbf{p}^{t_i}_{\mathrm{T_2 T_1}}||} \begin{bmatrix} ({}_{\mathrm{C_2}}\mathbf{p}^{t_i}_{\mathrm{T_2 T_1}})^{\mathrm{T}}[\mathbf{p}^{t_i}_{\mathrm{C_2 T_1}}]_\times & -s^{t_i}_{\mathrm{GC_2}}({}_{\mathrm{C_2}}\mathbf{p}^{t_i}_{\mathrm{T_2 T_1}})^{\mathrm{T}} & 0 \end{bmatrix}.
\end{aligned} \tag{5.85}
$$

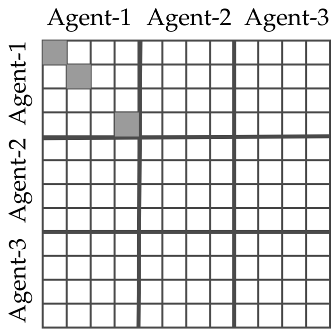
Fig. 5.4 shows the structure of the Hessian matrix computed using the three-agent setup depicted in Fig. 5.3. A small box in the figure represents the 7DoF pose at each timestamp. The non-zeros elements of the matrix are marked in gray. As shown in Fig. 5.4a, all the elements of the prior error's Hessian matrix $\sum_k \mathbf{H}_{\mathrm{pri,k}}$ are zeros except for the first row and column of each agent's entry. In addition, visual odometry error's Hessian $\sum_k \sum_{i,j} \mathbf{H}^{i,j}_{\mathrm{odo,k}}$ has no correlation between the different agents, as depicted in Fig. 5.4b. The Hessian of the agent-to-anchor ranges $\sum_k \sum_i \mathbf{H}^i_{\mathrm{r,k}}$ also does not have a inter-agent correlation, as illustrated in Fig. 5.4c, and its elements are all zeros except for three keyframes of the agent-1 at which the anchor connections are available. As can be seen in Fig. 5.4d, the Hessian of the inter-agent range error $\sum_{k,k'} \sum_i \mathbf{H}^i_{\mathrm{r,kk'}}$ has non-zeros components between different agents when inter-agent ranges are available. Fig. 5.4e shows the sparse $\mathbf{H}$ matrix that is computed by combining all the sub-matrices, i.e. $\mathbf{H} = \sum_k \mathbf{H}_{\mathrm{pri,k}} + \sum_k \sum_{i,j} \mathbf{H}^{ij}_{\mathrm{odo,k}} + \sum_k \sum_i \mathbf{H}^i_{\mathrm{r,k}} + \sum_{k,k'} \sum_i \mathbf{H}^i_{\mathrm{r,kk'}}$. As shown in this figure, both agent-to-anchor and inter-agent range measurements do not induce a big complexity to the system (Hessian matrix), so accurate collaborative SLAM is possible without requiring substantial



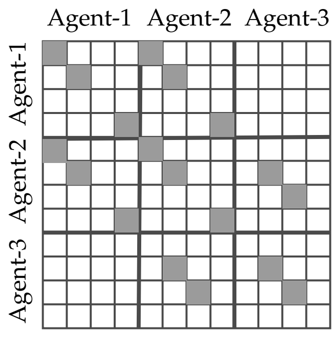
(a) Hessian matrix of the prior error ($\sum_k \mathbf{H}_{\text{pri,k}}$)



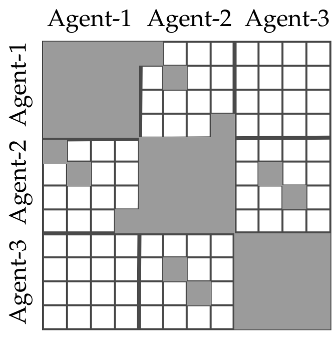
(b) Hessian matrix of the visual odometry error ($\sum_k \sum_{i,j} \mathbf{H}^{i,j}_{\text{odo,k}}$)



(c) Hessian matrix of the agent-to-anchor range error ($\sum_k \sum_i \mathbf{H}^{i}_{\text{r,k}}$)



(d) Hessian matrix of the inter-agent range error ($\sum_{k,k'} \sum_i \mathbf{H}^{i}_{\text{r,kk}'}$)



(e) The combination of all the Hessian matrices ($\mathbf{H} = \sum_k \mathbf{H}_{\text{pri,k}} + \sum_k \sum_{i,j} \mathbf{H}^{ij}_{\text{odo,k}} + \sum_k \sum_i \mathbf{H}^{i}_{\text{r,k}} + \sum_{k,k'} \sum_i \mathbf{H}^{i}_{\text{r,kk}'}$)

Figure 5.4.: An example of the Hessian matrix structure computed with the three-agent setup

computational power.

After the multi-agent data fusion, the processing unit transmits the updated keyframe poses back to the robots. Then, map points are locally updated on each robot's onboard computer using the differences of the keyframe poses before and after the data fusion. For example, the agent-1 in Fig. 5.2 computes the 7DoF camera poses with respect to its own local reference frame as

$$\begin{aligned}\hat{\mathbf{S}}^{t_i}_{\mathrm{L}_1\mathrm{C}_1} &= \left(\mathbf{S}_{\mathrm{GL}_1}\right)^{-1}\hat{\mathbf{S}}^{t_i}_{\mathrm{GC}_1} \\ &= \begin{bmatrix}\hat{\mathbf{R}}^{t_i}_{\mathrm{GC}_1} & \hat{\mathbf{t}}^{t_i}_{\mathrm{C}_1\mathrm{G}} \\ 0 & \left(\hat{\mathrm{s}}^{t_i}_{\mathrm{GC}_1}\right)^{-1}\mathrm{s}_{\mathrm{GL}_1}\end{bmatrix} \\ &= \begin{bmatrix}\hat{\mathbf{R}}^{t_i}_{\mathrm{L}_1\mathrm{C}_1} & \hat{\mathbf{t}}^{t_i}_{\mathrm{C}_1\mathrm{L}_1} \\ 0 & \left(\hat{\mathrm{s}}^{t_i}_{\mathrm{L}_1\mathrm{C}_1}\right)^{-1}\end{bmatrix}.\end{aligned} \tag{5.86}$$

To keep the local poses as 6DoF matrices, $\hat{\mathrm{s}}^{t_i}_{\mathrm{L}_1\mathrm{C}_1}$ is multiplied to the translation vector as

$$\hat{\mathbf{T}}^{t_i}_{\mathrm{L}_1\mathrm{C}_1} = \begin{bmatrix}\hat{\mathbf{R}}^{t_i}_{\mathrm{L}_1\mathrm{C}_1} & \hat{\mathrm{s}}^{t_i}_{\mathrm{L}_1\mathrm{C}_1}\hat{\mathbf{t}}^{t_i}_{\mathrm{C}_1\mathrm{L}_1} \\ 0 & 1\end{bmatrix}.$$

Using the difference of the keyframe pose before the data fusion $\mathbf{T}^{t_i}_{\mathrm{L}_1\mathrm{C}_1}$ and the updated pose after the fusion $\hat{\mathbf{T}}^{t_i}_{\mathrm{L}_1\mathrm{C}_1}$, the homogeneous coordinates of the map points $\hat{\tilde{\mathbf{X}}}^i_{\mathrm{L}_1}$ can be updated as

$$\hat{\tilde{\mathbf{X}}}^i_{\mathrm{L}_1} = \hat{\mathbf{T}}^{t_i}_{\mathrm{L}_1\mathrm{C}_1}\left(\mathbf{T}^{t_i}_{\mathrm{L}_1\mathrm{C}_1}\right)^{-1}\begin{pmatrix}\mathbf{X}^i_{\mathrm{L}_1} \\ 1\end{pmatrix} = \begin{pmatrix}\hat{\mathbf{X}}^i_{\mathrm{L}_1} \\ (\hat{\mathrm{s}}^{t_i}_{\mathrm{L}_1\mathrm{C}_1})^{-1}\end{pmatrix} \tag{5.87}$$

Then, these homogeneous coordinates are converted to the 3D Euclidean coordinates by multiplying the scale to the first three elements, i.e. $\hat{\mathbf{X}}^i_{\mathrm{L}_1} \leftarrow \hat{\mathrm{s}}^{t_i}_{\mathrm{L}_1\mathrm{C}_1}\hat{\mathbf{X}}^i_{\mathrm{L}_1}$.

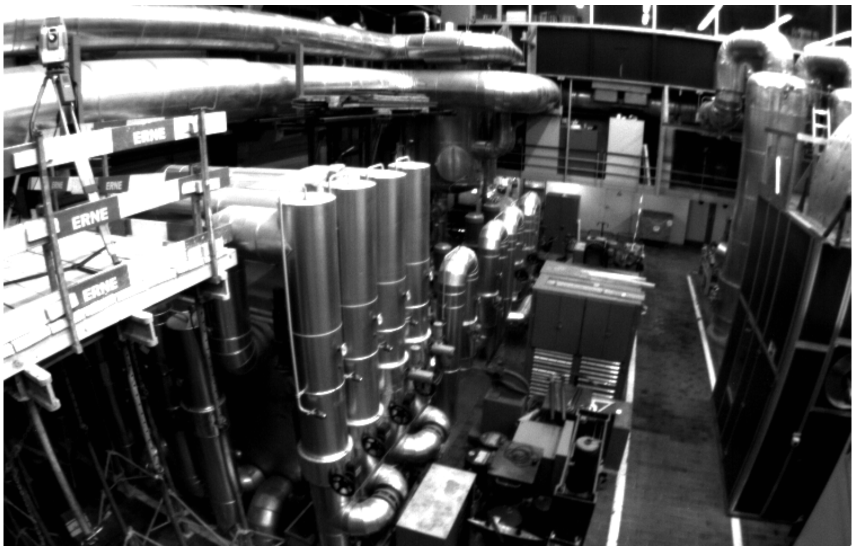
For the other robots as the agent-2, the relative similarity matrix from the agent-2's reference frame to the agent-1's reference frame $\left(\mathbf{S}_{\mathrm{L}_2\mathrm{L}_1}\right)^{-1}$ needs to be additionally multiplied with the optimal poses:

$$\begin{aligned}\hat{\mathbf{S}}^{t_i}_{\mathrm{L}_2\mathrm{C}_2} &= \left(\mathbf{S}_{\mathrm{L}_2\mathrm{L}_1}\right)^{-1}\left(\mathbf{S}_{\mathrm{GL}_1}\right)^{-1}\hat{\mathbf{S}}^{t_i}_{\mathrm{GC}_2} \\ &= \begin{bmatrix}(\mathbf{R}_{\mathrm{L}_1\mathrm{L}_2})^{\mathrm{T}}\hat{\mathbf{R}}^{t_i}_{\mathrm{GC}_2} & (\mathbf{R}^{t_i}_{\mathrm{L}_1\mathrm{L}_2})^{\mathrm{T}}\hat{\mathbf{t}}^{t_i}_{\mathrm{C}_2\mathrm{G}} - \left(\hat{\mathrm{s}}^{t_i}_{\mathrm{GC}_2}\right)^{-1}\mathrm{s}_{\mathrm{GL}_1}\mathrm{s}_{\mathrm{L}_1\mathrm{L}_2}(\mathbf{R}_{\mathrm{L}_1\mathrm{L}_2})^{\mathrm{T}}\mathbf{t}_{\mathrm{L}_2\mathrm{L}_1} \\ 0 & \left(\hat{\mathrm{s}}^{t_i}_{\mathrm{GC}_2}\right)^{-1}\mathrm{s}_{\mathrm{GL}_1}\mathrm{s}_{\mathrm{L}_1\mathrm{L}_2}\end{bmatrix} \\ &= \begin{bmatrix}\hat{\mathbf{R}}^{t_i}_{\mathrm{L}_2\mathrm{C}_2} & \hat{\mathbf{t}}^{t_i}_{\mathrm{C}_2\mathrm{L}_2} \\ 0 & \left(\hat{\mathrm{s}}^{t_i}_{\mathrm{L}_2\mathrm{C}_2}\right)^{-1}\end{bmatrix}.\end{aligned}$$

Then, the local poses $\hat{\mathbf{T}}^{t_i}_{\mathrm{L}_2\mathrm{C}_2}$ and the map points $\mathbf{X}^i_{\mathrm{L}_2}$ are updated as the agent-1's case.

5.3. System Evaluation in Two Application Scenarios

CoVOR-SLAM was evaluated in both indoor and outdoor application scenarios. First, four Machine Hall (MH) sequences of the EuRoC dataset [Burri et al., 2016] were used to test the system in an indoor environment. The original images were sequentially obtained using a single drone in a machine hall, and Fig. 5.5a shows a sample image. It was assumed that four MH sequences were obtained simultaneously using four drones to test CoVOR-SLAM using multiple robots, and the ground truth trajectory of each drone is depicted in Fig. 5.6a. To evaluate the system in an outdoor environment, sequence 00 of the KITTI dataset [Geiger et al., 2012] was used. The original images were obtained using a car in a residential area, and a sample image is shown in Fig. 5.5b. The entire trajectory was equally divided into four as depicted in Fig. 5.6b, and the assumption was that four cars traveled each part of the trajectory at the same time. Table 5.1 summarizes the total traveling time, distance, and the dimension of the datasets.


(a) A sample image of the EuRoC dataset Machine Hall (MH) sequences

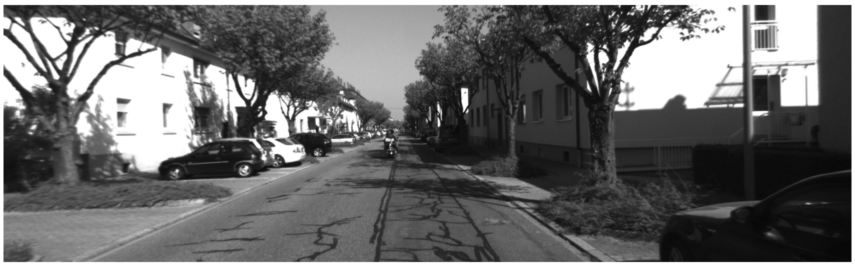
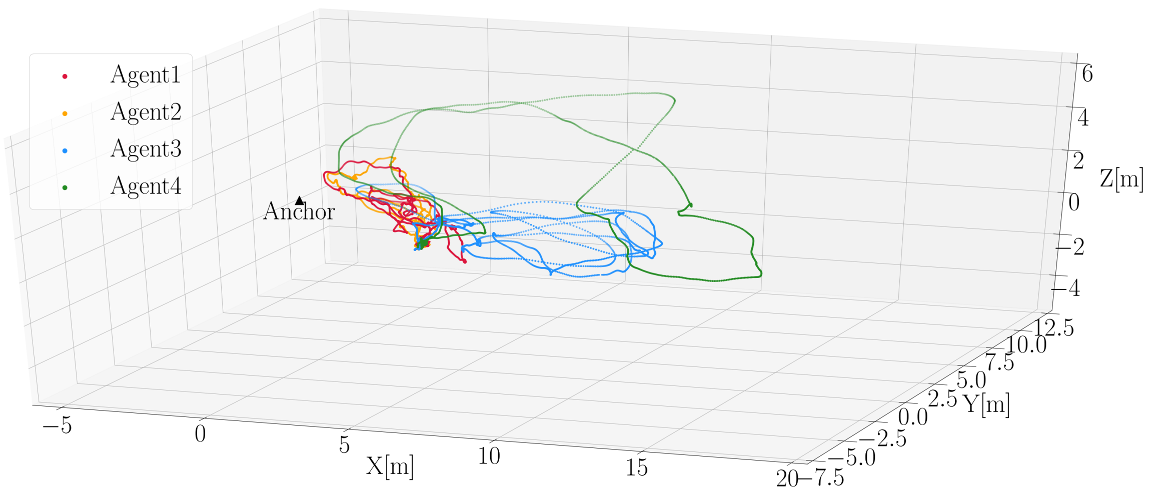
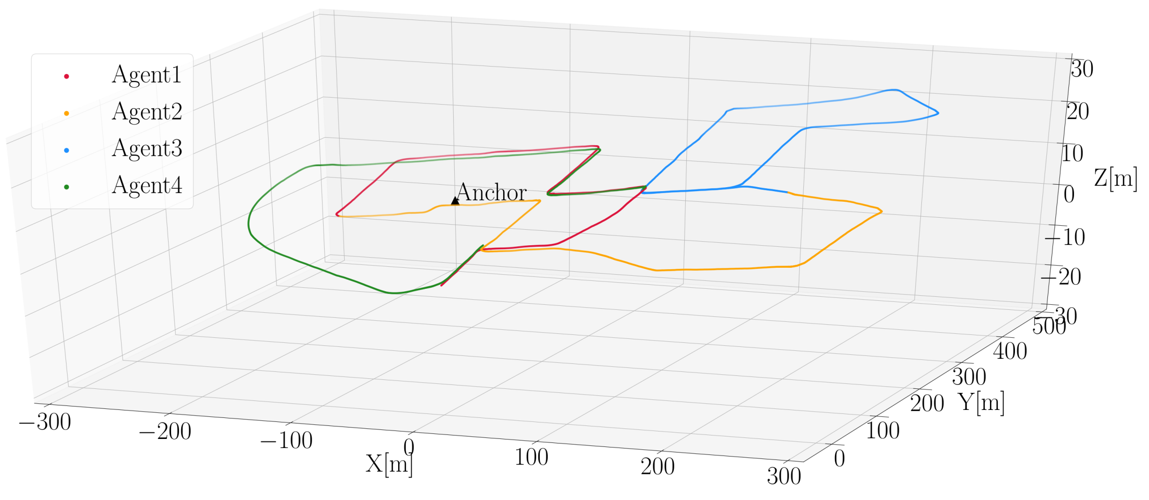

