Skolkovo Institute of Science and Technology

Jacobian Approximation for State Estimation in Robotics via Optimization on Manifolds

Doctoral Thesis
by

Kazii Botashev

Doctoral Program in Computational and Data Science and Engineering

Supervisor
Associate Professor, Gonzalo Ferrer

Moscow — 2024

I hereby declare that the work presented in this thesis was carried out by myself at Skolkovo Institute of Science and Technology, Moscow, except where due acknowledgement is made, and has not been submitted for any other degree.

Candidate (Kazii Botashev)

Supervisor (Prof. Gonzalo Ferrer)

Abstract

Robotics has developed into a multidisciplinary field that encompasses the design, construction, and application of robots across various sectors, from autonomous delivery systems to household automation. This dissertation focuses on state estimation, a fundamental aspect of robotics, particularly in the contexts of time-continuous trajectory representation and visual localization, both of which are important for enabling effective robot operation in dynamic environments.

State estimation involves tasks such as localization and mapping, where robots need to determine their position and orientation over time. Traditional methods commonly employ discrete-time representations of robotic motion, which can become inefficient when handling high-rate or asynchronous sensor data. This research focuses on efficient computation of Jacobians for 3D trajectory estimation that facilitates the integration of information from high-frequency sensors, reducing the computational demands associated with discrete state variables.

Additionally, visual localization is essential for allowing robots to navigate and understand their surroundings. This dissertation explores innovative map representations, such as 3D Gaussian Splatting (3DGS), which enhance the accuracy of pose estimation through advanced rendering techniques. By deriving analytical forms of the Jacobian related to these rendering methods, the research supports effective optimization processes needed for accurate camera pose estimation.

The contributions of this thesis are twofold. First, it investigates time-continuous state estimation through optimization on manifold, offering empirical evaluations of Jacobian consistency in 3D trajectory estimation. Second, it addresses rendering-based visual localization by proposing methodologies for optimizing camera poses within the context of non-convex localization problems.

By examining these areas within the framework of optimization on manifold, this dissertation aims to contribute to the understanding of state estimation in robotics and provide practical solutions to relevant challenges in the field. This work offers insights and methodologies that advance both theoretical concepts and practical applications in robotic state estimation.

Publications

During my PhD studies, I was involved in the following publications. The main body of the dissertation is based on the publications listed only as "main author".

Main author

This section presents the main publications of the author related to the PhD dissertation.

1. **Kazii Botashev**, Vladislav Pyatov, Gonzalo Ferrer, Stamatios Lefkimmiatis. GSLoc: Visual Localization with 3D Gaussian Splatting. In 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2024 [CORE A] - accepted

2. **Kazii Botashev**, and Gonzalo Ferrer. Analytical jacobian approximation for direct optimization of a trajectory of interpolated poses on SE(3). In 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 9198–9204. IEEE, 2023. [CORE A]

3. **Kazii Botashev**, and Gonzalo Ferrer. Photorealistic Rendering with Camera-Refined 3D Gaussian Splatting. In Journal Science Prospects [ISSN 2077-6810, HAC (VAK)] - accepted

Co-author

This section contains publications prepared during graduate studies (not directly related to the PhD dissertation).

3. Sergey Osipenko, **Kazii Botashev**, Eugene Nikolaev, Yury Kostyukevich Transfer learning for small molecule retention predictions. Journal of chromatography. A, 1644, 462119. [Q1-Q2]

If everything seems under control, you are not going fast enough

Acknowledgments

I would like to express my deepest gratitude to everyone who has contributed to the completion of this dissertation.

First and foremost, I would like to extend my sincerest thanks to my supervisor, Professor Gonzalo Ferrer, whose unwavering support, patience, and invaluable guidance have been instrumental throughout my research journey. His expertise and insights have profoundly shaped my work.

I extend my heartfelt thanks to Dr. Stamatis Lefkimmiatis for his mentoring and assistance, which have significantly contributed to these results. His expertise and encouragement have made a remarkable difference in my research.

None of this would have been possible without the unwavering support and love of my dear wife, Dina, who has always believed in me more than anyone else, including myself. In a remarkable turn of events, she gave birth to our daughter just yesterday. While we have yet to decide on her name (instead, I am focusing on finishing this text now), I know that she is a pure angel and a beautiful bundle of love that my wife gifted me.

I am immensely grateful to my family for their unwavering support and encouragement. My father Ruslan, mother Anna and sister Mariam have been my constant inspiration, instilling in me the values of hard work and perseverance, and providing me with the foundation to pursue my dreams.

I would like to give special recognition to my sweet dog, Leya, who has always been by my side, offering warmth and comfort during those sleepless nights spent working on this dissertation.

Finally, I dedicate this work to all those who have believed in me and motivated me to pursue my academic goals. Thank you for being an integral part of my journey.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Robotics is a multidisciplinary field that encompasses the design, construction, operation, and application of robots. It finds diverse applications across numerous sectors. Currently, the advancement of robotics has led to many functional systems: robots are now replacing human labor in warehouses and delivering food with autonomous rovers. Once seen as a novelty, robots are gradually becoming an everyday reality. For instance, robotic vacuum cleaners have become so commonplace that they no longer surprise most people.

Regardless of their primary tasks, all robotic systems must address the challenges of localization, mapping, planning, execution, and control. Perception in robotics specifically focuses on the first two aspects. Effectively solving these tasks involves constructing a map of the environment to facilitate localization.

While robots perform a variety of functions, solving the localization problem and understanding the robot's state are important for addressing many other issues, such as planning.

In other words, every robotics task invariably includes components related to state estimation. In the context of robotics, "state" refers to a set of quantities—such as position, velocity, and orientation—that, when known, can accurately describe the robot's motion over time. Humans continually estimate their state on a subliminal level, often without being aware of this complex process. The process of state estimation in our brains occurs so unconsciously and naturally that we can consider our state an integral and inseparable part of our consciousness.