(b) A sample image of the KITTI dataset sequence 00

Figure 5.5.: Sample images of the EuRoC and KITTI datasets



(a) Four agents' ground truth trajectories from EuRoC-MH

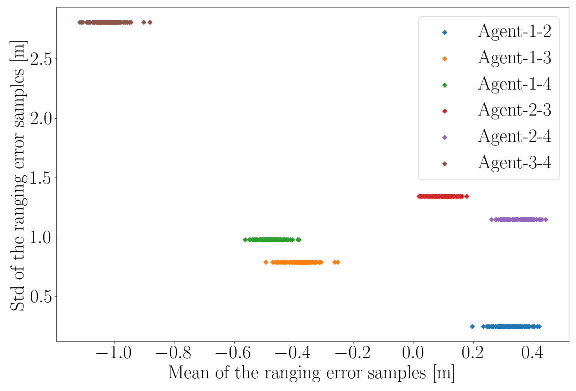


(b) Four agents' ground truth trajectories from KITTI-00

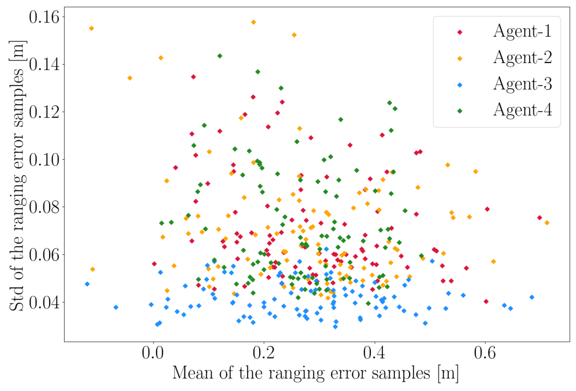
Figure 5.6.: Ground truth trajectories of the swarm robotic system with four agents

Table 5.1.: Summary of the EuRoC-MH and KITTI-00 datasets

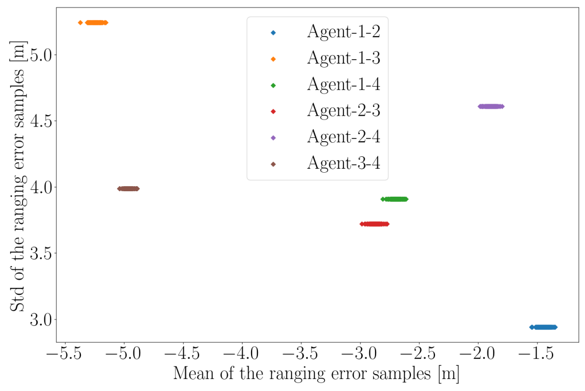
Dataset	Agent ID	Total time [s]	Total distance [m]	Dimension (X[m] × Y[m] × Z[m])
EuRoC-MH	1	181.85	80.55	7.80 × 11.20 × 2.45
	2	149.95	73.39	7.35 × 11.90 × 2.36
	3	131.50	130.85	13.04 × 9.59 × 1.99
	4	98.75	91.71	19.51 × 17.37 × 3.31
KITTI-00	1	117.56	827.65	259.16 × 375.15 × 10.81
	2	117.56	870.42	466.36 × 235.70 × 15.00
	3	117.56	941.25	187.96 × 277.08 × 12.37
	4	117.56	1081.98	340.79 × 394.28 × 13.67



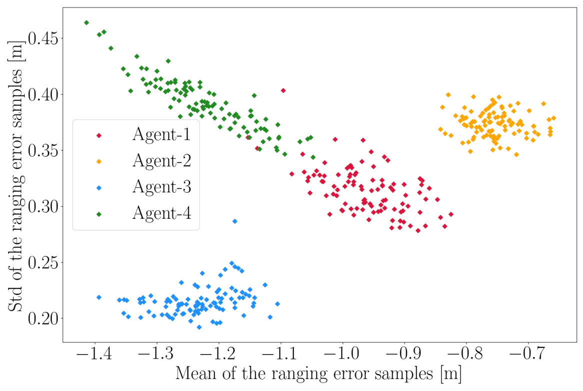
(a) The mean and std of the inter-agent ranging error samples (EuRoC-MH)



(b) The mean and std of the agent-to-anchor ranging error samples (EuRoC-MH)



(c) The mean and std of the inter-agent ranging error samples (KITTI-00)



(d) The mean and std of the agent-to-anchor ranging error samples (KITTI-00)

Figure 5.7.: The mean and standard deviation (std) values of the inter-agent and agent-to-anchor ranging error samples

Range measurements were synthetically generated including the ranging errors computed using the statistical error model derived in Appendix B because neither EuRoC-MH nor KITTI-00 datasets include range measurements. To generate agent-to-anchor range measurements, an anchor was set at $[-3, 5, 0]\text{m}^T$ for EuRoC-MH and at $[-50, 150, 10]\text{m}^T$ for KITTI-00. Since the ranging error model includes random variables, 100 sets of inter-agent and agent-to-anchor range samples were generated. Fig. 5.7 shows the mean and standard deviation values of the inter-agent and agent-to-anchor ranging error samples. The average of the mean and standard deviation values are summarized in Table 5.2. For example, the first row of this table shows that the averages of the mean and standard deviation values of the ranging error samples between agent-1 and agent-2 are 0.33m and 0.25m for EuRoC-MH, and -1.44m and 2.94m for KITTI-00.

In addition, the maximum distance was set to ensure more realistic range measurements, i.e. when robots are located beyond the maximum value, no range measurement was obtained

Table 5.2.: The average of the mean and standard derivation (std) values of the ranging error samples (EuRoC-MH and KITTI-00)

		EuRoC-MH		KITTI-00	
	Agent-ID	Avg. of mean [m]	Avg. of std [m]	Avg. of mean [m]	Avg. of std [m]
Inter-agent ranges	1-2	0.33	0.25	-1.44	2.94
	1-3	-0.38	0.79	-5.24	5.24
	1-4	-0.48	0.98	-2.69	3.91
	2-3	0.1	1.34	-2.88	3.72
	2-4	0.36	1.15	-1.89	4.61
	3-4	-1.02	2.81	-4.96	3.99
	Avg.	-0.18	1.22	-3.18	4.07
Agent-to-anchor ranges	1	0.29	0.07	-0.96	0.31
	2	0.28	0.07	-0.75	0.37
	3	0.27	0.04	-1.23	0.22
	4	0.27	0.07	-1.22	0.39
	Avg.	0.28	0.063	-1.04	0.32

Table 5.3.: The range measurement availability of the EuRoC-MH and KITTI-00 datasets

Dataset	Agent ID	Anchor ranges [%]	Inter-agent ranges [%]		
			≥ 1	$= 2$	$= 3$
EuRoC-MH	1	59.83	37.61	8.55	0.85
	2	50.47	51.40	5.61	0.93
	3	4.95	28.71	5.94	0.99
	4	15.38	20.19	6.73	0.00
	Avg.	32.66	34.48	6.71	0.69
KITTI-00	1	71.63	25.61	4.15	4.50
	2	53.99	28.12	5.43	5.11
	3	21.45	30.28	8.52	2.84
	4	57.10	23.96	5.62	1.78
	Avg.	51.04	26.99	5.93	3-56

at that location. Table 5.3 shows the range measurement availability of each agent compared to the total traveling time. For instance, agent-1 of KITTI-00 obtained range measurements to the anchor for 71.63% of its total traveling time, and it was connected to at least one robot for 25.61% of the mission. For 4.15% and 4.50% of the traveling time, agent-1 was connected to two and three other robots, respectively.

Analysis of relative positioning accuracy CoVOR-SLAM was conducted using monocular visual odometry and inter-agent ranges (without anchor-to-agent ranges) to analyze the impact of inter-agent ranges on the relative positioning accuracy. The relative position between agent-A

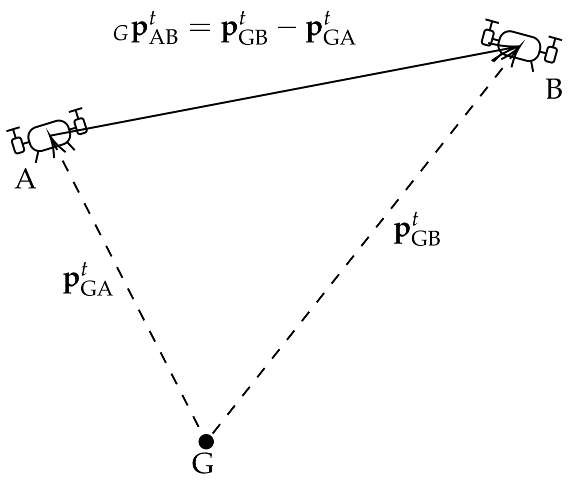
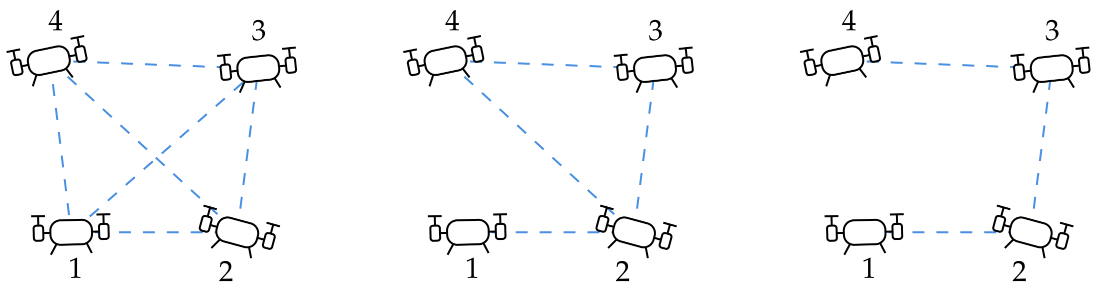

${}_{\mathrm{G}}\mathbf{p}^t_{\mathrm{AB}} = \mathbf{p}^t_{\mathrm{GB}} - \mathbf{p}^t_{\mathrm{GA}}$
B
A
$\mathbf{p}^t_{\mathrm{GB}}$
$\mathbf{p}^t_{\mathrm{GA}}$
G

Figure 5.8.: The relative position between agent-A and agent-B at t



(a) Test case 1 (b) Test case 2 (c) Test case 3

Figure 5.9.: Three test cases of inter-agent connectivity options

and agent-B at t is defined as

$$
{}_{\mathrm{G}}\mathbf{p}^t_{\mathrm{AB}} = \mathbf{p}^t_{\mathrm{GB}} - \mathbf{p}^t_{\mathrm{GA}}, \tag{5.88}
$$

as can be seen in Fig. 5.8.

In addition, three test cases were defined as shown in Fig. 5.9. All six possible connections between the four agents were enabled in test case 1. In test case 2, it was assumed that the communication links between agent-1 and agent-3 were lost, as well as the connections between agent-1 and agent-4. Agent-2 and agent-4 were further disconnected in test case 3.

Fig. 5.10 shows the relative trajectories estimated using monocular visual odometry produced with the KITTI-00 images and all available inter-agent range measurements between the four agents (MonoVO+Inter(Case1), blue). The ground truth relative trajectories are gray, and the relative trajectories estimated using only monocular VO are yellow (MonoVO(gtScale0)). Since the positions estimated using monocular VO are up-to-scale, they are scaled with the first keyframe's ground truth values ($\mathrm{s}_{\mathrm{A}}^{t_1}$ and $\mathrm{s}_{\mathrm{B}}^{t_1}$), before the relative positions are computed:

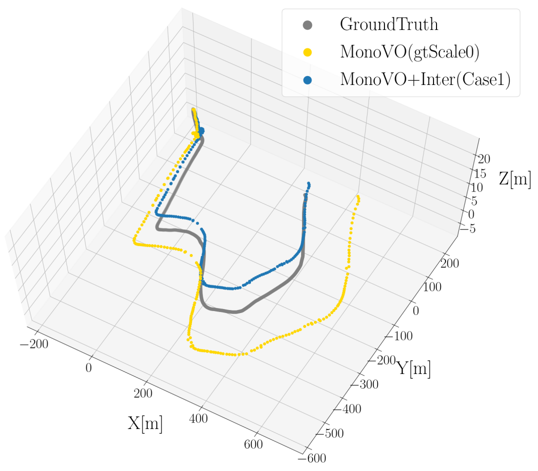
$$ {}_{\mathrm{G}}\mathbf{p}_{\mathrm{AB}}^{t} = \mathrm{s}_{\mathrm{B}}^{t_1}\,\mathbf{p}_{\mathrm{GB}}^{t} - \mathrm{s}_{\mathrm{A}}^{t_1}\,\mathbf{p}_{\mathrm{GA}}^{t}. \tag{5.89}$$

This figure shows that the relative positioning errors are substantially reduced by fusing inter-agent range measurements with monocular visual odometry measurements, even when the agents are connected less than 50% of the entire traveling time, as summarized in Table 5.3.

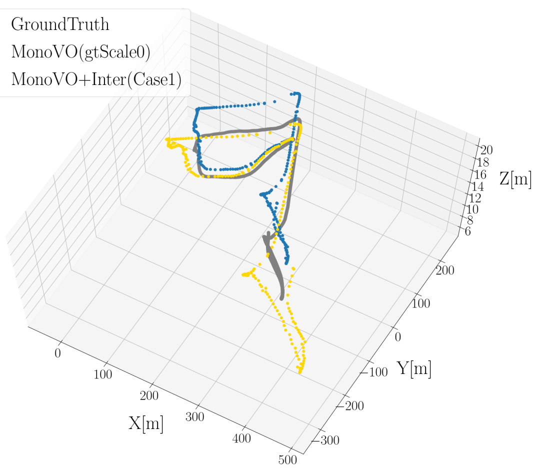
Fig. 5.11 shows the simulation results obtained using all the 100 inter-agent ranging sequences as the empirical CDF. Four agents' monocular VO measurements were generated using EuRoC-MH for Fig. 5.11a, and 100 range measurements were generated employing each setting defined in Fig. 5.9. Using these monocular VO and inter-agent ranges, CoVOR-SLAM was run 100×3 test cases, and 100 RMSE values of each simulation case are depicted as an empirical CDF in Fig. 5.11a. The same procedure was carried out using KITTI-00 to generate Fig. 5.11b.

As can be seen in Fig. 5.11b, CoVOR-SLAM obtains much smaller relative positioning errors compared to MonoVO(gtScale0), without scaling the position estimates with the first keyframe's ground truth scale. This result shows that CoVOR-SLAM can solve not only the scale ambiguity problem of monocular VO, but it can also effectively mitigate the positioning errors accumulated over time, although the agents are connected to each other less than 50% of the traveling time (see Table 5.3). CoVOR-SLAM reduces the relative positioning errors most effectively when all the possible communication links are available between the agents (MonoVO+Inter(Case1), blue). Although the inter-agent connections are partially lost as test case 2 and 3 in Fig. 5.9, CoVOR-SLAM can still provide much less relative positioning errors compared to MonoVO(gtScale0), as shown with the orange line (MonoVO+Inter(Case2)) and the green line (MonoVO+Inter(Case3)) in Fig. 5.11b. Moreover, the RMSE values are not increased significantly from MonoVO+Inter(Case1).

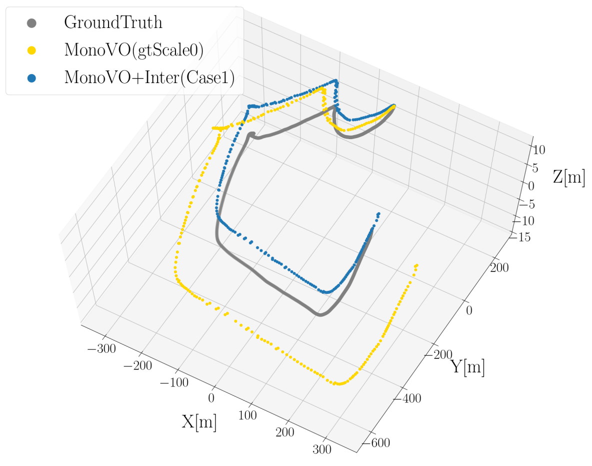
Fig. 5.11a shows the empirical CDF obtained with EuRoC-MH. As KITTI-00, CoVOR-SLAM can provide more accurate relative position estimates when more agents can obtain the inter-agent communication links. Unlike KITTI-00, monocular VO can accurately estimate the shape of each agent's trajectory using EuRoC-MH because many differentiable features are available in the machine hall as shown in Fig. 5.5a. However, note that the positions estimated using monocular VO are still up-to-scale, and are scaled with the ground truth values, as Eq. (5.89), while CoVOR-SLAM can accurately estimate the ambiguous scale of monocular VO, although



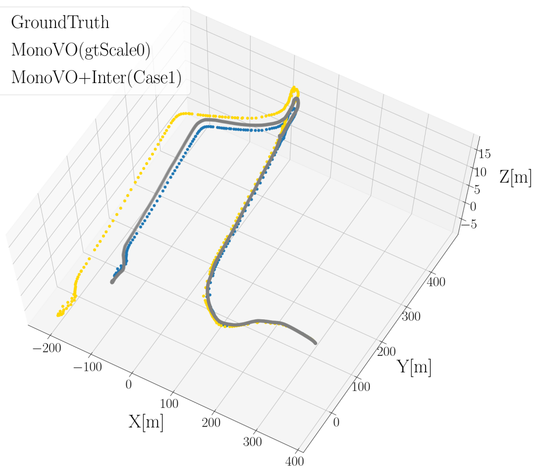
(a) The relative trajectories from agent-1 to agent-2



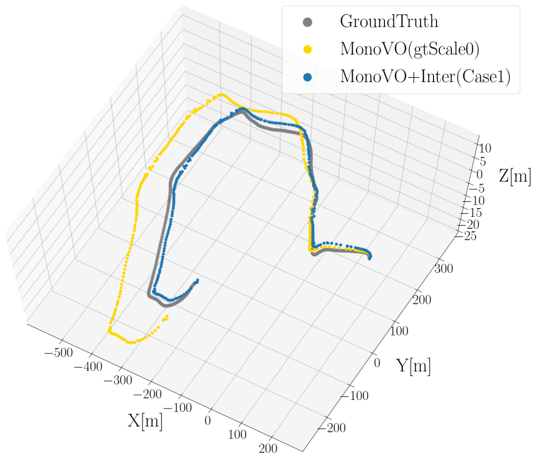
(b) The relative trajectories from agent-1 to agent-3



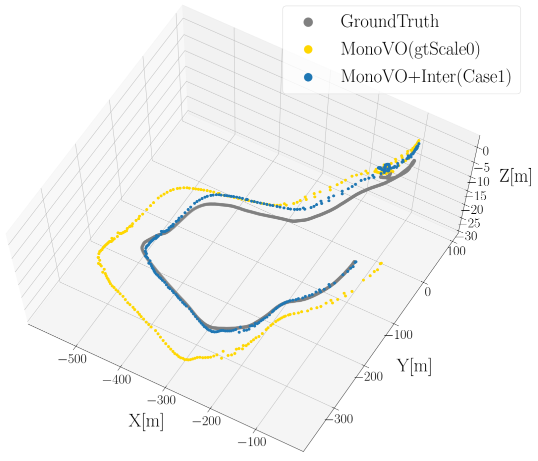
(c) The relative trajectories from agent-1 to agent-4