In robotics, we model these processes using mathematical models. Instead of relying on human senses, robots utilize various sensors, such as cameras, lidars, and IMUs (Inertial Measurement Units). Typically, a combination of these sensors is employed; no universal model is suitable for every robotics problem. Each task requires careful consideration, allowing for the selection of the optimal parametrization of the state. One approach to parameterizing the state involves considering how time is treated within a specific task. Continuous-time parametrization has dominated robotics and control for many decades due to the long-standing use of analog electronics. Nowadays, digital computers prevail, and discrete-time parametrization is the most common and proven solution for many problems; however, it can be inefficient for setups involving high-frequency sensors such as IMU or time-continuous data capture that is characteristic for LiDARs.

Another critical aspect is the representation of the state itself. There are various ways to mathematically parameterize the robot's rotation and position. The state—more precisely referred to as the "pose"—encompasses both orientation (rotation) and position, which can be expressed using combinations of vectors, angles, quaternions, etc.

The correct choice of parametrization is more crucial than it may initially appear. This significance stems from the nature of most state estimation algorithms, which typically involve solving some form of optimization problem. To tackle such problems, one often needs to compute a corresponding gradient related to the state and other optimization variables. Herein lies the importance of state parametrization, as it directly affects the dimensionality of the optimization problem and influences the efficiency of gradient computation—specifically, the Jacobian related to the optimization states.

In essence, the solution to many robotics tasks can be broadly generalized as selecting the appropriate state parametrization and finding an accurate, efficient method for computing the associated Jacobian. Therefore, the objective of this dissertation is to contribute to the understanding of these two aspects by exploring apprximated Jacobian computation for on manifold state estimation in robotics.

1.1 Motivation

We focus our efforts on the topic of state estimation, specifically investigating two challenging scenarios: time-continuous state estimation and visual localization with dense image alignment.

Pose estimation problems are often formulated using a discrete-time representation of a sequence of 3D poses over time—commonly referred to as a trajectory. This discrete-time representation is the default approach due to its established theory in state estimation and the clarity of the results produced Dellaert and Kaess [2006], Cunningham et al. [2013]. Moreover, many trajectory estimation approaches for robot-sensor systems successfully utilize a discrete-time formulation, as it allows for manageable state sizes Kaess et al. [2008], Wang et al. [2018]. However, this becomes problematic for setups that incorporate high-rate or asynchronous sensors Huai et al. [2021], Gohard et al. [2019], rendering the discrete-time formulation unsuitable for such scenarios without additional assumptions. A more straightforward solution is to adopt a time-continuous trajectory representation, which enables the integration of output from high-rate sensors, thus avoiding the estimation of a discrete state for each observation—a method that would otherwise result in an overwhelming number of state variables to estimate. Furthermore, time-continuous trajectories facilitate the natural fusion of multiple asynchronous sensors operating at various high rates. This representation also allows for the direct calculation of the Jacobian since most modern state estimation techniques are based around optimization.

Visual localization, the process of determining the camera pose using a visual representation of a known scene, is critical in various applications, including robot navigation, self-driving cars, and augmented/virtual reality Lynen et al. [2015], Heng et al. [2019]. The primary objective of visual localization through camera alignment is to determine the 6 degrees of freedom (6DoF) for the camera pose—its position and orientation—in a 3D environment represented by a known map, given an input query image.

The map representation of a known scene, a core component of every localization method, can take various forms. Recently, 3D Gaussian Splatting (3DGS) Kerbl

et al. [2023] has emerged, demonstrating high-quality real-time novel view synthesis at full HD resolution. Specifically, 3DGS represents the 3D scene with a collection of 3D anisotropic Gaussians that serve as rendering primitives, with parameters directly optimized from a set of available posed images during training. This innovative and distinctive map representation shows promise in effectively addressing the challenges associated with camera pose estimation and visual localization.

1.2 Objectives and contributions

While the two tasks mentioned may not be directly related, they are both relevant state estimation problems that share common requirements: the choice of state parametrization, the need to solve optimization problems to estimate the state, and the necessity of efficient and accurate Jacobian computation.

Optimization on manifold is recognized in the literature for its efficiency and accuracy in robotic tasks. This dissertation aims to explore relevant state estimation challenges in robotics—specifically time-continuous pose estimation and dense rendering-based image alignment—through the lens of optimization on manifold. This exploration involves deriving the relevant Jacobians and addressing all associated tasks and challenges.

This thesis makes several contributions to the field of state estimation in robotics. The contributions regarding time-continuous state estimation are as follows:

- **Contribution 1** Deriving an approximate form of the Jacobian for linearly interpolated poses on $\mathbf{SE}(3)$ using an approximation based on the commutativity property of infinitesimal group elements.

- **Contribution 2** Empirical evaluation of this approach on several synthetic and real-world datasets, demonstrating its consistency in robustly achieving accurate results of state estimation of 3D trajectories via optimization on manifold.

The contributions related to rendering-based visual state estimation are as follows:

- **Contribution 3** Investigating the viability of splatting-based rendering as a map representation. We analytically derive the gradients of the renderings with respect to camera poses for on manifold pose optimization and implement a visual localization pipeline. This pipeline is evaluated on indoor synthetic and real scenes, providing a comprehensive quantitative analysis.

- **Contribution 4** Exploring convergence limitations that arise from the highly non-convex nature of the dense localization problem. We propose a coarse-to-fine optimization strategy, applying gradually fading Gaussian blur to the query and rendered images. This approach helps overcome suboptimal convergence issues related to high-frequency image details. Additionally, we propose an effective method for improving localization results by enhancing camera initialization obtained through image retrieval, extending the image base with rendered camera frames.

1.3 Thesis Overview

The thesis is organised as follows:

- **Chapter 1: Introduction** This chapter outlines the motivations for undertaking this thesis and the original goals set at the outset of this research.

- **Chapter 2: Rigid Body Transformations using Lie Algebra for Robotics**

  This chapter describes the mathematical properties of rotation groups and rigid body transformations (RBT) that are necessary for the subsequent discussions.