(d) The relative trajectories from agent-2 to agent-3

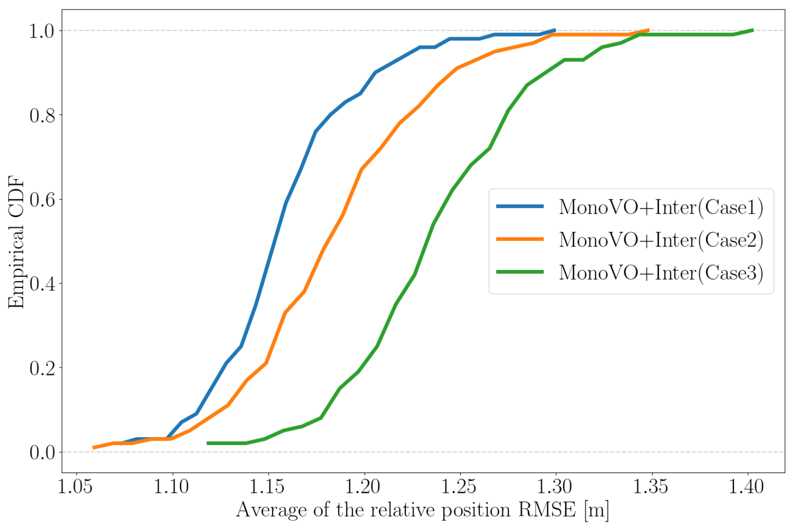


(e) The relative trajectories from agent-2 to agent-4

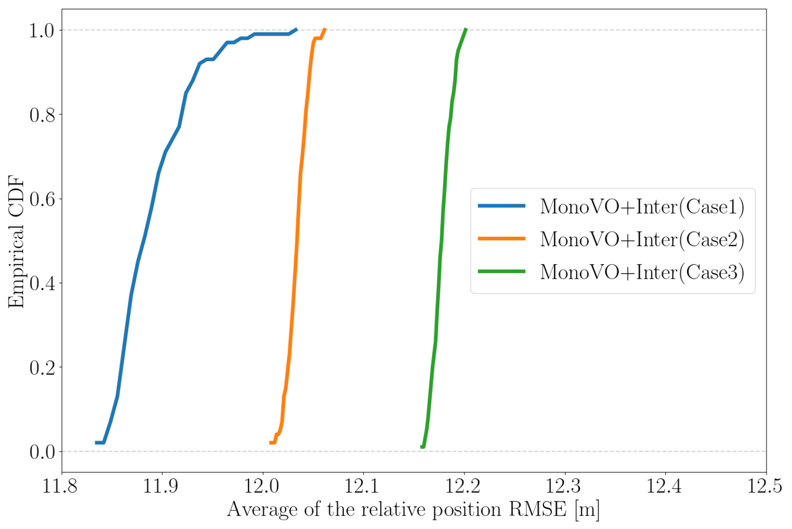


(f) The relative trajectories from agent-3 to agent-4

Figure 5.10.: The relative trajectories between the agents of KITTI-00. The ground truth is depicted in gray. The trajectories estimated by monocular VO are yellow, and the trajectories estimated by fusing monocular VO and inter-agent range measurements are blue



(a) The CDF of the relative position RMSE obtained using CoVOR-SLAM run with monocular VO produced with EuRoC-MH and inter-agent ranges (The relative position RMSE obtained using MonoVO(gtScale)= 0.77m)

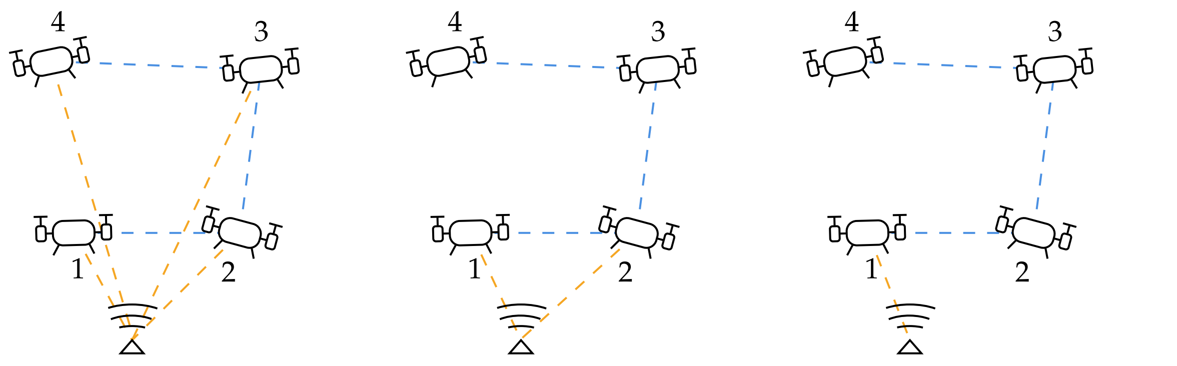


(b) The CDF of the relative position RMSE obtained using CoVOR-SLAM run with monocular VO produced with KITTI-00 and inter-agent ranges (The relative position RMSE obtained using MonoVO(gtScale)= 111.75m)

Figure 5.11.: The empirical CDF of the average relative position RMSE obtained using CoVOR-SLAM run with the four agents' monocular visual odometry measurements and inter-agent range measurements generated in line with the three test case settings

inter-agent range measurements are biased and include noise (see Table 5.2) and most of the inter-agent connections are available less than 50% of the traveling time (see Table 5.3).

Analysis of absolute positioning accuracy CoVOR-SLAM was executed with monocular visual odometry, inter-agent as well as agent-to-anchor range measurements to evaluate



(a) To1234: all the agents obtain agent-to-anchor-range measurements

(b) To12: agent-1 and agent-2 obtain agent-to-anchor range measurements

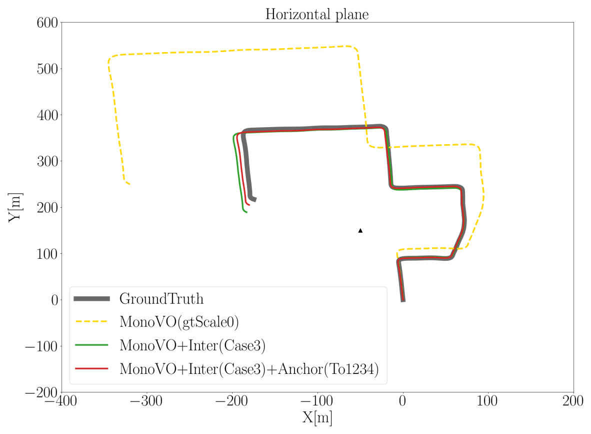
(c) To1: only agent-1 obtains agent-to-anchor range measurements

Figure 5.12.: Three setups of agent-to-anchor ranging

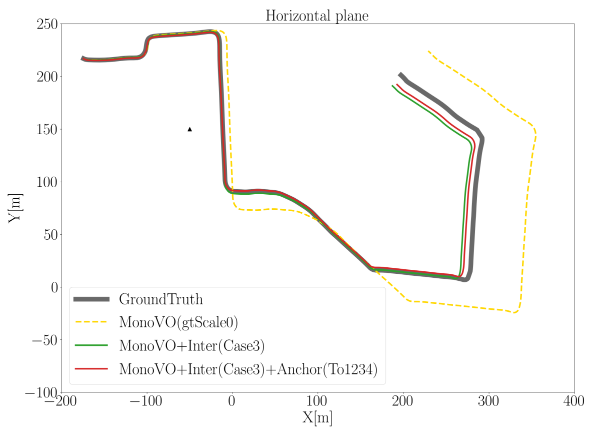
the system in terms of absolute positioning accuracy. For the simulation, inter-agent range measurements were generated employing test case 3 in Fig. 5.9c. Ranges between an anchor and four agents were generated using the test cases shown in Fig. 5.12. In the first setting (To1234), all fours agents are connected to the anchor, and agent-1 and agent-2 are connected to the anchor in the second setup (To12). In the last setting (To1), only agent-1 obtains agent-to-anchor range measurements.

Fig. 5.13 shows the four agents' absolute positions in the horizontal plane estimated using CoVOR-SLAM. For this simulation, the four agents' visual odometry measurements were produced with the KITTI-00 images. Ranges between the agents were generated employing test case 3 in Fig. 5.9c, and agent-to-anchor ranges were generated between the anchor and all four agents as in Fig. 5.12a. The positions estimated using monocular VO, inter-agent ranges, and agent-to-anchor ranges are red (MonoVO+Inter(Case3)+Anchor(To1234)), and the positions estimated using monocular VO and inter-agent ranges are green (MonoVO+Inter(Case3)). The yellow lines show the trajectories estimated using only monocular VO and scaled with the first keyframe's ground truth values, while the ground truth trajectories are gray.

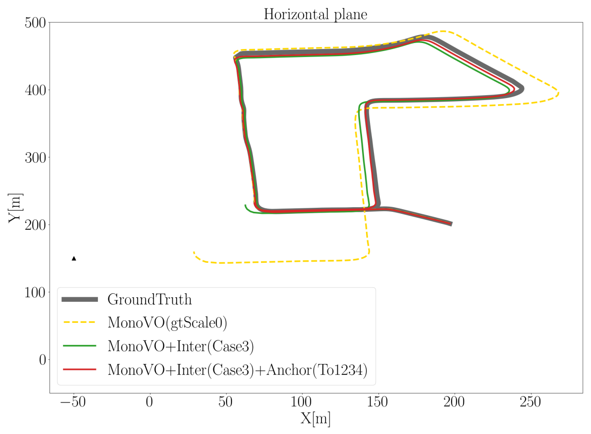
As shown in Fig. 5.13, all four trajectories estimated using MonoVO+Inter(Case3) are much closer to the ground truth, compared to the trajectories estimated MonoVO(gtScale0). This result shows that the absolute positioning error of monocular VO can be substantially reduced by fusing inter-agent ranges with monocular VO. By exploiting agent-to-anchor range measurements in addition to the inter-agent ranges, CoVOR-SLAM can estimate the agents' absolute trajectories most accurately, i.e. the trajectories estimated using MonoVO+Inter(Case3)+Anchor(To1234) are closest to the ground truth trajectories for all four



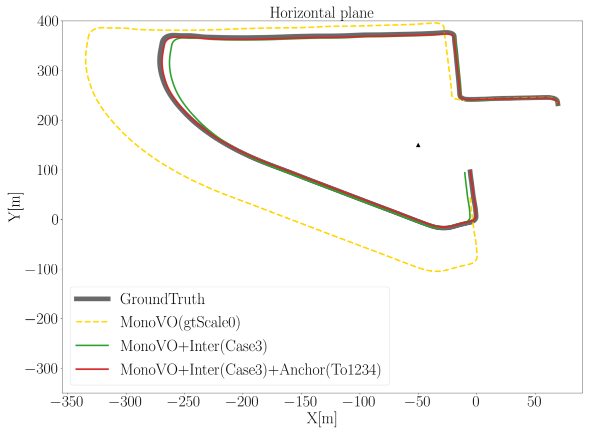
(a) Agent-1's horizontal trajectory estimates



(b) Agent-2's horizontal trajectory estimates



(c) Agent-3's horizontal trajectory estimates



(d) Agent-4's horizontal trajectory estimates

Figure 5.13.: Four agents' horizontal trajectories estimated using only monocular VO with the KITTI-00 images (yellow); estimated using monocular VO and inter-agent ranges (green); and estimated using monocular VO, inter-agent ranges, and agent-to-anchor ranges (red). The ground truth trajectories are depicted in gray

agents. In addition, map points are accurately estimated using CoVOR-SLAM as seen in Fig. 5.14. Map points should be generated along the ground truth trajectory with the KITTI-00 images since the images were obtained using a front-facing camera on a car in a residential area. As a result, the map points shown in this figure can be considered as accurate estimates, although the ground truth map points are unavailable.

Fig. 5.15 shows the absolute position RMSE values as the empirical CDF, which are obtained using both the EuRoC-MH and KITTI-00 datasets and all the 100×3 agent-to-anchor ranges. For Fig. 5.15a, monocular VO measurements were generated using EuRoC-MH. Inter-agent ranges were generated employing test case 3 in Fig. 5.9c, and 100×3 samples of agent-to-anchor ranges were generated using the three settings shown in Fig. 5.12. Using these measurements, CoVOR-SLAM was run 100×3 times, and the average RMSE values of the four agents' absolute

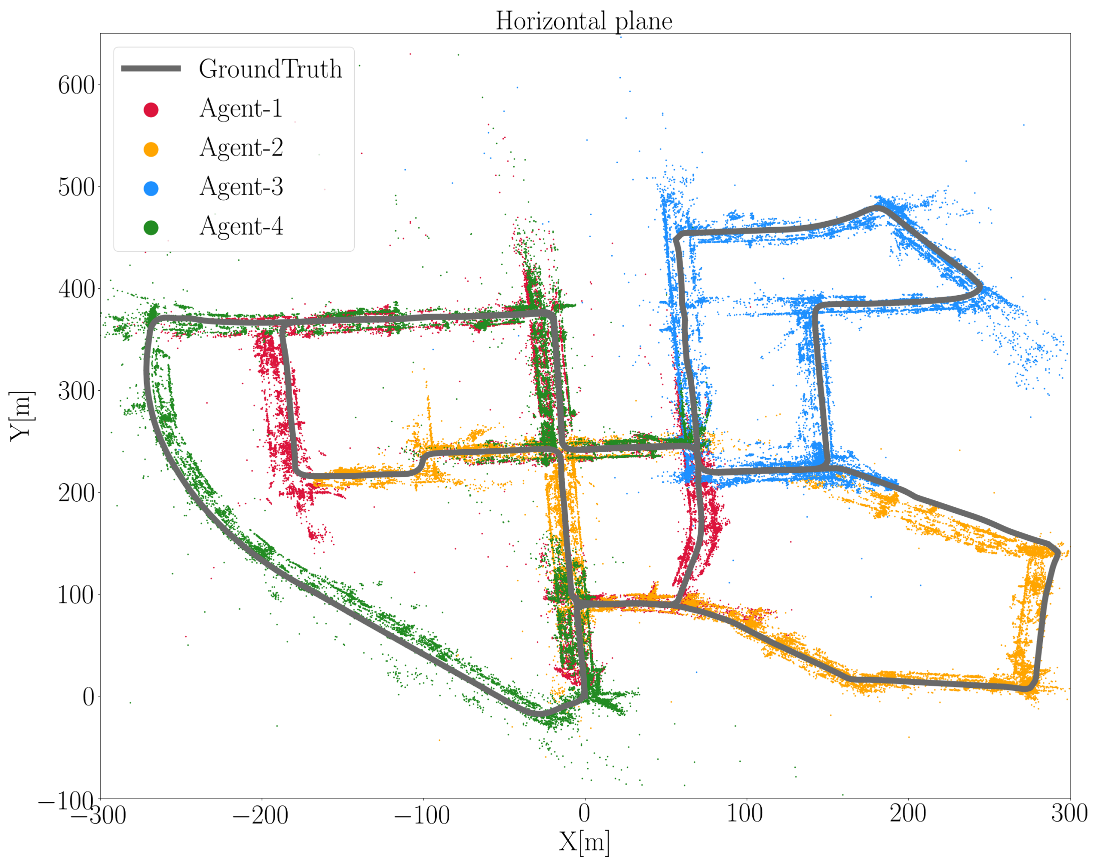
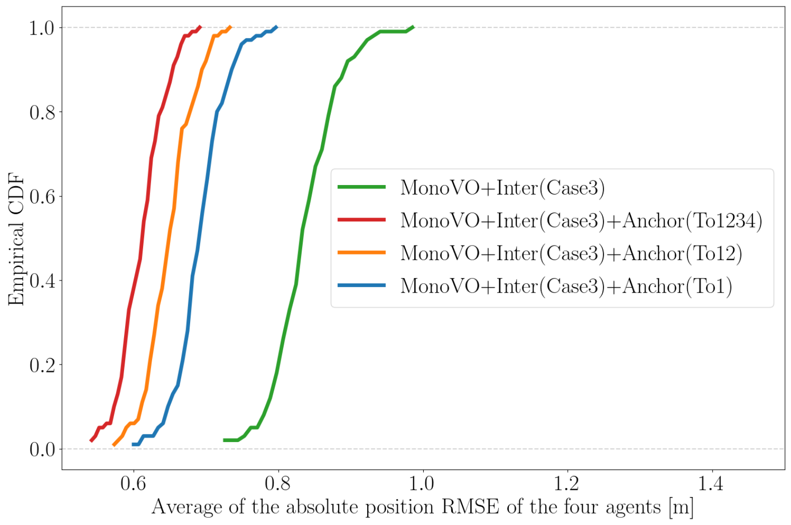


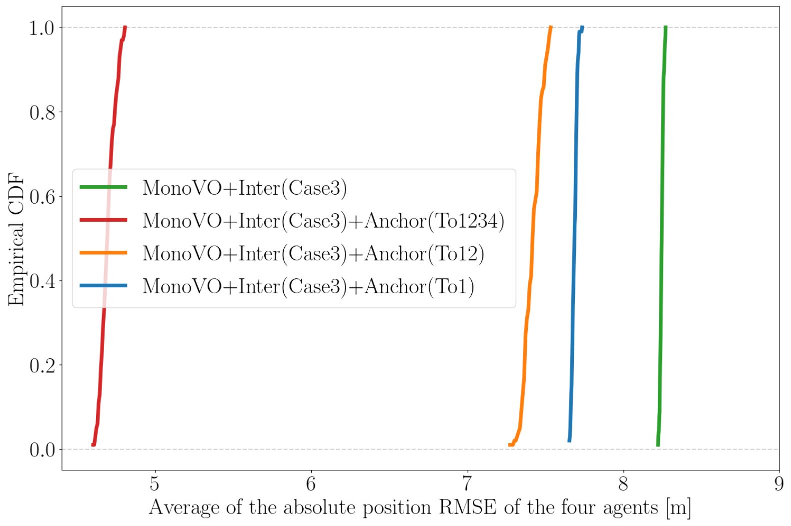
Figure 5.14.: Map points estimated employing four agents using CoVOR-SLAM run with monocular VO produced with the KITTI-00 images, inter-agent ranges generated employing test case 3's setup, and ranges between the anchor and all the four agents (MonoVO+Inter(Case3)+Anchor(To1234))

position estimates are plotted as the empirical CDF in Fig. 5.15a. Additionally, the RMSE values obtained using CoVOR-SLAM run with monocular VO and only inter-agent ranges are depicted in green (MonoVO+Inter(Case3)). Fig. 5.15b shows the average RMSE values of the four agents' absolute positions estimated using the same procedure but with the KITTI-00 dataset, instead of the EuRoC-MH dataset.

As shown in Fig. 5.15b, all the RMSE values obtained with CoVOR-SLAM are smaller than MonoVO(gtScale0) (66.15m), even though all the agents are connected to the anchor less than 60% of their traveling time, as shown in Table 5.3. CoVOR-SLAM can obtain the smallest RMSE values when all four agents are connected to the anchor (MonoVO+Inter(Case3)+Anchor(To1234), red), but it can still achieve more accurate absolute position estimates than MonoVO(gtScale0) as well as MonoVO+Inter(Case3) when only agent-1 is connected to the anchor (MonoVO+Inter(Case3)+Anchor(To1), blue). This is because the other agents can benefit from the anchor connections using inter-agent connections.



(a) The CDF of the average absolute position RMSE values obtained using CoVOR-SLAM with EuRoC-MH (The absolute position RMSE obtained with MonoVO(gtScale)= 0.26m)



(b) The CDF of the average absolute position RMSE values obtained using CoVOR-SLAM with KITTI-00 (The absolute position RMSE obtained with MonoVO(gtScale)= 66.15m)

Figure 5.15.: The empirical CDF of the average RMSE values of the four agents' absolute positions estimated using CoVOR-SLAM operated with monocular visual odometry, inter-agent ranges generated employing the setting of test case 3, and agent-to-anchor ranges generated employing the three connectivity setup

As aforementioned, monocular visual odometry can accurately estimate the shape of each agent's trajectories with EuRoC-MH using of the many differentiable features in the machine hall. However, note again that the absolute scale of the positions cannot be estimated only with monocular visual odometry in the real world, so the positions estimated using monocular visual odometry are manually scaled with the first keyframe's ground truth value for comparison, whereas CoVOR-SLAM can estimate the absolute scale using the range measurements without manually scaling the position estimates.

Analysis of the communication requirements Fig. 5.16 shows the data size transmitted with different numbers of keyframe messages to run CoVOR-SLAM (CoVOR-SLAM, red). It was assumed that both inter-agent and agent-to-anchor range measurements were included in the keyframe messages. This figure also illustrates the data size transmitted for two state-of-the-art collaborative VSLAM with inter-agent loop closures: centralized CCM-SLAM (Upper Bound (UB) in dark blue, and Lower Bound (LB) in sky blue) and decentralized dSLAM (green). As can be seen in this figure, CCM-SLAM requires the greatest communication capability because all agents transmit their local map data to the server and only the server carries out the computationally demanding tasks of inter-agent loop closures, such as loop detection, map fusion, and global map optimization. dSLAM successfully reduced about 44% of the data compared to the CCM-SLAM's lower bound to run collaborative VSLAM in a decentralized manner. CoVOR-SLAM requires much less communication capability compared to dSLAM, reducing the data size by about 95%. This is because visual information can be completely discarded from the keyframe messages. Consequently, CoVOR-SLAM can be useful in those environments where no adequate communication network is available, e.g. in deep sea and extraterrestrial areas. Moreover, a powerful server is not necessary to run the data fusion of monocular VO and range measurements. Therefore, the system architecture can be chosen flexibly (whether centralized or decentralized), depending on the mission and environments.

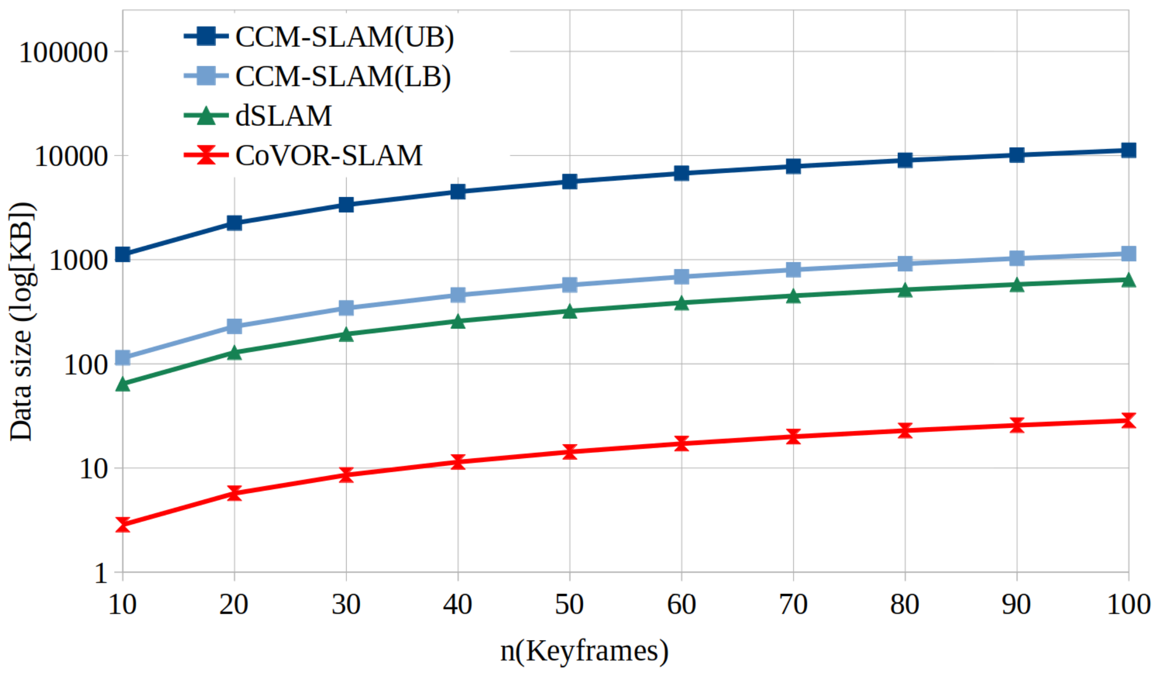


Figure 5.16.: The data size transmitted for collaborative VSLAM with inter-agent loop closures: centralized CCM-SLAM (UB in dark blue and LB in sky blue) and decentralized dSLAM (green). The data size transmitted for CoVOR-SLAM is red

6. Summary and Conclusions

6.1. Summary

This dissertation developed a cooperative SLAM method using visual and ranging sensors for swarm robotic systems to explore areas where GNSS is unreliable or unavailable. VSLAM was selected to estimate each agent's poses and to map environments because cameras are cost-effective onboard sensors that provide abundant information about environments observed without complex infrastructures. VSLAM can be divided into two parts, as introduced in Chapter 2 - the front-end and back-end. In the front-end, images (pixels) are converted into visual measurements, which are used to estimate the camera poses and map point coordinates by tracking and local mapping in the back-end.

However, tracking and local mapping are dead-reckoning processes, so the estimation errors of the camera poses and map point coordinates accumulate over time without limit. The loop closing technique can effectively mitigate the errors, using the map point matches between the current keyframe and the keyframes stored in the map database (loop information). Nevertheless, substantial computing power is required to continuously compare the map points between the current keyframe and the database as well as to process the global map with the loop information once a loop is detected. Furthermore, robots have to travel back to areas previously observed to search for a loop, which significantly constrains their movements and the whole mission planning. In addition, the loop closing method cannot solve the scale ambiguity problem of monocular VSLAM.

In Chapter 3, a SLAM method using visual odometry and range measurements (VOR-SLAM) was proposed to accurately estimate the camera poses and map points without requiring significant computing power. The simplest setup (a single robot and a reference point) was used to introduce the concept. Each agent's poses and map points were estimated using monocular VO or SLAM. Instead of applying the loop closing method, the graph-based data fusion of visual odometry and range measurements was used to mitigate estimation errors and to resolve scale ambiguity, which is inherent to monocular VSLAM. VOR-SLAM was analyzed in terms of sensitivity to the ranging error and availability, using public image datasets and range measurements synthetically generated using a statistical ranging error model. The simulation results showed that VOR-SLAM can estimate the camera poses and map points more accurately than monocular VO, even when range measurements are noisy and biased and only available less than 50% of the mission time. Moreover, VOR-SLAM requires much less processing power compared to loop closures, yet can still achieve comparable positioning accuracy without scale ambiguity. VOR-SLAM was also tested using real experimental data obtained using a camera and UWB modules mounted on the rover system developed by DLR-KN in two outdoor environments (a football field and a gravel pit). The experimental results showed that VOR-SLAM accurately estimates the rover's poses and map points using real range measurements obtained with low-cost UWB sensors.

After conducting intensive analyses using a single robot, Chapter 4 reviewed the state-of-the-art collaborative VSLAM research for swarm robotic systems. This chapter focused on the approaches using the inter-agent loop closing, the most widely studied and tested methods. The inter-agent loop closing can substantially reduce the estimation errors of the relative poses between robots, using the map point matches observed by different robots (inter-agent loops). Computationally demanding tasks of inter-agent loop closures, such as inter-agent loop detection and multi-map fusion, can be processed using a central server. In this centralized system architecture, each robot's onboard computer only needs to estimate its local poses and map points, which does not require large computational capability. However, robots need to remain in the areas where they can communicate with the server, which is impractical in many cases. Thus, one of the onboard computers can be used to process inter-agent loop closures in a distributed manner, using only neighboring robots' map data.

Cooperative SLAM for swarm robotic systems using visual odometry and range measurements (CoVOR-SLAM) was proposed in Chapter 5 to reduce even further the computational requirement. In CoVOR-SLAM, monocular VO or VSLAM was used to estimate each robot's local poses and map points, and then all available range measurements (both agent-to-anchor and inter-agent ranges) were used to mitigate both relative and absolute positioning errors. CoVOR-SLAM was evaluated in indoor and outdoor application scenarios using real images and ranges synthetically generated using a statistical ranging error model. The simulation results showed that CoVOR-SLAM substantially reduces the robots' positioning and mapping errors without scale ambiguity in both indoor and outdoor environments. Moreover, CoVOR-

SLAM requires much less computing power than the state-of-the-art inter-agent loop closures because visual information is completely excluded from the map fusion process. The system architecture of swarm robotic systems can be flexibly chosen (centralized or decentralized), depending on the mission and environment, since a powerful server is not required to execute CoVOR-SLAM. Furthermore, CoVOR-SLAM requires much fewer communication network capabilities compared to inter-agent loop closures. Thus, CoVOR-SLAM is useful in environments, such as deep sea and extraterrestrial areas, where a sufficient communication network is unavailable. In addition, a large number of robots can participate in a mission since each robot transmits only small amounts of data to the processing unit when using CoVOR-SLAM.

6.2. Conclusions; Applications and Future Work

CoVOR-SLAM can be applied in various environments where GNSS is unreliable or unavailable, without complex infrastructures to obtain range measurements. For instance, the communication channels between the robots can be used to obtain inter-agent range measurements, and signals transmitted from a base station (that are already available in mission areas) can be employed to obtain agent-to-anchor range measurements. The following are application scenarios of CoVOR-SLAM using various ranging sources:

- Urban applications: In deep urban areas, GNSS is unreliable, but agent-to-anchor ranges can be obtained by employing the signals from cellular networks (e.g. 5G). CoVOR-SLAM can be used to estimate the robots' poses and map points with onboard cameras and range measurements.
- Indoor applications: GNSS is unreliable or unavailable in indoor areas, but CoVOR-SLAM can be applied using ranges obtained using e.g. Wi-Fi signals and bluetooth.
- Extraterrestrial applications: Robots can use onboard cameras for self-localization in extraterrestrial areas where GNSS is unavailable (e.g. the Curiosity Mars rover from NASA). In such environments, radio-based ranging sensors, such as UWB modules, can be employed to obtain agent-to-anchor range measurements for CoVOR-SLAM. Inter-agent ranges can be also obtained without additional hardware, using communication channels between the robots.
- Deep sea applications: Submarines cannot obtain GNSS signals when conducting a mission. Instead of traveling back to the surface to obtain GNSS signals, which consumes

significant time and energy, CoVOR-SLAM can be applied to accurately estimate the their positions and map the environments.

In this dissertation, CoVOR-SLAM was tested using only a single type of platform. Further system tests could be carried out using different types of platforms. For example, when drones and rovers cooperatively conduct a mission in a swarm robotic system, ranges between the robots can be easily obtained using the inter-agent communication channels, though it is hard to detect inter-agent loops (since two robots observe the scenes from different angles). In such cases, CoVOR-SLAM can estimate the robots' poses and map points much more accurately than collaborative VSLAM using inter-agent loop closing. Furthermore, additional sensors can be integrated into the onboard system. CoVOR-SLAM has a simple framework, so other sensors can be easily integrated without significant system modification. For example, IMU can be used with cameras (Visual-INertial System (VINS)) to obtain more accurate odometry measurements without scale ambiguity, before fusing the odometry with range measurements.

A. 3D Motion Description with Lie Groups and Algebras

The Lie groups and algebra [Eade, 2013] are used to describe the rigid camera motions for VSLAM, which are combinations of translation and rotation motion. The matrices in Lie groups can be inverted, differentiated, and interpolated, so they satisfy the requirements to describe the motions. In this section, the general properties of Lie groups and algebras are explained, and then the three subgroups, SO(3), SE(3), and Sim(3) are detailed, including the differentiation and Jacobian computations of each group's action.

A.1. General Properties

Fig. A.1 illustrates the robot's translational motion from point A to B with a rotation around the Z-axis. A 4×4 pose matrix in Lie groups can be used to describe the combined rigid motion:

$$\mathbf{T}_{\text{AB}} = \left[\begin{array}{ccc|c} \cos\theta & -\sin\theta & 0 & x \\ \sin\theta & \cos\theta & 0 & y \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right] = \begin{bmatrix} \mathbf{R}_{\text{AB}} & \mathbf{t}_{\text{BA}} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix},$$

where $\mathbf{R}_{\text{AB}}$ is the rotation matrix from B to A around the Z-axis, and $\mathbf{t}_{\text{BA}}$ is the translation vector from B to A.

A $n \times n$ matrix in a Lie group $\mathbf{T}$ and the $n \times n$ matrix in the corresponding Lie algebra Ξ have the following exponential relationship:

$$\mathbf{T} = \exp \Xi = \lim_{n\to\infty} \left(\mathbf{I} + \frac{\Xi}{n}\right)^n = \sum_{k=0}^{\infty} \frac{\Xi^k}{k!}. \tag{A.90}$$

The matrix Ξ in the Lie algebra is a linear combination (vector space) of a tangent vector $\boldsymbol{\xi} \in \mathbb{R}^n$

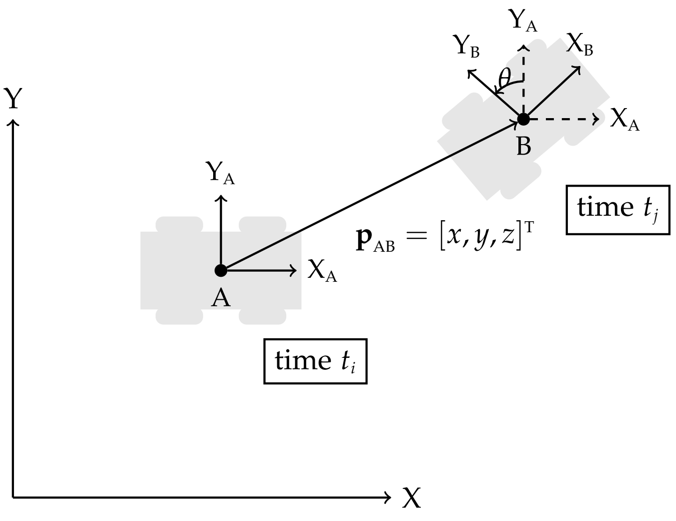


Figure A.1.: An example of a rover's simultaneous rotational and translational motion. The rotation angle from A to B with respect to the Z-axis is θ, and the position vector from A to B is $\mathbf{p}_{AB}$.

generated using the Lie group generators $\mathbf{G}_i$:

$$\Xi = \sum_i^n \xi_i \mathbf{G}_i. \tag{A.91}$$

The adjoint map Imagine that a vector $\mathbf{x}$ defined in the global frame is converted to the local frame using the matrix transformation $\mathbf{T}$, i.e. $\mathbf{x}' = \mathbf{T}\mathbf{x}$. The adjoint mapping,

$$\operatorname{Adj}(\mathbf{T})\,\mathbf{A} = \mathbf{T}\mathbf{A}\,(\mathbf{T})^{-1},$$

transforms the vector $\mathbf{x}'$ back to the global frame, and then convert the vector $(\mathbf{T})^{-1}\mathbf{x}'$ using the matrix transformation $\mathbf{A}$. Finally, the vector $\mathbf{A}\,(\mathbf{T})^{-1}\mathbf{x}'$ is transformed to the global frame using $\mathbf{T}$. When $\mathbf{A}$ can be represented with a matrix Ξ in the Lie algebra as $\mathbf{A} = \exp \Xi$,

$$\begin{aligned} \operatorname{Adj}(\mathbf{T})\,\mathbf{A} &= \operatorname{Adj}(\mathbf{T}) \exp \Xi \\ &= \mathbf{T} \exp \Xi\,(\mathbf{T})^{-1} \\ &= \exp\left(\operatorname{Adj}(\mathbf{T})\,\Xi\right). \end{aligned} \tag{A.92}$$

If the matrix $\mathbf{T}$ is also in the Lie group,

$$\operatorname{Adj}(\mathbf{T})\,\Xi \triangleq \mathbf{T}\Xi\,(\mathbf{T})^{-1}. \tag{A.93}$$

The following relationship can be derived, when Eq. (A.93) is substituted to Eq. (A.92):

$$\mathbf{T}\exp\Xi\,(\mathbf{T})^{-1} = \exp(\mathrm{Adj}\,(\mathbf{T})\,\Xi) = \exp\left(\mathbf{T}\Xi\,(\mathbf{T})^{-1}\right). \tag{A.94}$$