- **Chapter 3: Jacobian Approximation for Direct Optimization of a Trajectory of Interpolated Poses on SE(3)** This chapter addresses the task of time-continuous state estimation and optimization on manifold related to the time-continuous trajectory representation using direct linear interpolation on $\mathbf{SE}(3)$.

- **Chapter 4: Analytical Jacobian for Camera Pose Estimation with Splatting Based Rendering** This chapter focuses on the task of rendering-based camera pose estimation and presents a solution using the proposed method of dense image alignment via optimization on manifold.

- **Chapter 5: Closing remarks**: Finally, this chapter summarizes and concludes the thesis.

Chapter 2

Rigid Body Transformations using Lie Algebra for Robotics

In this chapter we will describe the mathematical properties for the groups of rotations and RBT. We will also consider some interesting properties and how we can actually use these groups in multiple ways, such as state variables, vector transformations or frame operations.

Although Lie group theory encompasses a vast range of topics, this discussion has focused on key concepts and definitions critical to our analysis. The following material is firmly based on material presented in comprehensive manuscripts of Blanco-Claraco [2021], Barfoot [2017], Fernández-Madrigal and Blanco-Claraco [2013], Eade [2014], Sola et al. [2018] for which we recommend consulting readers seeking a deeper understanding of the subject and a more comprehensive insight.

2.1 Basic concepts

We focus our discussion around the 3-dimensional Euclidean space $\mathbb{R}^3$. The transformation in this space is defined as a function or mapping by

$$f: \quad \mathbb{R}^3 \to \mathbb{R}^3. \tag{2.1}$$

At this moment, let's define this mapping function by an arbitrary invertible

3×3 matrix $\mathbf{R}$ that maps point $\mathbf{p}$ to point $\widetilde{\mathbf{p}}$ as

$$\widetilde{\mathbf{p}} = \mathbf{R}\mathbf{p}. \tag{2.2}$$

The collection of all matrices $\mathbf{R}$ forms the general linear group represented as $\mathbf{GL}(3, \mathbb{R})$. Within the vast array of options for $\mathbf{R}$, the orthogonal matrices that satisfy $\mathbf{R}\mathbf{R}^\top = \mathbf{R}^\top\mathbf{R} = \mathbf{I_3}$ and have determinants of ± 1 define the orthogonal group, $\mathbf{O}(3) \subset \mathbf{GL}(3, \mathbb{R})$. It is essential to note that the operation for this group is simply the standard matrix multiplication, which guarantees that multiplying any two matrices in $\mathbf{O}(3)$ results in another matrix in $\mathbf{O}(3)$. These matrices correspond to isometries, or transformations that preserve distances between two points. Here, we specifically consider isometries with a determinant of $+1$, known as proper isometries, which together constitute the group of proper orthogonal transformations, or the special orthogonal group $\mathbf{SO}(3) \subset \mathbf{O}(3)$.

The matrices in $\mathbf{SO}(3)$ are related exclusively only to rotations. Therefore, it can be used to 1) transform vectors and rotate them into new reference frames; 2) to transform reference frames as well (with coincident origins); 3) another valuable application is to express orientations.

To incorporate translations, we utilize 4×4 transformation matrices $\mathbf{T}$ and expand 3D points by adding a fourth homogeneous coordinate as

$$\begin{pmatrix} \widetilde{p}_x \\ \widetilde{p}_y \\ \widetilde{p}_z \\ 1 \end{pmatrix} = \left(\begin{array}{ccc|c} & \mathbf{R} & & \begin{matrix} t_x \\ t_y \\ t_z \end{matrix} \\ \hline 0 & 0 & 0 & 1 \end{array}\right) \begin{pmatrix} p_x \\ p_y \\ p_z \\ 1 \end{pmatrix}. \tag{2.3}$$

Matrices $\mathbf{T}$ defined in this way are known as poses and form the group of affine rigid motions refereed as Special Euclidean Group $\mathbf{SE}(3) \subset \mathbf{GL}(4, \mathbb{R})$. We can also refer these matrices as Rigid Body Transformations (RBT).

As we defined, a pose is described with two parts: 3-dimensional translation vector $\mathbf{t}$ and 3×3 matrix $\mathbf{R}$ referred as rotation matrix. Rotation matrix $\mathbf{R}$ is special orthogonal and formed with columns that form orthonormal vector basis or

coordinate frame. Therefore, poses have overall 6 degrees of freedom and might be also refereed in literature as 6D poses.

The potential uses of $\mathbf{SE}(3)$ include

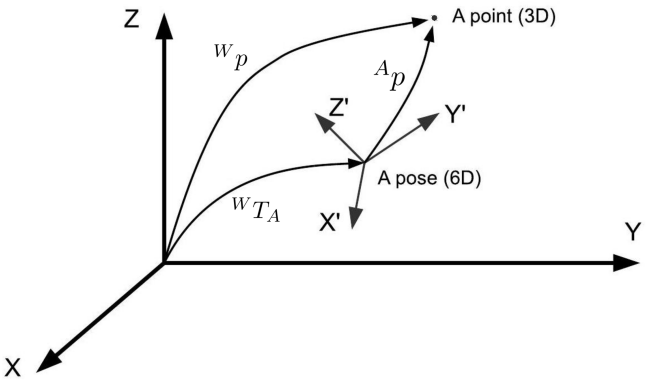


Figure 2-1: Schematic description of the point transformation Source: Blanco-Claraco [2021]

1. Transform points from one reference from to another as

$$^{W}\mathbf{p} = {}^{W}\mathbf{T}_{A}{}^{A}\mathbf{p}. \tag{2.4}$$

2. Transform reference frames as

$$^{W}\mathbf{T}_{B} = {}^{W}\mathbf{T}_{A}{}^{A}\mathbf{T}_{B}. \tag{2.5}$$

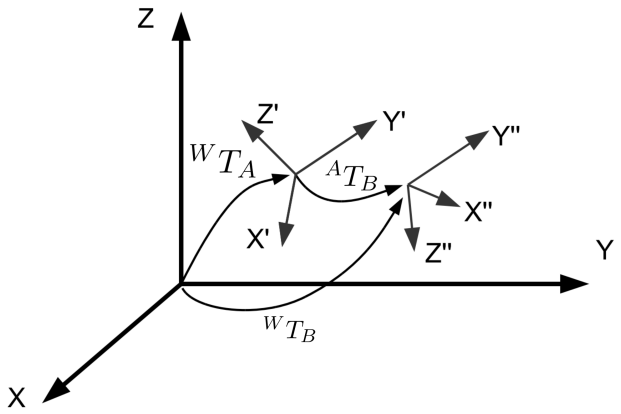


Figure 2-2: Schematic description of the pose composition Source: Blanco-Claraco [2021]

The 4×4 transformation matrices $\mathbf{T}$ are the way to represent 6D poses yet

quite not compact. Next we introduce some other commonly employed 6D pose parameterizations.

2.2 Common pose parameterizations

2.2.1 3D translation and Euler angles

Any 6D pose as we already shown consists of two parts: one related to translation, other to rotation. While the first one describes a translation displacement and is always defined as a 3D vector, the way of treating the rotation component might differ.

One approach is to define rotation with Euler angles such are yaw (ϕ), pitch (θ) and roll (ψ). In this way an arbitrary 6D pose is described as

$$\mathbf{P} = [x \;\; y \;\; z \;\; \phi \;\; \theta \;\; \psi]. \tag{2.6}$$

Any arbitrary rotation might be split in three independent rotation angles around different axis. There are different ways and conventions of defining these angles and axis. One of the most employed in robotics is represented in Figure 2-3. It starts with rotation around z-axis with yaw, continues with pitch around just just obtained y-axis and ends with a roll around modified x-axis.

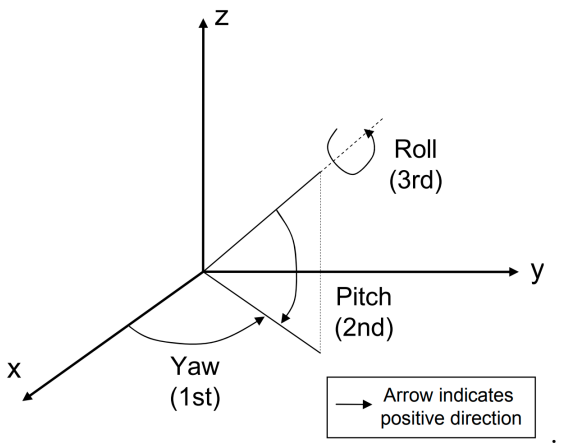


Figure 2-3: A common convention for the angles yaw, pitch and roll. Source: Fernández-Madrigal and Blanco-Claraco [2013]

This 6D pose representation has 6 degrees of freedom that is a minimal possible. Beside this advantage, there is one aspect that heavily limits the usage of this representation in favor of other less compact ones. This representation has degenerate situation named gimbal lock. It arises when pitch $\theta = \pm 90°$ causing alignment of z and x axis resulting in loss of one degree of freedom in the rotation representation. In short, changing roll becomes equivalent to changing yaw.

2.2.2 3D translation and Quaternion

Another way to describe rotation part in the 6D pose is quaternion defined as

$$\mathbf{P} = [x \;\; y \;\; z \;\; q_x \;\; q_y \;\; q_z \;\; q_r]. \tag{2.7}$$

Here, the unit quaternion elements are $[q_x, q_y, q_z, q_r]$. A simple intuition behind a quaternion is to threat an arbitrary rotation as a rotation by angle θ around some axis defined by $\vec{v} = (v_x, v_y, v_z)$ that are related to quaternion elements as

$$q_x = \sin\frac{\theta}{2}v_x; \quad q_y = \sin\frac{\theta}{2}v_y; \quad q_z = \sin\frac{\theta}{2}v_z; \quad q_r = \cos\frac{\theta}{2}; \tag{2.8}$$

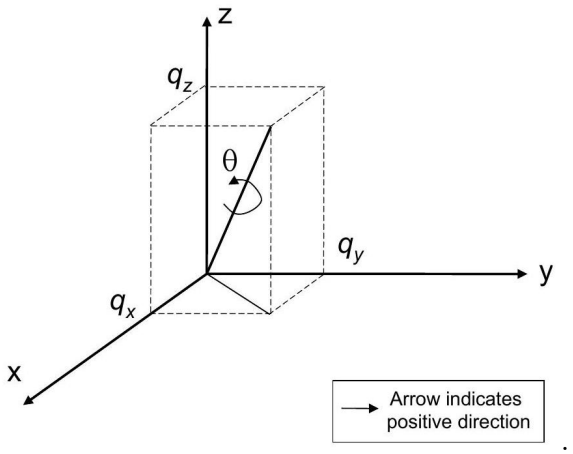


Figure 2-4: A quaternion can be seen as a rotation around an arbitrary 3D axis. Source: Fernández-Madrigal and Blanco-Claraco [2013]

2.3 Rigid Body Transformations using Lie Algebra

In this and all the following sections of the current chapter we will recite the mathematical properties of rotation groups and rigid body transformations (RBT) closely following the extensive material presented in Ferrer [2021], Barfoot [2017] and Eade [2018].

2.3.1 Rotations

All possible matrix rotations in 3D belong to the special orthogonal group defined as

$$\mathbf{SO}(3) = \left\{ \mathbf{R} \in \mathbb{R}^{3\times 3} \mid \mathbf{R}\mathbf{R}^\top = \mathbf{I} \wedge \det(\mathbf{R}) = 1 \right\}. \tag{2.9}$$

In this context, the binary operation between two elements of the group is matrix multiplication. Since matrix multiplication is non-commutative, this group is also non-commutative.