Derivative of the group actions The measurement function of VSLAM is often modeled with a camera matrix $\mathbf{T}$ defined in the Lie group and a m-dimensional vector $\mathbf{p} \in \mathbb{R}^m$ as

$$\mathbf{h}(\mathbf{T}) = \mathbf{T}\mathbf{p}.$$

which is called a group action employing the matrix $\mathbf{T}$. When the camera matrix is changed with a small difference $\Delta\Xi$, the measurement function can be written as

$$\begin{aligned}
\mathbf{h}(\mathbf{T}\exp\Delta\Xi) &= \mathbf{T}\exp\Delta\Xi\mathbf{p} && \text{(A.95)}\\
&\approx \mathbf{T}(\mathbf{I}+\Delta\Xi)\mathbf{p}\\
&= \mathbf{T}\mathbf{p} + \mathbf{T}\Delta\Xi\mathbf{p}\\
&= \mathbf{T}\mathbf{p} + \mathbf{T}\sum_j\sum_i \mathbf{G}_{\mathrm{i}}(:,j)\mathbf{p}(j)\Delta\boldsymbol{\xi}\\
&= \mathbf{T}\mathbf{p} + \mathbf{T}\mathbf{F}(\mathbf{p})\Delta\boldsymbol{\xi}, && \text{(A.96)}
\end{aligned}$$

where $\mathbf{G}_{\mathrm{i}}(:,j)$ is the j-th column of the matrix $\mathbf{G}_{\mathrm{i}}$, and $\mathbf{F}(\mathbf{p}) \triangleq \sum_j\sum_i \mathbf{G}_{\mathrm{i}}(:,j)\mathbf{p}(j)$. To linearize Eq. (A.95), the following relationship is used:

$$\begin{aligned}
\exp\mathbf{A} = \sum_{n=0}^{\infty}\frac{\mathbf{A}^n}{n!} &= \mathbf{I}+\mathbf{A}+\frac{\mathbf{A}^2}{2!}+\frac{\mathbf{A}^3}{3!}+\cdots\\
&\approx \mathbf{I}+\mathbf{A}.
\end{aligned}$$

Eq. (A.96) can be differentiated with respect to the twist coordinates $\boldsymbol{\xi}$ as

$$\frac{\partial\mathbf{h}(\mathbf{T})}{\partial\Delta\boldsymbol{\xi}} = \mathbf{T}\mathbf{F}(\mathbf{p}). \tag{A.97}$$

Derivative of the inverse group actions When the measurement function is inverse group action as

$$\mathbf{h}(\mathbf{T}) = (\mathbf{T})^{-1}\,\mathbf{p},$$

it can be expanded as

$$\begin{aligned}\mathbf{h}((\mathbf{T}\exp\Delta\Xi)^{-1}) &= (\mathbf{T}\exp\Delta\Xi)^{-1}\,\mathbf{p}\\ &= \exp(-\Delta\Xi)\,(\mathbf{T})^{-1}\,\mathbf{p}\\ &\approx (\mathbf{I}-\Delta\Xi)\,(\mathbf{T})^{-1}\,\mathbf{p}\\ &= (\mathbf{T})^{-1}\,\mathbf{p} - \Delta\Xi\,(\mathbf{T})^{-1}\,\mathbf{p}\\ &= (\mathbf{T})^{-1}\,\mathbf{p} + \mathbf{F}(-\,(\mathbf{T})^{-1}\,\mathbf{p})\Delta\xi,\end{aligned}$$

where $\exp\Delta\Xi$ is the small change of the matrix $\mathbf{T}$. In this case, the partial derivative of the measurement function with respect to the twist coordinates $\boldsymbol{\xi}$ is

$$\mathbf{J}_{\mathbf{e}}(\xi) = \frac{\partial\mathbf{e}(\mathbf{T},\mathbf{z})}{\partial\Delta\boldsymbol{\xi}} = \frac{\partial\mathbf{h}(\mathbf{T})}{\partial\Delta\boldsymbol{\xi}} = \mathbf{F}(-\,(\mathbf{T})^{-1}\,\mathbf{p}). \tag{A.98}$$

Derivative of the matrix differences The error function of the odometry measurements between t_1 and t_2 can be modeled as

$$\boldsymbol{\varphi}(\mathbf{T}_1,\mathbf{T}_2,\mathbf{Z}) = \mathbf{Z}^{-1}(\mathbf{T}_1^{-1}\mathbf{T}_2) \tag{A.99}$$

where $\mathbf{Z}$ is the measurement, and $\mathbf{T}_1$ and $\mathbf{T}_2$ are the pose estimates at t_1 and t_2, respectively. When a small difference $\exp\Xi_1$ in $\mathbf{T}_1$ induces a change in the error function $\exp\Xi_{\boldsymbol{\varphi}}$, Eq. (A.99) can be re-written as

$$\begin{aligned}&\boldsymbol{\varphi}(\mathbf{T}_1,\mathbf{T}_2,\mathbf{Z})\exp\Xi_{\boldsymbol{\varphi}} = \boldsymbol{\varphi}((\mathbf{T}_1\exp\Xi_1),\mathbf{T}_2,\mathbf{Z}),\\ &\Leftrightarrow \mathbf{Z}^{-1}(\mathbf{T}_1^{-1}\mathbf{T}_2)\exp\Xi_{\boldsymbol{\varphi}} = \mathbf{Z}^{-1}((\mathbf{T}_1\exp\Xi_1)^{-1}\,\mathbf{T}_2).\end{aligned} \tag{A.100}$$

By multiplying the inverse of $\mathbf{Z}^{-1}(\mathbf{T}_1^{-1}\mathbf{T}_2)$ on the both sides of Eq. (A.100),

$$\begin{aligned}\exp\Xi_{\boldsymbol{\varphi}} &= \left(\mathbf{T}_2^{-1}\mathbf{T}_1\right)^{-1}(\exp\Xi_1)^{-1}\left(\mathbf{T}_2^{-1}\mathbf{T}_1\right)\\ &= \left(\left(\mathbf{T}_2^{-1}\mathbf{T}_1\right)\exp\Xi_1\left(\mathbf{T}_2^{-1}\mathbf{T}_1\right)^{-1}\right)^{-1}.\end{aligned} \tag{A.101}$$

Using the relation Eq. (A.94),

$$\begin{aligned}\left(\left(\mathbf{T}_2^{-1}\mathbf{T}_1\right)\exp\Xi_1\left(\mathbf{T}_2^{-1}\mathbf{T}_1\right)^{-1}\right)^{-1} &= \exp\left(\left(\left(\mathbf{T}_2^{-1}\mathbf{T}_1\right)\exp\Xi_1\left(\mathbf{T}_2^{-1}\mathbf{T}_1\right)^{-1}\right)^{-1}\right)\\ &= -\exp\left(\left(\mathbf{T}_2^{-1}\mathbf{T}_1\right)\exp\Xi_1\left(\mathbf{T}_2^{-1}\mathbf{T}_1\right)^{-1}\right)\\ &= -\exp\left(\mathrm{Adj}\left(\mathbf{T}_2^{-1}\mathbf{T}_1\right)\Xi_1\right).\end{aligned} \tag{A.102}$$

By substituting Eq. (A.102) to Eq. (A.101), the following equation can be derived:

$$\Xi_{\boldsymbol{\varphi}} = -\mathrm{Adj}\left(\mathbf{T}_2^{-1}\mathbf{T}_1\right)\Xi_1.$$

Thus, the derivative of $\Xi_{\boldsymbol{\varphi}}$ with respect to the difference of the first matrix Ξ_1 is

$$\frac{\partial\Xi_{\boldsymbol{\varphi}}}{\partial\Xi_1} = -\mathrm{Adj}\left(\mathbf{T}_2^{-1}\mathbf{T}_1\right). \tag{A.103}$$

the derivative of Eq. (A.99) with respect to the second pose is $\mathbf{I}_n$ for the n-dimensional Lie group matrix $\mathbf{T}_2$.

Similarly, the error function can be written as

$$\begin{aligned}&\boldsymbol{\varphi}(\mathbf{T}_1, \mathbf{T}_2, \mathbf{Z})\exp\Xi_{\boldsymbol{\varphi}} = \boldsymbol{\varphi}(\mathbf{T}_1, \mathbf{T}_2\exp\Xi_2, \mathbf{Z})\\ &\Leftrightarrow \mathbf{Z}^{-1}(\mathbf{T}_1^{-1}\mathbf{T}_2)\exp\Xi_{\boldsymbol{\varphi}} = \mathbf{Z}^{-1}(\mathbf{T}_1^{-1}\mathbf{T}_2\exp\Xi_2),\end{aligned}$$

when the difference of the second matrix $\exp\Xi_2$ induces the error $\exp\Xi_{\boldsymbol{\varphi}}$, and $\exp\Xi_2$ and $\exp\Xi_{\boldsymbol{\varphi}}$ have the following relationship:

$$\Xi_{\boldsymbol{\varphi}} = \Xi_2.$$

Thus, when $\mathbf{T}_2$ is a n-dimensional matrix,

$$\frac{\partial\Xi_{\boldsymbol{\varphi}}}{\partial\Xi_2} = \mathbf{I}_n. \tag{A.104}$$

A.2. Rotations: SO(3)

The 3×3 rotation matrix $\mathbf{R}$ is the representative example in the Lie group SO(3). The rotation matrix is orthogonal and can be inverted by transposing as $(\mathbf{R})^{-1} = (\mathbf{R})^{\mathrm{T}}$. A 3×3 skew matrix

$[\boldsymbol{\omega}]_\times$ of the tangent vector $\boldsymbol{\omega} = [\omega_1, \omega_2, \omega_3]^T$ in the corresponding Lie algebra so(3) can be expanded as

$$[\boldsymbol{\omega}]_\times = \omega_1 \mathbf{G}_1 + \omega_2 \mathbf{G}_2 + \omega_3 \mathbf{G}_3 = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \in \mathrm{so}(3),$$

where the generators $\mathbf{G}_i$ are defined as

$$\mathbf{G}_1 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}, \mathbf{G}_2 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix}, \mathbf{G}_3 = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

The tangent vector $\boldsymbol{\omega}$ can be mapped to the rotation matrix in the group SO(3) by the exponential mapping:

$$\begin{aligned} \mathbf{R} &= \exp[\boldsymbol{\omega}]_\times \\ &= \mathbf{I}_3 + \left(1 - \frac{\theta^2}{3!} + \frac{\theta^4}{5!} + \cdots\right)[\boldsymbol{\omega}]_\times + \left(\frac{1}{2!} - \frac{\theta^2}{4!} + \frac{\theta^4}{6!} + \cdots\right)[\boldsymbol{\omega}]_\times^2 \\ &= \mathbf{I}_3 + \frac{\sin\theta}{\theta}[\boldsymbol{\omega}]_\times + \frac{1-\cos\theta}{\theta^2}[\boldsymbol{\omega}]_\times^2 \\ &= (\cos\theta)\mathbf{I}_3 + \frac{\sin\theta}{\theta}[\boldsymbol{\omega}]_\times + \frac{1-\cos\theta}{\theta^2}\boldsymbol{\omega}\,(\boldsymbol{\omega})^{\mathrm{T}}, \end{aligned} \tag{A.105}$$

where $\theta = \sqrt{(\boldsymbol{\omega})^{\mathrm{T}}\boldsymbol{\omega}}$. Eq. (A.105) is equivalent to the Rodrigues' formula [Murray, 2017] described using the axis-angle rotation representation with the rotation angle θ and the rotation axis $\boldsymbol{\omega}/\theta$.

The logarithmic transformation can be applied to convert matrices in the group SO(3) to so(3),

$$[\boldsymbol{\omega}]_\times = \frac{\theta}{2\sin\theta}(\mathbf{R} - (\mathbf{R})^{\mathrm{T}}) \tag{A.106}$$

$$\theta = \arccos\frac{tr(\mathbf{R}) - 1}{2}. \tag{A.107}$$

The adjoint map The adjoint of a rotation matrix $\mathbf{R}$ is equivalent to the axis rotation from $\boldsymbol{\omega}$ to $\mathbf{R}\boldsymbol{\omega}$:

$$\mathrm{Adj}\,(\mathbf{R})\,[\boldsymbol{\omega}]_\times = \mathbf{R}[\boldsymbol{\omega}]_\times(\mathbf{R})^{\mathrm{T}} = [\mathbf{R}\boldsymbol{\omega}]_\times,$$

Differentiation of the group actions A matrix $\mathbf{R}$ in the Lie group SO(3) can rotate a vector $\mathbf{x}$ in $\mathbb{R}^3$ or convert the reference frame of the vector as

$$\mathbf{y}(\mathbf{R}, \mathbf{x}) = \mathbf{R}\mathbf{x}.$$

This can be written as following when there is a small increment in $\boldsymbol{\omega}$;

$$\mathbf{y}(\boldsymbol{\omega}) = \mathbf{R} \exp[\boldsymbol{\omega}]_\times \mathbf{x}.$$

Assuming $\exp[\boldsymbol{\omega}]_\times \approx \mathbf{I}_3 + [\boldsymbol{\omega}]_\times$, the derivative of $\mathbf{y}(\boldsymbol{\omega})$ with respect to the rotation parameters $\boldsymbol{\omega}$ can be derived as

$$\begin{aligned}\frac{\partial \mathbf{y}(\boldsymbol{\omega})}{\partial \boldsymbol{\omega}} &= \mathbf{R}\frac{\partial \exp[\boldsymbol{\omega}]_\times \mathbf{x}}{\partial \boldsymbol{\omega}} \\ &\approx \mathbf{R}\frac{\partial [\boldsymbol{\omega}]_\times \mathbf{x}}{\partial \boldsymbol{\omega}} \\ &= \mathbf{R}[-\mathbf{x}]_\times \triangleq \mathbf{R}H[\mathbf{x}],\end{aligned} \tag{A.108}$$

where $H[\mathbf{x}] = [-\mathbf{x}]_\times$ is a 3×3 matrix. In this equation, $\frac{\partial [\boldsymbol{\omega}]_\times \mathbf{x}}{\partial \boldsymbol{\omega}} = [-\mathbf{x}]_\times$ because

$$[\boldsymbol{\omega}]_\times \mathbf{x} = \boldsymbol{\omega} \times \mathbf{x} = -\mathbf{x} \times \boldsymbol{\omega} = [-\mathbf{x}]_\times \boldsymbol{\omega}.$$

Similarly, the inverse group actions $\mathbf{y} = (\mathbf{R})^\top \mathbf{x}$ can be differentiated as

$$\frac{\partial \mathbf{y}(\boldsymbol{\omega})}{\partial \boldsymbol{\omega}} = H[-(\mathbf{R})^\top \mathbf{x}] = [\mathbf{y}]_\times. \tag{A.109}$$

A.3. Rigid Transformations: SE(3)

The Lie group SE(3) is a subgroup of the general linear group GL(4). It includes the 4×4 matrices that describe the 3D rigid motions:

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \in \mathrm{SE}(3),$$

where $\mathbf{R} \in \mathrm{SO}(3)$ is a rotation matrix, and $\mathbf{t} \in \mathbb{R}^3$ is a 3×1 translation vector.

The matrices in SE(3) can be inverted as

$$(\mathbf{T})^{-1} = \begin{bmatrix} (\mathbf{R})^{\mathrm{T}} & -(\mathbf{R})^{\mathrm{T}}\mathbf{t} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \in \mathrm{SE}(3), \tag{A.110}$$

and two matrices $\mathbf{T}_1$ and $\mathbf{T}_2$ in SE(3) can be multiplied as

$$\mathbf{T}_1\mathbf{T}_2 = \begin{bmatrix} \mathbf{R}_1\mathbf{R}_2 & \mathbf{R}_1\mathbf{t}_2 + \mathbf{t}_1 \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \in \mathrm{SE}(3). \tag{A.111}$$

The corresponding Lie algebra se(3) includes 4×4 matrices $\Xi \in se(3)$ which are vector spaces of the twist coordinates $\boldsymbol{\xi} = [\boldsymbol{\omega}, \boldsymbol{\nu}]^T \in \mathbb{R}^6$:

$$\Xi = \omega_1\mathbf{G}_1 + \omega_2\mathbf{G}_2 + \omega_3\mathbf{G}_3 + \nu_1\mathbf{G}_4 + \nu_2\mathbf{G}_5 + \nu_3\mathbf{G}_6 = \begin{bmatrix} [\boldsymbol{\omega}]_\times & \boldsymbol{\nu} \\ 0 & 0 \end{bmatrix} \in \mathrm{se}(3),$$

where the generators $\mathbf{G}_i$ are

$$\mathbf{G}_1 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \mathbf{G}_2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \mathbf{G}_3 = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{G}_4 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \mathbf{G}_5 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \mathbf{G}_6 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

A matrix in the Lie algebra se(3) can be mapped to the corresponding matrix in the Lie group SE(3) by the closed-form exponential mapping as

$$\begin{aligned} \mathbf{T} &= \exp \Xi \\ &= \mathbf{I}_4 + \Xi + \frac{1}{2!}\begin{bmatrix} [\boldsymbol{\omega}]_\times^2 & [\boldsymbol{\omega}]_\times \nu \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} + \frac{1}{3!}\begin{bmatrix} [\boldsymbol{\omega}]_\times^3 & [\boldsymbol{\omega}]_\times^2 \nu \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} + \cdots \\ &= \begin{bmatrix} \exp[\boldsymbol{\omega}]_\times & \mathbf{V}\boldsymbol{\nu} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{V}\boldsymbol{\nu} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix}, \end{aligned}$$

where

$$\begin{aligned}
\theta &= \sqrt{(\boldsymbol{\omega})^{\mathrm{T}}\boldsymbol{\omega}} \\
A &= \sin\theta/\theta \\
B &= (1-\cos\theta)/\theta^2 \\
C &= (1-A)/\theta^2 \\
\mathbf{R} &= \mathbf{I}_3 + A[\boldsymbol{\omega}]_\times + B[\boldsymbol{\omega}]_\times^2 \\
\mathbf{V} &= \mathbf{I}_3 + B[\boldsymbol{\omega}]_\times + C[\boldsymbol{\omega}]_\times^2.
\end{aligned}$$

A matrix in the Lie group SE(3) can be mapped back to the algebra se(3) using Eq. (A.106) and Eq. (A.107) for the rotation part, and the translation part can be mapped employing $\boldsymbol{\nu} = (\mathbf{V})^{-1}\mathbf{t}$, where the inverse of $\mathbf{V}$ is

$$(\mathbf{V})^{-1} = \mathbf{I}_3 - \frac{1}{2}[\boldsymbol{\omega}]_\times + \frac{1}{\theta^2}\left(1-\frac{A}{2B}\right)[\boldsymbol{\omega}]_\times^2.$$

The adjoint map The adjoint map of the matrix $\mathbf{T}$ in the Lie group SE(3) is

$$\mathrm{Adj}(\mathbf{T})\,\boldsymbol{\Xi} = \mathbf{T}\boldsymbol{\Xi}(\mathbf{T})^{-1} = \begin{bmatrix} [\mathbf{R}\boldsymbol{\omega}]_\times & \mathbf{t}\times\mathbf{R}\boldsymbol{\omega} + \mathbf{R}\boldsymbol{\nu} \\ \mathbf{0}_{1\times 3} & 0 \end{bmatrix}.$$

With the twist coordinates $\boldsymbol{\xi} = [\boldsymbol{\omega}, \boldsymbol{\nu}]^T$, the adjoint map $\mathrm{Adj}(\mathbf{T})$ can be computed as

$$\begin{pmatrix} \boldsymbol{\omega}' \\ \boldsymbol{\nu}' \end{pmatrix} = \mathrm{Adj}(\mathbf{T}) \begin{pmatrix} \boldsymbol{\omega} \\ \boldsymbol{\nu} \end{pmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{0}_{3\times 3} \\ [\mathbf{t}]_\times\mathbf{R} & \mathbf{R} \end{bmatrix} \begin{pmatrix} \boldsymbol{\omega} \\ \boldsymbol{\nu} \end{pmatrix},$$

as explained in [Murray, 2017].

Differentiation of the group actions A matrix $\mathbf{T}$ in SE(3) can transform a 4×1 homogeneous vector $\tilde{\mathbf{x}} = [\mathbf{x}, 1]^T$ as

$$\tilde{\mathbf{y}}(\mathbf{T}, \tilde{\mathbf{x}}) = \begin{pmatrix} \mathbf{y} \\ 1 \end{pmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \begin{pmatrix} \mathbf{x} \\ 1 \end{pmatrix} = \mathbf{T}\tilde{\mathbf{x}}.$$

A small increment of the pose $\boldsymbol{\xi}$ induces the change of the vector $\tilde{\mathbf{y}}$ as

$$\tilde{\mathbf{y}}(\boldsymbol{\xi}) = \mathbf{T} \exp \boldsymbol{\xi} \tilde{\mathbf{x}}.$$

Assuming $\exp \boldsymbol{\xi} \approx \mathbf{I}_3 + \Xi$, the derivative of $\tilde{\mathbf{y}}$ of the variable $\boldsymbol{\xi}$ can be computed as

$$\frac{\partial \tilde{\mathbf{y}}(\boldsymbol{\xi})}{\partial \boldsymbol{\xi}} = \mathbf{T} \frac{\partial \Xi \tilde{\mathbf{x}}}{\partial \boldsymbol{\xi}},$$

where

$$\Xi \tilde{\mathbf{x}} = \begin{bmatrix} \boldsymbol{\omega} \times \mathbf{x} + \boldsymbol{\nu} \\ 0 \end{bmatrix}.$$

Since the first row of $\Xi \tilde{\mathbf{x}}$ is

$$\boldsymbol{\omega} \times \mathbf{x} + \boldsymbol{\nu} = \begin{bmatrix} -[\mathbf{x}]_\times & \mathbf{I}_3 \end{bmatrix} \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{\nu} \end{bmatrix} = \begin{bmatrix} -[\mathbf{x}]_\times & \mathbf{I}_3 \end{bmatrix} \boldsymbol{\xi},$$

the differentiation of $\tilde{\mathbf{y}}$ of the twist coordinates $\boldsymbol{\xi}$ can be calculated as

$$\begin{aligned} \frac{\partial \tilde{\mathbf{y}}(\boldsymbol{\xi})}{\partial \boldsymbol{\xi}} &= \mathbf{T} \frac{\partial \Xi \tilde{\mathbf{x}}}{\partial \boldsymbol{\xi}} \\ &= \mathbf{T} \begin{bmatrix} -[\mathbf{x}]_\times & \mathbf{I}_3 \\ \mathbf{0}_{1\times 3} & 0 \end{bmatrix} \triangleq \mathbf{T} H[\mathbf{x}]. \end{aligned} \tag{A.112}$$

Similarly, the inverse transformation $\mathbf{y} = (\mathbf{T})^{-1} \mathbf{x}$ can be differentiated as

$$\frac{\partial \tilde{\mathbf{y}}(\boldsymbol{\xi})}{\partial \boldsymbol{\xi}} = H[(-\mathbf{T})^{-1} \mathbf{x}] = \begin{bmatrix} [\mathbf{y}]_\times & -\mathbf{I}_3 \\ \mathbf{0}_{1\times 3} & 0 \end{bmatrix}. \tag{A.113}$$

A.4. Similarity Transformations: Sim(3)

The Lie group Sim(3) consists of 4×4 similarity matrices $\mathbf{S}$ which are defined with the rotation matrices $\mathbf{R} \in \mathrm{SO}(3)$, translation vectors $\mathbf{t} \in \mathbb{R}^3$, and scale factors $\mathrm{s} \in \mathbb{R}^1$:

$$\mathbf{S} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}_{1\times 3} & (\mathrm{s})^{-1} \end{bmatrix} \in \mathrm{Sim(3)}.$$

The inversion of a matrix in the Sim(3) group is

$$(\mathbf{S})^{-1} = \begin{bmatrix} (\mathbf{R})^{\mathrm{T}} & -\mathrm{s}(\mathbf{R})^{\mathrm{T}}\mathbf{t} \\ \mathbf{0}_{1\times 3} & \mathrm{s} \end{bmatrix} \in \mathrm{Sim}(3),$$

and the matrix multiplication of $\mathbf{S}_1$ and $\mathbf{S}_2$ in Sim(3) is

$$\mathbf{S}_1\mathbf{S}_2 = \begin{bmatrix} \mathbf{R}_1\mathbf{R}_2 & \mathbf{R}_1\mathbf{t}_2 + (\mathrm{s}_2)^{-1}\mathbf{t}_1 \\ \mathbf{0}_{1\times 3} & (\mathrm{s}_1\mathrm{s}_2)^{-1} \end{bmatrix} \in \mathrm{Sim}(3).$$

The corresponding Lie algebra sim(3) includes 4×4 matrices generated with the parameters $\boldsymbol{\xi} = [\boldsymbol{\omega}, \boldsymbol{\nu}, \lambda]^T \in \mathbb{R}^7$ as

$$\Xi = \omega_1\mathbf{G}_1 + \omega_2\mathbf{G}_2 + \omega_3\mathbf{G}_3 + \nu_1\mathbf{G}_4 + \nu_2\mathbf{G}_5 + \nu_3\mathbf{G}_6 + \lambda\mathbf{G}_7 = \begin{bmatrix} [\boldsymbol{\omega}]_{\mathrm{X}} & \boldsymbol{\nu} \\ \mathbf{0}_{1\times 3} & -\lambda \end{bmatrix} \in \mathrm{sim}(3),$$

with the generators $\mathbf{G}_i$ are

$$\mathbf{G}_1 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \mathbf{G}_2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \mathbf{G}_3 = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{G}_4 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \mathbf{G}_5 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \mathbf{G}_6 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \mathbf{G}_7 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}.$$

A matrix Ξ in the algebra sim(3) with the tangent vector (twist coordinates) $\boldsymbol{\xi} = [\boldsymbol{\omega}, \boldsymbol{\nu}, \lambda]^T$ can be convert to the equivalent matrix $\mathbf{S}$ in the Sim(3) group using the close-form exponential mapping as

$$\begin{aligned}\mathbf{S} &= \exp \Xi \\ &= \mathbf{I}_4 + \Xi + \frac{1}{2!}\begin{bmatrix}[\boldsymbol{\omega}]_\times^2 & [\boldsymbol{\omega}]_\times \boldsymbol{\nu} - \lambda \boldsymbol{\nu} \\ \mathbf{0}_{1\times 3} & \lambda^2\end{bmatrix} + \frac{1}{3!}\begin{bmatrix}[\boldsymbol{\omega}]_\times^3 & [\boldsymbol{\omega}]_\times^2 \boldsymbol{\nu} - \lambda[\boldsymbol{\omega}]_\times \boldsymbol{\nu} + \lambda^2 \boldsymbol{\nu} \\ \mathbf{0}_{1\times 3} & -\lambda^3\end{bmatrix} + \cdots \\ &= \begin{bmatrix}\mathbf{R} & \mathbf{V}\boldsymbol{\nu} \\ \mathbf{0}_{1\times 3} & \exp(-\lambda)\end{bmatrix},\end{aligned}$$

where the matrices $\mathbf{R}$ and $\mathbf{V}$ are

$$\begin{aligned}\mathbf{R} &= \mathbf{I}_3 + a[\boldsymbol{\omega}]_\times + b[\boldsymbol{\omega}]_\times^2 \\ \mathbf{V} &= A\mathbf{I}_3 + B[\boldsymbol{\omega}]_\times + C[\boldsymbol{\omega}]_\times^2,\end{aligned}$$

with the coefficients $a, b, A, B,$ and C.

The adjoint map The adjoint map of the matrix $\mathbf{S}$ in the Lie group Sim(3) is

$$\mathrm{Adj}\,(\mathbf{S})\,\Xi = \mathbf{S}\Xi\,(\mathbf{S})^{-1} = \begin{bmatrix}[\mathbf{R}\boldsymbol{\omega}]_\times & \mathrm{s}(\mathbf{t} \times \mathbf{R}\boldsymbol{\omega} + \mathbf{R}\boldsymbol{\nu} - \lambda \mathbf{t}) \\ \mathbf{0}_{1\times 3} & -\lambda\end{bmatrix}.$$

The adjoint map $\mathrm{Adj}\,(\mathbf{S})$ can be also multiplied with twist coordinates $\boldsymbol{\xi} = [\boldsymbol{\omega}, \boldsymbol{\nu}, \lambda]^T$ as

$$\begin{pmatrix}\boldsymbol{\omega}' \\ \boldsymbol{\nu}' \\ \lambda'\end{pmatrix} = \mathrm{Adj}\,(\mathbf{S})\begin{pmatrix}\boldsymbol{\omega} \\ \boldsymbol{\nu} \\ \lambda\end{pmatrix} = \begin{bmatrix}\mathbf{R} & \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times 1} \\ \mathrm{s}[\mathbf{t}]_\times \mathbf{R} & \mathrm{s}\mathbf{R} & -\mathrm{s}\mathbf{t} \\ \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & 1\end{bmatrix}\begin{pmatrix}\boldsymbol{\omega} \\ \boldsymbol{\nu} \\ \lambda\end{pmatrix}, \tag{A.114}$$

as introduced in [Murray, 2017].

Differentiation of the group actions A matrix in the Lie group Sim(3) can conduct a similarity transformation of a homogeneous vector $\tilde{\mathbf{x}} = [\mathbf{x}, 1]^T$:

$$\tilde{\mathbf{y}} = \begin{pmatrix}\mathbf{y} \\ (\mathrm{s})^{-1}\end{pmatrix} = \begin{bmatrix}\mathbf{R} & \mathbf{t} \\ \mathbf{0}_{1\times 3} & (\mathrm{s})^{-1}\end{bmatrix}\begin{pmatrix}\mathbf{x} \\ 1\end{pmatrix} = \mathbf{S}\tilde{\mathbf{x}}.$$

The vector $\tilde{\mathbf{y}}(\boldsymbol{\xi})$ can be differentiated with respect to the twist coordinates $\boldsymbol{\xi}$ as

$$\frac{\partial \tilde{\mathbf{y}}(\boldsymbol{\xi})}{\partial \boldsymbol{\xi}} = \mathbf{S}\frac{\partial (\exp \Xi)\tilde{\mathbf{x}}}{\partial \boldsymbol{\xi}} \approx \mathbf{S}\frac{\partial \Xi \tilde{\mathbf{x}}}{\partial \boldsymbol{\xi}}.$$

$\Xi\tilde{\mathbf{x}}$ in this equation can be expanded as

$$\Xi\tilde{\mathbf{x}} = \begin{bmatrix} [\boldsymbol{\omega}]_\times & \boldsymbol{\nu} \\ \mathbf{0}_{1\times 3} & -\lambda \end{bmatrix} \begin{pmatrix} \mathbf{x} \\ 1 \end{pmatrix} = \begin{bmatrix} -[\mathbf{x}]_\times & \mathbf{I}_3 & 0 \\ \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & -1 \end{bmatrix} \begin{pmatrix} \boldsymbol{\omega} \\ \boldsymbol{\nu} \\ \lambda \end{pmatrix},$$

so the differentiation of this group action can be derived as

$$\frac{\partial \tilde{\mathbf{y}}(\boldsymbol{\xi})}{\partial \boldsymbol{\xi}} = \mathbf{S}\frac{\partial}{\partial \boldsymbol{\xi}} \begin{bmatrix} -[\mathbf{x}]_\times & \mathbf{I}_3 & 0 \\ \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & -1 \end{bmatrix} \boldsymbol{\xi} = \mathbf{S} \begin{bmatrix} -[\mathbf{x}]_\times & \mathbf{I}_3 & 0 \\ \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & -1 \end{bmatrix} \triangleq \mathbf{S}H[\mathbf{x}]. \tag{A.115}$$

In case of an inverse transformation, such as

$$\tilde{\mathbf{y}} = \begin{pmatrix} \mathbf{y} \\ \mathrm{s} \end{pmatrix} = \begin{bmatrix} (\mathbf{R})^\mathrm{T} & -\mathrm{s}(\mathbf{R})^\mathrm{T}\mathbf{t} \\ \mathbf{0}_{1\times 3} & \mathrm{s} \end{bmatrix} \begin{pmatrix} \mathbf{x} \\ 1 \end{pmatrix} = (\mathbf{S})^{-1}\tilde{\mathbf{x}},$$

the differentiation can be derived similarly as

$$\frac{\partial \tilde{\mathbf{y}}(\boldsymbol{\xi})}{\partial \boldsymbol{\xi}} = \frac{\partial}{\partial \boldsymbol{\xi}} \begin{bmatrix} [\mathbf{y}]_\times & -\mathrm{s}\mathbf{I}_3 & 0 \\ \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \mathrm{s} \end{bmatrix} \boldsymbol{\xi} = \begin{bmatrix} [\mathbf{y}]_\times & -\mathrm{s}\mathbf{I}_3 & 0 \\ \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \mathrm{s} \end{bmatrix} \triangleq H[\tilde{\mathbf{y}}]. \tag{A.116}$$

A.5. Iterative Estimation of the States Defined in the Lie Groups

VSLAM estimates the camera pose matrices $\boldsymbol{\Theta} = \{\mathbf{T}_\mathrm{i}\ \forall i \in \{1, 2, \ldots, N\}\}$ using the visual measurements $\mathbf{Z} = \{\mathbf{z}_i\ \forall i \in \{1, 2, \ldots, M\}\}$ observed by the camera. For VSLAM, the maximum a posteriori (MAP) approach can be applied to search the optimal state variables $\hat{\boldsymbol{\Theta}}$ that maximize a posteriori, $P(\boldsymbol{\Theta}|\mathbf{Z})$.

With the Bayes theorem, a posteriori can be described with a prior of the unknowns $P(\boldsymbol{\Theta})$ and the likelihood of the measurements $P(\mathbf{Z}|\boldsymbol{\Theta}) = l(\boldsymbol{\Theta}; \mathbf{Z})$ as

$$\hat{\boldsymbol{\Theta}} = \underset{\boldsymbol{\Theta}}{\operatorname{argmax}}\, P(\boldsymbol{\Theta}|\mathbf{Z}) = \underset{\boldsymbol{\Theta}}{\operatorname{argmax}}\, \frac{P(\mathbf{Z}|\boldsymbol{\Theta})}{P(\mathbf{Z})} P(\boldsymbol{\Theta}).$$

Assuming $P(\boldsymbol{\Theta})$ and $P(\mathbf{Z})$ are constant, a posteriori is proportional to the likelihood of the state variable $\boldsymbol{\Theta}$ given the measurements $\mathbf{Z}$:

$$P(\boldsymbol{\Theta}|\mathbf{Z}) \propto P(\mathbf{Z}|\boldsymbol{\Theta}) = l(\boldsymbol{\Theta};\mathbf{Z}). \tag{A.117}$$

If the likelihood function $l(\boldsymbol{\Theta};\mathbf{Z})$ is a Gaussian distribution with the mean value computed using the measurement function $\mathbf{h}(\boldsymbol{\Theta})$ and the covariance matrix $\boldsymbol{\Sigma}$, it can be described as

$$\begin{aligned} l(\boldsymbol{\Theta};\mathbf{Z}) &= P(\mathbf{Z}|\boldsymbol{\Theta}) \sim \mathcal{N}(\mathbf{h}(\boldsymbol{\Theta}),\boldsymbol{\Sigma}) \\ &= \frac{1}{\sqrt{(2\pi)^k \det\boldsymbol{\Sigma}}} \exp\left(-\frac{1}{2}\sum_i \|\mathbf{z}_i - \mathbf{h}(\mathbf{T}_{\mathrm{i}})\|^2_{\Sigma_i}\right), \\ &= \frac{1}{\sqrt{(2\pi)^k \det\boldsymbol{\Sigma}}} \exp\left(-\frac{1}{2}\sum_i \|\mathbf{e}(\mathbf{T}_{\mathrm{i}},\mathbf{z})_i\|^2_{\Sigma_i}\right), \end{aligned} \tag{A.118}$$

where $\mathbf{T}_{\mathrm{i}}$ is the camera poses that observe the visual measurement $\mathbf{z}_i$, and $\mathbf{e}(\mathbf{T}_{\mathrm{i}},\mathbf{z}_i)$ is the measurement error function. The MAP estimation is equivalent to the least-squares estimation when the likelihood function $l(\boldsymbol{\Theta};\mathbf{Z})$ is Gaussian distributed:

$$\hat{\boldsymbol{\Theta}} = \underset{\boldsymbol{\Theta}}{\operatorname{argmax}}\, P(\boldsymbol{\Theta}|\mathbf{Z}) = \underset{\boldsymbol{\Theta}}{\operatorname{argmin}} \sum_i \|\mathbf{e}(\mathbf{T}_{\mathrm{i}},\mathbf{z}_i)\|^2_{\Sigma_i}. \tag{A.119}$$