The four axioms of groups are:

- Closure: $\mathbf{R}_1, \mathbf{R}_2 \in \mathbf{SO}(3) \implies \mathbf{R}_1\mathbf{R}_2 \in \mathbf{SO}(3)$
- Associativity: $\mathbf{R}_1 \left(\mathbf{R}_2\mathbf{R}_3\right) = \left(\mathbf{R}_1\mathbf{R}_2\right)\mathbf{R}_3$
- Identity element: $\exists!\mathbf{I} \in \mathbf{SO}(3) : \mathbf{R}\mathbf{I} = \mathbf{I}\mathbf{R} = \mathbf{R}$. There exists a unique rotation $\mathbf{I}$ that satisfies this condition, and this element is the identity matrix.
- Inverse element: $\exists!\mathbf{R}^{-1} \in \mathbf{SO}(3) : \mathbf{R}\mathbf{R}^{-1} = \mathbf{I}$. From the definition of the group, we can derive that the inverse element is given by $\mathbf{R}^{-1} = \mathbf{R}^\top$.

The closure axiom implies that multiple rotations can be combined, yielding a valid rotation as a result of this sequence of operations. It is important to consider the order of the rotations, as the group operation is matrix multiplication, which is generally non-commutative as

$$\mathbf{R}_1\mathbf{R}_2 \neq \mathbf{R}_2\mathbf{R}_1.$$

2.3.2 Rigid Body Transformations

Analogous to $\mathbf{SO}(3)$, all possible rigid body transformation (RBT) matrices conform to the Special Euclidean group as

$$\mathbf{SE}(3) = \left\{ \mathbf{T} = \left[\begin{array}{cc} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{array} \right] \middle| \, \mathbf{R} \in \mathbf{SO}(3) \wedge \mathbf{t} \in \mathbb{R}^3 \right\}. \tag{2.10}$$

This group represents the result of a rotation followed by a translation, with the group operation defined as matrix multiplication.

The four axioms of groups are also satisfied:

- Closure: $\mathbf{T}_1, \mathbf{T}_2 \in \mathbf{SE}(3) \Longrightarrow \mathbf{T}_1\mathbf{T}_2 \in \mathbf{SE}(3)$

  The combination of two transformation matrices is represented as

$$\mathbf{T}_1\mathbf{T}_2 = \left[\begin{array}{cc} \mathbf{R}_1 & \mathbf{t}_1 \\ 0 & 1 \end{array} \right] \left[\begin{array}{cc} \mathbf{R}_2 & \mathbf{t}_2 \\ 0 & 1 \end{array} \right] = \left[\begin{array}{cc} \mathbf{R}_1\,\mathbf{R}_2 & \mathbf{R}_1\mathbf{t}_2 + \mathbf{t}_1 \\ 0 & 1 \end{array} \right] \in \mathbf{SE}(3)$$

- Associativity: $\mathbf{T}_1\left(\mathbf{T}_2\mathbf{T}_3\right) = \left(\mathbf{T}_1\mathbf{T}_2\right)\mathbf{T}_3$.

- Identity element: $\exists!\mathbf{I} \in \mathbf{SE}(3)$ such that $\mathbf{TI} = \mathbf{T}$. There exists a unique identity element in the group corresponding to a rigid body transformation. Specifically, this is the 4×4 identity matrix.

- Inverse element: $\exists!\mathbf{T}^{-1} \in \mathbf{SE}(3)$ such that $\mathbf{T}\mathbf{T}^{-1} = \mathbf{I}$. From the definition, we can rearrange terms to show that the inverse corresponds to

$$\mathbf{T}^{-1} = \left[\begin{array}{cc} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{array} \right]^{-1} = \left[\begin{array}{cc} \mathbf{R}^\top & -\mathbf{R}^\top\mathbf{t} \\ 0 & 1 \end{array} \right] \in \mathbf{SE}(3). \tag{2.11}$$

As a result of the closure axiom, one can chain a sequence of rigid body transformations (RBT) to obtain a valid transformation, much like in the case of rotations. The physical interpretation is that a sequence of different frames can be composed to form a general frame.

For RBTs, the order of transformations is also significant following

$$\mathbf{T}_1\mathbf{T}_2 \neq \mathbf{T}_2\mathbf{T}_1,$$

where the left-hand side and the right-hand side represent elements of the group, but they are generally not equal.

2.4 Lie Algebra for Rotations $\mathbf{SO}(3)$

This section is devoted to explaining the Lie algebra for rotations. Following the Ferrer [2021], the same principles can later be applied to derive similar results for rigid body transformations in $\mathbf{SE}(3)$.

2.4.1 Infinitesimal Increments over Rotations $\mathbf{SO}(3)$

First, it is important to understand the structure of infinitesimal variations in a rotation matrix.

As discussed previously, $\mathbf{R}$ is orthonormal and has a positive determinant, which constrains the space of solutions in differential form. A natural question arises regarding the group of rotations and rigid body transformations: What is the most efficient minimal representation, and how many degrees of freedom are involved?

To illustrate this, let's consider a rotation $\mathbf{R} \in \mathbf{SO}(3)$, and examine a smooth rotation that provides an infinitesimal update to $\mathbf{R}$ over time as

$$\dot{\mathbf{R}} = \mathbf{W}\mathbf{R} \quad \text{s.t.} \quad \mathbf{R}\mathbf{R}^\top = \mathbf{I}, \tag{2.12}$$

$$\dot{\mathbf{R}}\mathbf{R}^\top + \mathbf{R}(\dot{\mathbf{R}})^\top = 0, \tag{2.13}$$

$$\mathbf{W}\underbrace{\mathbf{R}\mathbf{R}^\top}_{\mathbf{I}} + \underbrace{\mathbf{R}\mathbf{R}^\top}_{\mathbf{I}}\mathbf{W}^\top = 0 \Longleftrightarrow \mathbf{W} = -\mathbf{W}^\top. \tag{2.14}$$

The group of matrices that satisfy 2.14 is known as Skew symmetric matrices. For 2D, the group looks like this $\begin{bmatrix} 0 & -\omega \\ \omega & 0 \end{bmatrix}$ and for 3D rotations

$$\mathbf{W} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} = \boldsymbol{\omega}^{\wedge}. \tag{2.15}$$

The hat operator $(\cdot)^{\wedge}$ denotes the construction of a skew-symmetric matrix. In fact, the group is also a Lie group; in particular, $\boldsymbol{\omega}^{\wedge} \in \mathfrak{so}(3)$ around the identity element ($\mathbf{R} = \mathbf{I}$) is referred to as the tangent space at the identity.