The nonliear error function $\mathbf{e}(\mathbf{T}_{\mathrm{i}},\mathbf{z}_i)$ in Eq. (A.119) can be linearized using the twist coordinates difference $\Delta\boldsymbol{\xi}$ as

$$\begin{aligned} \mathbf{e}(\mathbf{T}^* \oplus \Delta\mathbf{T},\mathbf{z}) &= \mathbf{e}(\mathbf{T}^* \exp \Delta\Xi,\mathbf{z}) \\ &\approx \mathbf{e}(\mathbf{T}^*,\mathbf{z}) + \left.\frac{\partial \mathbf{e}(\mathbf{T},\mathbf{z})}{\partial \Delta\boldsymbol{\xi}}\right|_{\mathbf{T}=\mathbf{T}^*} \Delta\boldsymbol{\xi}. \end{aligned} \tag{A.120}$$

By substituting Eq. (A.120) into Eq. (A.119), the twist coordinates $\Delta\hat{\boldsymbol{\theta}} = \{\Delta\boldsymbol{\xi}_i\ \forall i \in \{1,2,\ldots,N\}\}$ can be the state variables of the least-squares estimation (instead of the matrices $\Delta\boldsymbol{\Theta}$):

$$\Delta\hat{\boldsymbol{\theta}} = \underset{\Delta\boldsymbol{\theta}}{\operatorname{argmin}} \sum_i \left\| \mathbf{e}(\tilde{\mathbf{T}}_{\mathrm{i}},\mathbf{z}_{\mathrm{i}}) + \left.\frac{\partial \mathbf{e}(\mathbf{T}_{\mathrm{i}},\mathbf{z}_{\mathrm{i}})}{\partial \Delta\boldsymbol{\xi}_i}\right|_{\mathbf{T}=\tilde{\mathbf{T}}} \Delta\boldsymbol{\xi}_i \right\|^2_{\Sigma}. \tag{A.121}$$

The deterministic solution of this equation can be obtained using the normal equation,

$$\mathbf{H}\Delta\hat{\boldsymbol{\theta}} = -\mathbf{b}, \tag{A.122}$$

which is derived by differentiating the cost function of Eq. (A.121). In Eq. (A.122), $\mathbf{H}$ =

$(\mathbf{J_e})^{\mathrm{T}}\,\mathbf{\Sigma}^{-1}\mathbf{J_e}$ and $\mathbf{b} = (\mathbf{J_e})^{\mathrm{T}}\,\mathbf{\Sigma}^{-1}\mathbf{e}$. Alternatively, Eq. (A.121) can be iteratively solved using the Gaussian-Newton method or Levenberg-Marquardt method. The optimal increments of the twist coordinates $\Delta\hat{\boldsymbol{\xi}}$ estimated with Eq. (A.121) are then used to update the pose matrices $\mathbf{T}_{\mathrm{i}}$ employing the exponential mapping explained in Eq. (A.90):

$$\mathbf{T}^{*} \leftarrow \mathbf{T}^{*} \oplus \Delta\mathbf{T}^{*} = \mathbf{T}^{*} \exp \Delta\Xi^{*}. \tag{A.123}$$

$\Delta\hat{\Xi}$ is a vector space of $\Delta\hat{\boldsymbol{\xi}}$ as defined in Eq. (A.91).

B. Statistical Modeling of Ranging Error

This section explains the statistical ranging error model that was employed to generate range measurements for the system evaluation in Section 3.3 and Section 5.3. For the error modeling, range measurements were obtained in two different outdoor environments (a football field and gravel pit) using IEEE 802.15.4-2011 Ultra-Wide Band (UWB) standard sensors with DW1000 IC chips from Decawave (Qorvo) mounted on the rover system developed by DLR-KN (see Fig. 3.17). As shown in Fig. 3.18, the ranging errors (the differences between the ranges computed with the ground truth poses and the ranges measured with the UWB sensors) are corrupted with both noise and bias. The errors are modeled as the combination of the systematic errors b_{sys}, multipath errors b_m, and zero-mean Gaussian noise n as

$$\varepsilon = b_{sys} + b_m + n. \tag{B.124}$$

The systematic bias of the UWB sensors is induced by the signal power changes, as explained in the application note provided from Decawave (Qorvo) [Ltd, 2014]. Fig. B.1 shows that range measurements obtained with the UWB modules differ due to the changes of the signals received at the sensors (blue line). The systematic bias is modeled as a function of the angles between the LoS vector and two onboard antennas (θ_1 and θ_2 in Fig. B.2) and the distance between two ranging modules (d in Fig. B.2):

$$\begin{aligned} b_{sys}(\theta_1, \theta_0, d) &= g_\theta(\theta_1, \theta_2) g_d(d) \\ &= \left(c_{1,\theta} \sin(\theta_1 + \theta_0) \sin(\theta_2 + \theta_0) + c_{0,\theta}\right) \left(c_{1,d} d + c_{0,d}\right), \end{aligned} \tag{B.125}$$

assuming that the signal power are mainly dependent on those three parameters. The angles between the robot and anchor are defined as θ'_1 and θ'_2 in Fig. B.2, and the distance between them is defined as d' in the same figure.

The coefficients of Eq. (B.125), $\{c_{1,\theta}, c_{0,\theta}, \theta_0, c_{1,d}, c_{0,d}\}$, are estimated by the data fitting process using the real range measurements obtained with the UWB modules. The coefficients

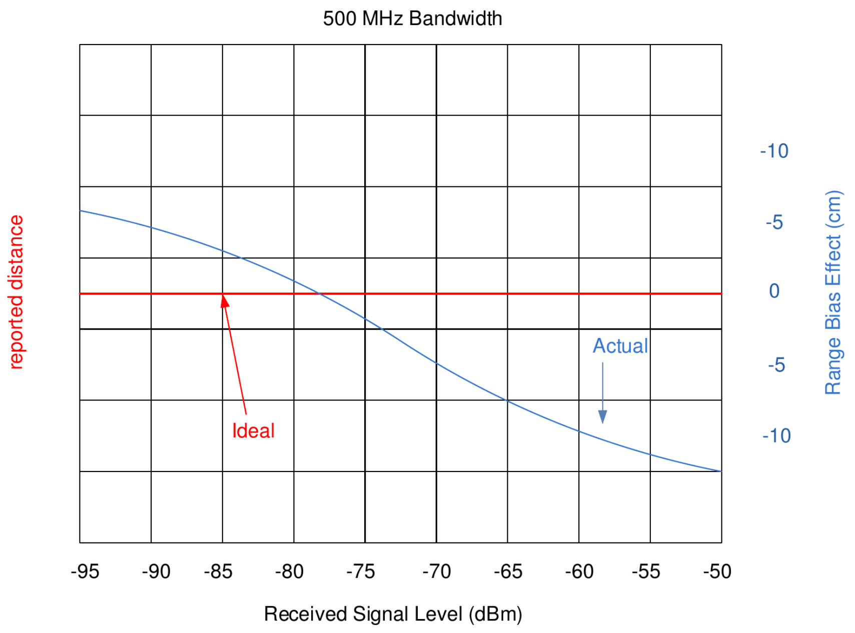


Figure B.1.: Ranging bias induced by the signal level received at the UWB modules (figure credit: [Ltd, 2014])

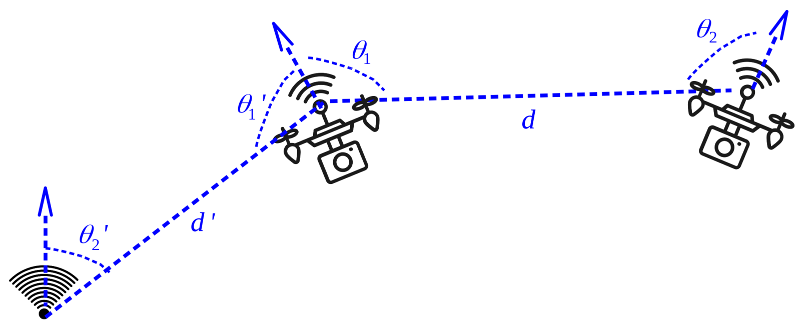


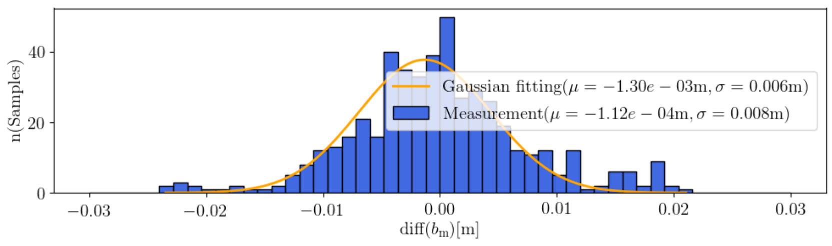
Figure B.2.: The definition of the angles between the LoS vector and two onboard antennas θ_1, θ_2, and the distance between them d

Table B.1.: The coefficients of the systematic bias model

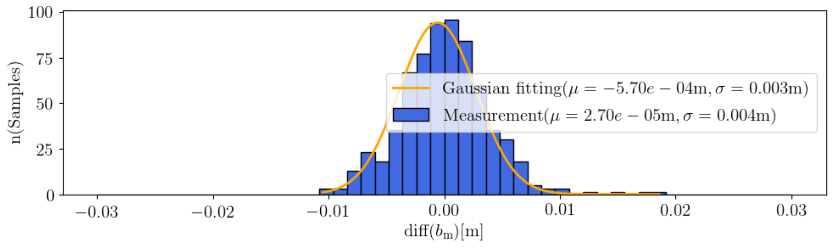
$c_{1,\theta}$	$c_{0,\theta}$	θ_0	$c_{1,d}$	$c_{0,d}$
-0.049	0.185	-0.395	-0.0589	1.921

$\{c_{1,\theta}, c_{0,\theta}, \theta_0\}$ of $g_\theta(\theta_1, \theta_2)$ are estimated first using the ranging errors ε, assuming that θ_1 and θ_2 are the most dominant factors in the systematic bias. Then, the coefficients $\{c_{1,d}, c_{0,d}\}$ of $g_d(d)$ are estimated using $\varepsilon / g_\theta(\theta_1, \theta_2)$. Table B.1 shows the coefficients estimated using this process.

The multipath bias is modeled using the ranging errors left after removing the systematic bias, i.e. $\varepsilon' \triangleq \varepsilon - b_{\text{sys}}$. Fig. B.3 shows the histograms of the differences between the adjacent values of the ranging errors left after eliminating the systematic bias, i.e. $\varepsilon'_i - \varepsilon'_{i-1}$. The orange lines are the Gaussian curves fitted using the differences. As can be seen in this figure, the

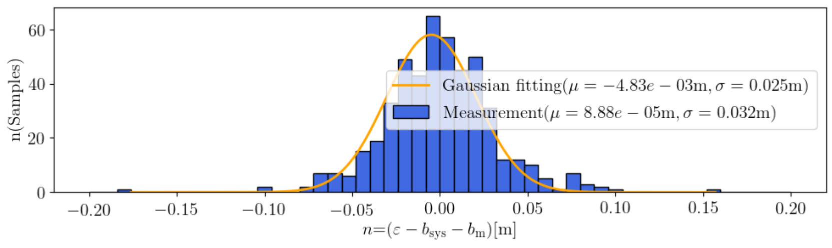


(a) The histogram of the differences of the multipath bias obtained in the football field. The Gaussian function fitted using the difference values is orange (mean=−0.0013m, std=0.006m)

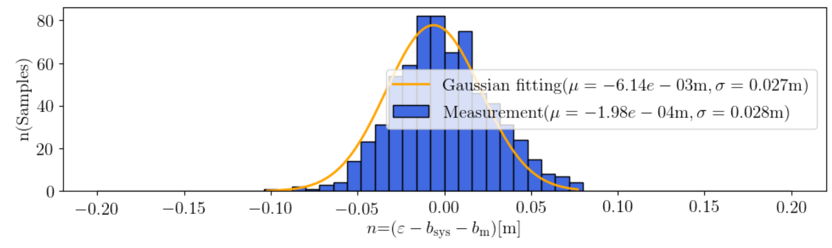


(b) The histogram of the differences of the multipath bias obtained in the gravel pit. The Gaussian function fitted using the difference values is orange (mean=−0.00057m, std=0.003m)

Figure B.3.: The histogram of the differences between the adjacent values of the multipath bias, i.e. $\text{diff}(b_m) = \varepsilon'_i - \varepsilon'_{i-1}$. The orange lines are the Gaussian functions fitted using the difference values



(a) The histogram of the ranging noise obtained in the football field. The Gaussian function fitted using the noise values is orange (mean=−0.0048m, std=0.025m)



(b) The histogram of the ranging noise obtained in the gravel field. The Gaussian function fitted using the noise values is orange (mean=−0.0061m, std=0.027m)

Figure B.4.: The histogram of the ranging noise, i.e. $n = \varepsilon - b_{sys} - b_m$. The orange lines are the Gaussian functions fitted using the noise values

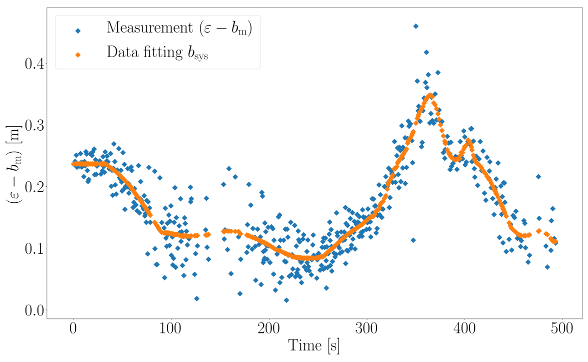
differences are well-fitted to the Gaussian curves, so the multipath bias can be modeled as the Gaussian random walk whose differences are Gaussian distributed.

Lastly, ranging noise is modeled as a zero-mean Gaussian distribution. Fig. B.4 shows the histogram of the ranging errors after eliminating both systematic and multipath bias, i.e. noise $n = \varepsilon - b_{\mathrm{sys}} - b_{\mathrm{m}}$. The orange curves are the Gaussian functions fitted using the noise values. Ranging noise is well-fitted to the fitting functions as shown in this figure, so it can be modeled as a zero-mean Gaussian distribution.

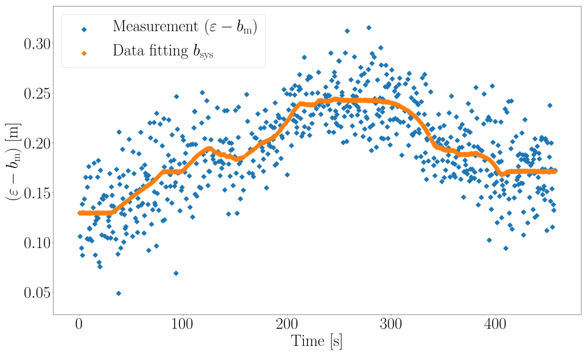
Verification of the systematic bias model In Fig. B.5, the blue dots are the ranging errors remaining after eliminating the multipath bias ($\varepsilon - b_{\mathrm{m}}$). The values computed with the systematic bias model $b_{\mathrm{sys}} = g_\theta(\theta_1, \theta_2) g_d(d)$ in Eq. (B.125) are plotted in this figure as the orange lines. If the systematic bias is properly modeled, the blue dots should follow the orange fitting curves because $\varepsilon - b_{\mathrm{m}} \simeq b_{\mathrm{sys}} + n$. Fig. B.5 shows the expected results for both datasets, so the systematic bias model derived in Eq. (B.125) is acceptable.

Fig. B.6 shows $g_\theta(\theta_1, \theta_2)$ over the angles θ_1 and θ_2 in orange as well as the values $\varepsilon - b_{\mathrm{m}}$ in blue. Since θ_1 and θ_2 are assumed to be the most dominant parameters of the systematic bias, the blue dots should follow the curve $g_\theta(\theta_1, \theta_2)$ ($\because \varepsilon - b_{\mathrm{m}} \simeq b_{\mathrm{sys}} + n$). This figure shows the expected trend, so the function $g_\theta(\theta_1, \theta_2)$ derived in Eq. (B.125) can be considered an acceptable model.

In addition, Fig. B.7 illustrates $(\varepsilon - b_m)/g_\theta(\theta_1, \theta_2)$ versus the true ranges computed with the ground truth poses in blue, while the curves obtained with the $g_d(d)$ model derived in Eq. (B.125) are orange. If $g_d(d)$ is properly modeled, the blue dots should follow the orange fitting curves because $(\varepsilon - b_m)/g_\theta(\theta_1, \theta_2) \simeq g_d(d) + n$. As this figure shows the expected results, the $g_d(d)$ model derived in Eq. (B.125) can be verified as a proper fit.

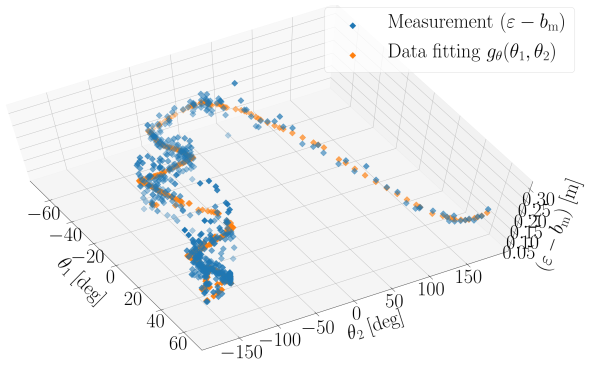


(a) $\varepsilon - b_m$ obtained in the football field (blue) and the systematic bias model b_{sys} (orange) over time

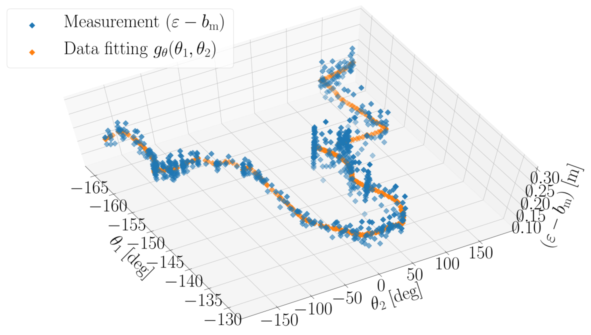


(b) $\varepsilon - b_m$ obtained in the gravel pit (blue) and the systematic bias model b_{sys} (orange) over time

Figure B.5.: The ranging errors after eliminating the multipath bias $\varepsilon - b_m \simeq b_{sys} + n$ (blue) and the values computed with the systematic bias model $b_{sys} = g_\theta(\theta_1, \theta_2) g_d(d)$ (orange) over time

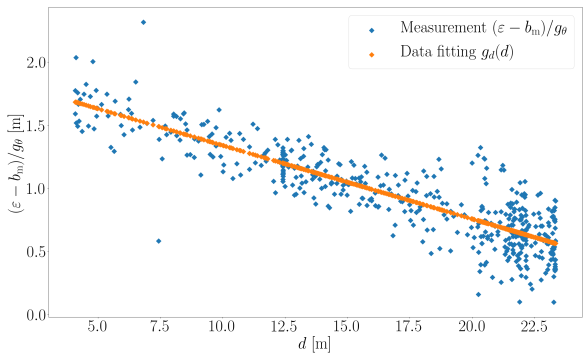


(a) $\varepsilon - b_m$ obtained in the football field (blue) and $g_\theta(\theta_1, \theta_2)$ (orange) versus θ_1 and θ_2

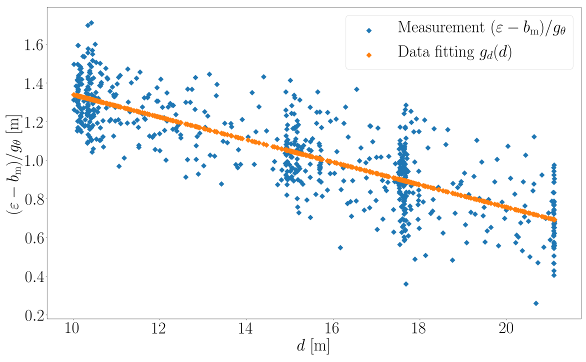


(b) $\varepsilon - b_m$ obtained in the gravel pit (blue) and $g_\theta(\theta_1, \theta_2)$ (orange) versus θ_1 and θ_2

Figure B.6.: The ranging errors after eliminating the multipath $\varepsilon - b_m$ (blue) and the values computed with the model $g_\theta(\theta_1, \theta_2)$ (orange) versus the angles θ_1 and θ_2



(a) $(\varepsilon - b_m)/g_\theta(\theta_1, \theta_2)$ obtained in the football field (blue) and $g_d(d)$ (orange) versus the true ranges



(b) $(\varepsilon - b_m)/g_\theta(\theta_1, \theta_2)$ obtained in the gravel pit (blue) and $g_d(d)$ (orange) versus the true ranges

Figure B.7.: $(\varepsilon - b_m)/g_\theta(\theta_1, \theta_2)$ (blue) and the values computed with the model $g_d(d)$ (orange) versus the true ranges

List of Figures

List of Tables

Bibliography

[Abdel-Aziz et al., 2015] Abdel-Aziz, Y., Karara, H., and Hauck, M. (2015). Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry. *Photogrammetric Engineering & Remote Sensing*, 81(2):103–107.

[Alam and Dempster, 2013] Alam, N. and Dempster, A. G. (2013). Cooperative positioning for vehicular networks: Facts and future. *IEEE transactions on intelligent transportation systems*, 14(4):1708–1717.

[Arandjelovic et al., 2016] Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T., and Sivic, J. (2016). Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5297–5307.

[Bay et al., 2008] Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L. (2008). Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359.

[Benini et al., 2013] Benini, A., Mancini, A., and Longhi, S. (2013). An imu/uwb/vision-based extended kalman filter for mini-uav localization in indoor environment using 802.15. 4a wireless sensor network. *Journal of Intelligent & Robotic Systems*, 70(1-4):461–476.

[Burri et al., 2016] Burri, M., Nikolic, J., Gohl, P., Schneider, T., Rehder, J., Omari, S., Achtelik, M. W., and Siegwart, R. (2016). The euroc micro aerial vehicle datasets. *The International Journal of Robotics Research*, 35(10):1157–1163.

[Cadena et al., 2016] Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I., and Leonard, J. J. (2016). Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on robotics*, 32(6):1309–1332.

[Chung et al., 2018] Chung, S.-J., Paranjape, A. A., Dames, P., Shen, S., and Kumar, V. (2018). A survey on aerial swarm robotics. *IEEE Transactions on Robotics*, 34(4):837–855.

[Cieslewski et al., 2018] Cieslewski, T., Choudhary, S., and Scaramuzza, D. (2018). Data-efficient decentralized visual slam. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2466–2473. IEEE.

[Cieslewski et al., 2015] Cieslewski, T., Lynen, S., Dymczyk, M., Magnenat, S., and Siegwart, R. (2015). Map api-scalable decentralized map building for robots. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6241–6247. IEEE.

[Cunningham et al., 2013] Cunningham, A., Indelman, V., and Dellaert, F. (2013). Ddf-sam 2.0: Consistent distributed smoothing and mapping. In *2013 IEEE international conference on robotics and automation*, pages 5220–5227. IEEE.

[Cunningham et al., 2010] Cunningham, A., Paluri, M., and Dellaert, F. (2010). Ddf-sam: Fully distributed slam using constrained factor graphs. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3025–3030. IEEE.

[Davison et al., 2007] Davison, A. J., Reid, I. D., Molton, N. D., and Stasse, O. (2007). Monoslam: Real-time single camera slam. *IEEE transactions on pattern analysis and machine intelligence*, 29(6):1052–1067.

[Dellaert et al., 2017] Dellaert, F., Kaess, M., et al. (2017). Factor graphs for robot perception. *Foundations and Trends® in Robotics*, 6(1-2):1–139.

[Eade, 2013] Eade, E. (2013). Lie groups for 2d and 3d transformations. *URL http://ethaneade.com/lie. pdf, revised Dec.*

[Engel et al., 2017] Engel, J., Koltun, V., and Cremers, D. (2017). Direct sparse odometry. *IEEE transactions on pattern analysis and machine intelligence*, 40(3):611–625.

[Engel et al., 2014] Engel, J., Schöps, T., and Cremers, D. (2014). Lsd-slam: Large-scale direct monocular slam. In *European conference on computer vision*, pages 834–849. Springer.

[Forster et al., 2014] Forster, C., Pizzoli, M., and Scaramuzza, D. (2014). Svo: Fast semi-direct monocular visual odometry. In *2014 IEEE international conference on robotics and automation (ICRA)*, pages 15–22. IEEE.

[Gadd and Newman, 2016] Gadd, M. and Newman, P. (2016). Checkout my map: Version control for fleetwide visual localisation. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5729–5736. IEEE.

[Gálvez-López and Tardos, 2012] Gálvez-López, D. and Tardos, J. D. (2012). Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics*, 28(5):1188–1197.

[Geiger et al., 2012] Geiger, A., Lenz, P., and Urtasun, R. (2012). Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE.

[Harris et al., 1988] Harris, C. G., Stephens, M., et al. (1988). A combined corner and edge detector. In *Alvey vision conference*, volume 15, pages 10–5244. Citeseer.

[Hartley and Zisserman, 2003] Hartley, R. and Zisserman, A. (2003). *Multiple view geometry in computer vision*. Cambridge university press.

[Hartley, 1997] Hartley, R. I. (1997). In defense of the eight-point algorithm. *IEEE Transactions on pattern analysis and machine intelligence*, 19(6):580–593.

[Horn, 1987] Horn, B. K. (1987). Closed-form solution of absolute orientation using unit quaternions. *Josa a*, 4(4):629–642.

[Kaess et al., 2012] Kaess, M., Johannsson, H., Roberts, R., Ila, V., Leonard, J. J., and Dellaert, F. (2012). isam2: Incremental smoothing and mapping using the bayes tree. *The International Journal of Robotics Research*, 31(2):216–235.

[Karrer et al., 2018] Karrer, M., Agarwal, M., Kamel, M., Siegwart, R., and Chli, M. (2018). Collaborative 6dof relative pose estimation for two uavs with overlapping fields of view. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6688–6693. IEEE.

[Kim et al., 2010] Kim, B., Kaess, M., Fletcher, L., Leonard, J., Bachrach, A., Roy, N., and Teller, S. (2010). Multiple relative pose graphs for robust cooperative mapping. In *2010 IEEE International Conference on Robotics and Automation*, pages 3185–3192. IEEE.

[Klein and Murray, 2007] Klein, G. and Murray, D. (2007). Parallel tracking and mapping for small ar workspaces. In *2007 6th IEEE and ACM international symposium on mixed and augmented reality*, pages 225–234. IEEE.

[Lee et al., 2018a] Lee, Y.-H., Zhu, C., Giorgi, G., and Guenther, C. (2018a). Stereo vision-based simultaneous localization and mapping with ranging aid. In *2018 IEEE/ION Position, Location and Navigation Symposium (PLANS)*, pages 404–409. IEEE.

[Lee et al., 2018b] Lee, Y.-H., Zhu, C., Giorgi, G., and Günther, C. (2018b). Fusion of monocular vision and radio-based ranging for global scale estimation and drift mitigation. *arXiv preprint arXiv:1810.01346*.

[Lee et al., 2020a] Lee, Y.-H., Zhu, C., Giorgi, G., and Günther, C. (2020a). Cooperative swarm localization and mapping with inter-agent ranging. In *2020 IEEE/ION Position, Location and Navigation Symposium (PLANS)*, pages 353–359. IEEE.

[Lee et al., 2020b] Lee, Y.-H., Zhu, C., Giorgi, G., and Günther, C. (2020b). Mitigation of odometry drift with a single ranging link in gnss-limited environments. pages 1117–1126.

[Leutenegger et al., 2011] Leutenegger, S., Chli, M., and Siegwart, R. Y. (2011). Brisk: Binary robust invariant scalable keypoints. In *2011 International conference on computer vision*, pages 2548–2555. Ieee.

[Levenberg, 1944] Levenberg, K. (1944). A method for the solution of certain non-linear problems in least squares. *Quarterly of applied mathematics*, 2(2):164–168.

[Lowe, 2004] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110.

[Ltd, 2014] Ltd, D. (2014). Aps011 (application note) - sources of error in dw1000 based two-way ranging (twr) schemes, version 1.0.

[Marquardt, 1963] Marquardt, D. W. (1963). An algorithm for least-squares estimation of nonlinear parameters. *Journal of the society for Industrial and Applied Mathematics*, 11(2):431–441.

[Mohanarajah et al., 2015] Mohanarajah, G., Usenko, V., Singh, M., D'Andrea, R., and Waibel, M. (2015). Cloud-based collaborative 3d mapping in real-time with low-cost robots. *IEEE Transactions on Automation Science and Engineering*, 12(2):423–431.

[Mouragnon et al., 2006] Mouragnon, E., Lhuillier, M., Dhome, M., Dekeyser, F., and Sayd, P. (2006). Real time localization and 3d reconstruction. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 363–370. IEEE.

[Mur-Artal et al., 2015] Mur-Artal, R., Montiel, J. M. M., and Tardos, J. D. (2015). Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163.

[Mur-Artal and Tardós, 2014] Mur-Artal, R. and Tardós, J. D. (2014). Fast relocalisation and loop closing in keyframe-based slam. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 846–853. IEEE.

[Mur-Artal and Tardós, 2017a] Mur-Artal, R. and Tardós, J. D. (2017a). Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262.

[Mur-Artal and Tardós, 2017b] Mur-Artal, R. and Tardós, J. D. (2017b). Visual-inertial monocular slam with map reuse. *IEEE Robotics and Automation Letters*, 2(2):796–803.

[Murray, 2017] Murray, R. M. (2017). *A mathematical introduction to robotic manipulation*. CRC press.

[Newcombe et al., 2011] Newcombe, R. A., Lovegrove, S. J., and Davison, A. J. (2011). Dtam: Dense tracking and mapping in real-time. In *2011 international conference on computer vision*, pages 2320–2327. IEEE.

[Nikookar and Oonincx, 2016] Nikookar, H. and Oonincx, P. (2016). An introduction to radio locationing with signals of opportunity. *Journal of Communication, Navigation, Sensing and Services (CONASENSE)*, 2016(1):1–10.

[Nistér et al., 2004] Nistér, D., Naroditsky, O., and Bergen, J. (2004). Visual odometry. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 1, pages I–I. Ieee.

[Nister and Stewenius, 2006] Nister, D. and Stewenius, H. (2006). Scalable recognition with a vocabulary tree. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2161–2168. Ieee.

[Olson and Abi-Rached, 2010] Olson, C. F. and Abi-Rached, H. (2010). Wide-baseline stereo vision for terrain mapping. *Machine Vision and Applications*, 21(5):713–725.

[Paull et al., 2015] Paull, L., Huang, G., Seto, M., and Leonard, J. J. (2015). Communication-constrained multi-auv cooperative slam. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 509–516. IEEE.

[Qin et al., 2018] Qin, T., Li, P., and Shen, S. (2018). Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020.

[Riazuelo et al., 2014] Riazuelo, L., Civera, J., and Montiel, J. M. (2014). C2tam: A cloud framework for cooperative tracking and mapping. *Robotics and Autonomous Systems*, 62(4):401–413.

[Richardson et al., 2013] Richardson, T. S., Jones, C. G., Likhoded, A., Sparks, E., Jordan, A., Cowling, I., and Willcox, S. (2013). Automated vision-based recovery of a rotary wing unmanned aerial vehicle onto a moving platform. *Journal of Field Robotics*, 30(5):667–684.

[Rizk et al., 2019] Rizk, Y., Awad, M., and Tunstel, E. W. (2019). Cooperative heterogeneous multi-robot systems: a survey. *ACM Computing Surveys (CSUR)*, 52(2):1–31.

[Rosten and Drummond, 2006] Rosten, E. and Drummond, T. (2006). Machine learning for high-speed corner detection. In *European conference on computer vision*, pages 430–443. Springer.

[Rublee et al., 2011] Rublee, E., Rabaud, V., Konolige, K., and Bradski, G. (2011). Orb: An efficient alternative to sift or surf. In *2011 International conference on computer vision*, pages 2564–2571. Ieee.

[Saeedi et al., 2016] Saeedi, S., Trentini, M., Seto, M., and Li, H. (2016). Multiple-robot simultaneous localization and mapping: A review. *Journal of Field Robotics*, 33(1):3–46.

[Sand et al., 2013] Sand, S., Zhang, S., Mühlegg, M., Falconi, G., Zhu, C., Krüger, T., and Nowak, S. (2013). Swarm exploration and navigation on mars. In *2013 International Conference on Localization and GNSS (ICL-GNSS)*, pages 1–6. IEEE.

[Schmuck and Chli, 2017] Schmuck, P. and Chli, M. (2017). Multi-uav collaborative monocular slam. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3863–3870. IEEE.

[Schmuck and Chli, 2019] Schmuck, P. and Chli, M. (2019). Ccm-slam: Robust and efficient centralized collaborative monocular simultaneous localization and mapping for robotic teams. *Journal of Field Robotics*, 36(4):763–781.

[Shi et al., 1994] Shi, J. et al. (1994). Good features to track. In *1994 Proceedings of IEEE conference on computer vision and pattern recognition*, pages 593–600. IEEE.

[Shi et al., 2018] Shi, Q., Cui, X., Li, W., Xia, Y., and Lu, M. (2018). Visual-uwb navigation system for unknown environments. pages 3111–3121.

[Siegwart et al., 2011] Siegwart, R., Nourbakhsh, I. R., and Scaramuzza, D. (2011). *Introduction to autonomous mobile robots*. MIT press.

[Strader et al., 2016] Strader, J., Gu, Y., Gross, J. N., De Petrillo, M., and Hardy, J. (2016). Cooperative relative localization for moving uavs with single link range measurements. In *2016 IEEE/ION Position, Location and Navigation Symposium (PLANS)*, pages 336–343. IEEE.

[Strasdat et al., 2012] Strasdat, H., Montiel, J. M., and Davison, A. J. (2012). Visual slam: why filter? *Image and Vision Computing*, 30(2):65–77.

[Tardioli et al., 2015] Tardioli, D., Montijano, E., and Mosteo, A. R. (2015). Visual data association in narrow-bandwidth networks. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2572–2577. IEEE.

[Thrun, 2002] Thrun, S. (2002). Probabilistic robotics. *Communications of the ACM*, 45(3):52–57.

[Triggs et al., 1999] Triggs, B., McLauchlan, P. F., Hartley, R. I., and Fitzgibbon, A. W. (1999). Bundle adjustment—a modern synthesis. In *International workshop on vision algorithms*, pages 298–372. Springer.

[Wang et al., 2017] Wang, C., Zhang, H., Nguyen, T.-M., and Xie, L. (2017). Ultra-wideband aided fast localization and mapping system. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1602–1609. IEEE.

[Xu et al., 2020] Xu, H., Wang, L., Zhang, Y., Qiu, K., and Shen, S. (2020). Decentralized visual-inertial-uwb fusion for relative state estimation of aerial swarm. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8776–8782. IEEE.

[Zhang et al., 2020] Zhang, S., Pöhlmann, R., Wiedemann, T., Dammann, A., Wymeersch, H., and Hoeher, P. A. (2020). Self-aware swarm navigation in autonomous exploration missions. *Proceedings of the IEEE*, 108(7):1168–1195.

[Zhou and Roumeliotis, 2006] Zhou, X. S. and Roumeliotis, S. I. (2006). Multi-robot slam with unknown initial correspondence: The robot rendezvous case. In *2006 IEEE/RSJ international conference on intelligent robots and systems*, pages 1785–1792. IEEE.

[Zhu, 2019] Zhu, C. (2019). *Cooperative Vision for Swarm Navigation*. PhD thesis, Technische Universität München.

[Ziegler et al., 2021] Ziegler, T., Karrer, M., Schmuck, P., and Chli, M. (2021). Distributed formation estimation via pairwise distance measurements. *IEEE Robotics and Automation Letters*, 6(2):3017–3024.

[Zou et al., 2019] Zou, D., Tan, P., and Yu, W. (2019). Collaborative visual slam for multiple agents: A brief survey. *Virtual Reality & Intelligent Hardware*, 1(5):461–482.