2.4.2 The Exponential Map

We have derived a differential form for rotations, given by $\boldsymbol{\omega}^{\wedge} \in \mathfrak{so}(3)$. The solution to the differential equation is of the form

$$\mathbf{R}(t) = e^{\boldsymbol{\omega}^{\wedge} t}\mathbf{R}(t_0), \tag{2.16}$$

where the resultant rotation is a function of time t. We need to solve this equation.

This integration requires a constant $\boldsymbol{\omega}$ and a final time, which we will set to $t = 1$, for instance. Some authors, in an abuse of notation, continue to use the angular velocity notation. We will adopt a different convention to distinguish between the derivative with units of [rad/s] and a simple angle. Accordingly, we define some angle $\theta \in \mathbb{R}^3$ that related to $\boldsymbol{\omega}$ as $\boldsymbol{\theta}^{\wedge} = \boldsymbol{\omega}^{\wedge} t \,|\, t = 1$.

Next, we will also consider the Taylor expansion around the identity rotation such as

$$\exp\left(\boldsymbol{\theta}^{\wedge}\right) = \mathbf{I} + \boldsymbol{\theta}^{\wedge} + \frac{1}{2!}\left(\boldsymbol{\theta}^{\wedge}\right)^2 + \frac{1}{3!}\left(\boldsymbol{\theta}^{\wedge}\right)^3 + \ldots = \sum_{n}^{\infty} \frac{1}{n!}\left(\boldsymbol{\theta}^{\wedge}\right)^n. \tag{2.17}$$

Skew-symmetric matrices exhibit a recursive property that is very useful, where $\theta = \|\boldsymbol{\theta}\|_2$ such as

$$\left(\boldsymbol{\theta}^{\wedge}\right)^2 = \boldsymbol{\omega}\boldsymbol{\omega}^{\top} - \theta^2\mathbf{I}, \tag{2.18}$$

$$\theta^2 = \omega_1^2 + \omega_2^2 + \omega_3^2, \tag{2.19}$$

$$\left(\boldsymbol{\theta}^{\wedge}\right)^{3} = \left(\boldsymbol{\omega}\boldsymbol{\omega}^{\top} - \theta^{2}\mathbf{I}\right)\boldsymbol{\theta}^{\wedge} = 0 - \theta^{2}\boldsymbol{\theta}^{\wedge}. \tag{2.20}$$

Continuing this process, one can derive a closed form for the series that arises from the simplification provided by skew-symmetric matrices, ultimately yielding the well-known Rodrigues [1840] formula

$$\mathbf{R} = \exp\left(\boldsymbol{\theta}^{\wedge}\right) = \mathbf{I} + \frac{\sin(\theta)}{\theta}\boldsymbol{\theta}^{\wedge} + \frac{1-\cos(\theta)}{\theta^{2}}\left(\boldsymbol{\theta}^{\wedge}\right)^{2}. \tag{2.21}$$

The exponential map is a surjective function, as a unique rotation can be achieved from different values of ω. The analogy with a 1D angle α is evident, where multiple values of $\alpha' = \alpha + i2\pi, \forall i \in \mathbb{Z}$ represent the same angle α.

This differential form is characterized by three variables as

$$\boldsymbol{\theta}^{\wedge} = \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}}_{G_1}\theta_1 + \underbrace{\begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix}}_{G_2}\theta_2 + \underbrace{\begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}}_{G_3}\theta_3. \tag{2.22}$$

The elements generated by linear combinations of matrices G_i referred also as Lie algebra matrix generators span a vector space. The tangent space $\mathfrak{so}(3)$ around the identity element in the Lie group closely resembles a 3D Euclidean space $\mathbb{R}^3$.

There is a sequence of operations that maps rotations to the tangent space defined as

$$(\cdot)^{\wedge} : \mathbb{R}^3 \to \mathfrak{so}(3)$$
$$\exp\left(\boldsymbol{\theta}^{\wedge}\right) : \mathfrak{so}(3) \to \mathbf{SO}(3).$$

In an abuse of notation, we can define the (capital) exponent as a composition of the functions above, which directly maps the manifold to rotations as

$$\mathrm{Exp}(\boldsymbol{\theta}) : \mathbb{R}^3 \to \mathbf{SO}(3). \tag{2.23}$$

Valuable properties of the exponent $\mathbf{R} = \mathrm{Exp}(\boldsymbol{\omega})$

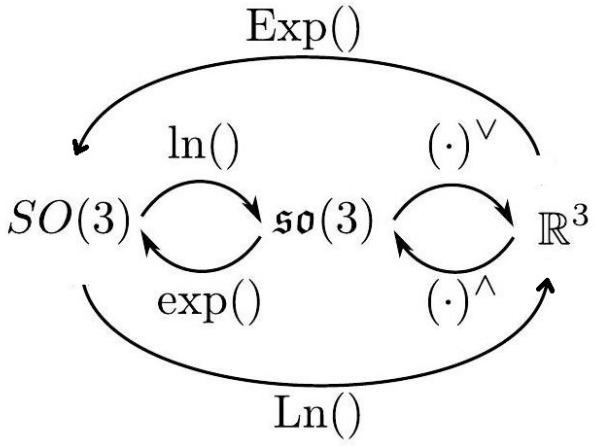


Figure 2-5: Mapping functions for $\mathbf{SO}(3)$

$$\mathrm{Exp}(-\boldsymbol{\theta}) = \mathbf{R}^{-1} = \mathbf{R}^\top, \tag{2.24}$$

$$\mathrm{Exp}(\tau\boldsymbol{\theta}) = \mathrm{Exp}(\boldsymbol{\theta})^\tau. \tag{2.25}$$

The inverse solution is the logarithm that can be obtained by writing the series of a rotation and its inverse (transpose) derived as

$$\begin{aligned}
\mathbf{R} &= \mathbf{I} + \frac{\sin(\theta)}{\theta}\boldsymbol{\theta}^\wedge + \frac{1-\cos(\theta)}{\theta^2}\left(\boldsymbol{\theta}^\wedge\right)^2 \\
\mathbf{R}^{-1} = \mathbf{R}^\top &= \mathbf{I} - \frac{\sin(\theta)}{\theta}\boldsymbol{\theta}^\wedge + \frac{1-\cos(\theta)}{\theta^2}\left(\boldsymbol{\theta}^\wedge\right)^2 \\
\mathbf{R} - \mathbf{R}^\top &= 0 + 2\frac{\sin(\theta)}{\theta}\boldsymbol{\theta}^\wedge + 0,
\end{aligned}$$

which leads to the expression

$$\boldsymbol{\theta}^\wedge = \log(\mathbf{R}) = \frac{\theta}{2\sin\theta}\left(\mathbf{R} - \mathbf{R}^\top\right). \tag{2.26}$$

The value of θ can be determined similarly by summing both expressions as

$$\begin{aligned}
\mathbf{R} + \mathbf{R}^\top &= 2\mathbf{I} + 0 + 2\frac{1-\cos(\theta)}{\theta^2}\left(\boldsymbol{\theta}^\wedge\right)^2 \\
\mathrm{Tr}\left(\mathbf{R} + \mathbf{R}^\top\right) &= 2\,\mathrm{Tr}(\mathbf{I}) + 2\frac{1-\cos(\theta)}{\theta^2}\,\mathrm{Tr}\left(\left(\boldsymbol{\theta}^\wedge\right)^2\right) \\
2\,\mathrm{Tr}(\mathbf{R}) &= 2\;3 + 2\frac{1-\cos(\theta)}{\theta^2}\,\mathrm{Tr}\left(\omega\omega^\top - \theta^2\mathbf{I}\right) \\
\mathrm{Tr}(\mathbf{R}) &= 3 + \frac{1-\cos(\theta)}{\theta^2}\left(\theta^2 - 3\theta^2\right) \Longrightarrow 2\cos(\theta) = \mathrm{Tr}(\mathbf{R}) - 1
\end{aligned}$$

which can be rearranged into the following equation to obtain θ as

$$\theta = \arccos\left(\frac{\mathrm{Tr}(\mathbf{R}) - 1}{2}\right). \tag{2.27}$$

The inverse operation is known as the logarithm, which first maps a rotation to the Lie algebra as

$$\ln(\mathbf{R}) : \mathbf{SO}(3) \to \mathfrak{so}(3)$$

and then maps from the Lie algebra to the manifold

$$(\cdot)^{\vee} : \mathfrak{so}(3) \to \mathbf{R}^3.$$

In similar manner Exp(), can be defined as a function that first maps a rotation to the Lie algebra and then to the manifold $\mathbb{R}^3$ as

$$\mathrm{Ln}(\mathbf{R}) : \mathbf{SO}(3) \to \mathbb{R}^3.$$

2.5 Lie Algebra for RBT $\mathbf{SE}(3)$

Following Ferrer [2021] the similar reasoning can be applied for rigid body transformations (RBT) via

$$\dot{\mathbf{T}} = \mathcal{W}\mathbf{T}, \quad \text{s.t.} \quad \mathbf{T} \in \mathbf{SE}(3), \tag{2.28}$$

where $\mathcal{W}$ represents a twist of 3D poses. If we expand this further, we obtain

$$\mathcal{W} = \dot{\mathbf{T}}\mathbf{T}^{-1} = \begin{bmatrix} \dot{\mathbf{R}} & \dot{\mathbf{t}} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R}^{\top} & -\mathbf{R}^{\top}\mathbf{t} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \dot{\mathbf{R}}\mathbf{R}^{\top} & -\dot{\mathbf{R}}\mathbf{R}^{\top}\mathbf{t} + \dot{\mathbf{t}} \\ 0 & 0 \end{bmatrix}. \tag{2.29}$$

One can identify the same result previously derived for $\mathbf{SO}(3)$, along with an additional term related to the rotated derivative of the translation as

$$\mathcal{W} = \begin{bmatrix} \boldsymbol{\omega}^{\wedge} & \boldsymbol{v} \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 & v_1 \\ \omega_3 & 0 & -\omega_1 & v_2 \\ -\omega_2 & \omega_1 & 0 & v_3 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{2.30}$$

Here, the components related to rotation, representing orientation, are denoted by $\boldsymbol{\omega}$ (angular velocities), while the components associated with the translation vector are represented as $\boldsymbol{v}$ (linear velocities).

This twist can be integrated to obtain a rigid body transformation that solves the differential equation 2.28

$$\mathbf{T}(t) = e^{\mathcal{W}t}\, \mathbf{T}\left(t_0\right). \tag{2.31}$$

Defining 6D-vector of coordinates in the Lie algebra $\mathfrak{se}(3)$ as

$$\boldsymbol{\xi} = \begin{bmatrix} \boldsymbol{\rho} \\ \boldsymbol{\theta} \end{bmatrix}, \tag{2.32}$$

this vector consists of two separate 3D components: $\boldsymbol{\theta}$, which defines the rotation, and $\boldsymbol{\rho}$, which defines the translation. As a result, the integration of the twist, composed of angular and linear velocities, is assumed to be constant over a fixed duration of time $t = 1$, expressed as

$$\mathcal{W}|_{t=1} = \boldsymbol{\xi}^{\wedge} = \begin{bmatrix} \boldsymbol{\theta}^{\wedge} & \boldsymbol{\rho} \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -\theta_3 & \theta_2 & \rho_1 \\ \theta_3 & 0 & -\theta_1 & \rho_2 \\ -\theta_2 & \theta_1 & 0 & \rho_3 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \tag{2.33}$$

where there are 6 elements on the 4×4 matrix of generators.

The Lie algebra $\mathfrak{se}(3)$ associated with the group of RBT $\mathbf{SE}(3)$ represents the group of infinitesimal RBT around the identity ($\mathcal{W} = \dot{\mathbf{T}}$). There exist operators that relate both groups.

In particular, the exponent map operator $\exp : \mathfrak{se}(3) \rightarrow \mathbf{SE}(3)$ is well-defined,

surjective, and has the closed form

$$\mathbf{T} = \mathrm{Exp}(\boldsymbol{\xi}) = \exp(\boldsymbol{\xi}^\wedge) = \begin{pmatrix} \mathrm{Exp}(\boldsymbol{\theta}^\wedge) & \mathbf{J}\boldsymbol{\rho} \\ 0 & 1 \end{pmatrix}, \tag{2.34}$$

$$\mathbf{J} = \mathbf{I_3} + \frac{1-\cos\theta}{\theta^2}\boldsymbol{\theta}^\wedge + \frac{\theta - \sin\theta}{\theta^3}\left(\boldsymbol{\theta}^\wedge\right)^2. \tag{2.35}$$

Here $\mathbf{J}$ is known as left Jacobian of $\mathbf{SO}(3)$ and $\theta = |\boldsymbol{\theta}|$

Valuable properties of the exponent $\mathbf{T} = \mathrm{Exp}(\boldsymbol{\xi})$

$$\mathrm{Exp}(-\boldsymbol{\xi}) = \mathbf{T}^{-1},$$
$$\mathrm{Exp}(\tau\boldsymbol{\xi}) = \mathrm{Exp}(\boldsymbol{\xi})^\tau$$

The inverse - logarithm map $\ln : \mathbf{SE}(3) \to \mathfrak{se}(3)$

$$\boldsymbol{\theta} = (\log \mathbf{R})^\vee, \tag{2.36}$$

$$\boldsymbol{\rho} = \mathbf{J}^{-1}\mathbf{t}, \tag{2.37}$$

where $\mathbf{R}$ and $\mathbf{t}$ rotation and translation in $\mathbf{SE}(3)$ pose. And $\mathbf{J}^{-1}$ is an inverse of left Jacobian .

$$\mathbf{J}^{-1} = \mathbf{I}_3 - \frac{1}{2}\boldsymbol{\theta}^\wedge + \left(\frac{1}{\theta^2} - \frac{1}{2\theta}\cot(\frac{\theta}{2})\right)\left(\boldsymbol{\omega}^\wedge\right)^2 \tag{2.38}$$

2.5.1 Adjoint of an element of $\mathbf{SE}(3)$

For some pose $\mathbf{T} \in \mathbf{SE}(3)$ we are looking for conditions in which

$$\mathbf{T}\,\mathrm{Exp}(\boldsymbol{\xi}) = \mathrm{Exp}\left(Adj_T\ \boldsymbol{\xi}\right)\ \mathbf{T},$$
$$\mathrm{Exp}\left(Adj_T\boldsymbol{\xi}\right) = \mathbf{T}\ \mathrm{Exp}(\boldsymbol{\xi})\mathbf{T}^{-1}$$

hold true for some $\boldsymbol{\xi}$ that is a vector in the manifold and expresses a transformation. Following Ferrer [2021] and leaving the detailed derivation out of scope the needed

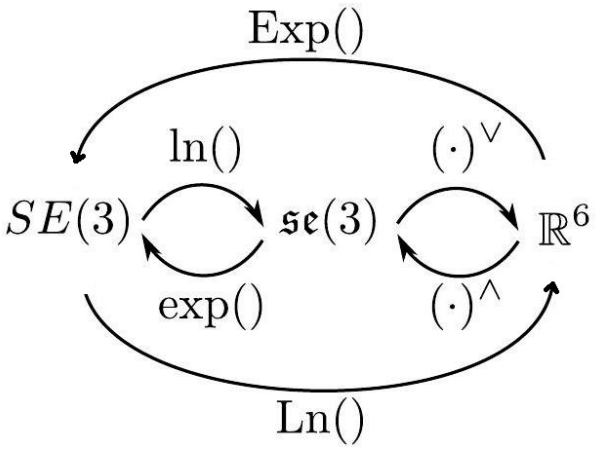


Figure 2-6: Mapping functions for $\mathbf{SE}(3)$

variable $Adj_{\mathbf{T}}$ is formulated as

$$Adj_{\mathbf{T}} = \begin{bmatrix} \mathbf{R} & \mathbf{t}^\wedge\mathbf{R} \\ 0 & \mathbf{R} \end{bmatrix}_{6\times 6} \tag{2.39}$$

and has a physical interpretation that it provides a linear transformation of coordinates in the tangent space around the identity to coordinates in the tangent space around $\mathbf{T}$.

2.6 Optimization problems on $\mathbf{SE}(3)$

Having discussed the essential concepts necessary to operate within $\mathbf{SE}(3)$ as a manifold, we will now explore solutions to some corresponding practical tasks.

This section based upon the material presented by Blanco-Claraco [2021] in meticulously crafted tutorial, adhering to the logical framework and structural organization employed.

2.6.1 Optimization Algorithms in Flat Euclidean Spaces

All optimization algorithms, regardless of their complexity, fundamentally implement the same basic process. For optimization on Euclidean space, both Gradient Descent and Gauss-Newton algorithms aim to minimize the sum of squared errors $S(\mathbf{x})$ between a prediction $\mathbf{f}(\mathbf{x})$ and an observation $\mathbf{z}$. It is achieved by iteratively refining

the corresponding state vector $\mathbf{x} \in \mathbb{R}^N$ until convergence as $\boldsymbol{x} \leftarrow \boldsymbol{x} + \boldsymbol{\delta}$, where the increments $\boldsymbol{\delta}$ are found as the solution to the equation

$$\left.\frac{\partial S(\mathbf{x} + \boldsymbol{\delta})}{\partial \boldsymbol{\delta}}\right|_{\boldsymbol{\delta}=0} = 0. \tag{2.40}$$

However, what occurs if our state vector $\mathbf{x}$ includes an element from $\mathbf{SE}(3)$? Optimization algorithms are typically designed to find solutions within Euclidean spaces. This necessitates that we handle poses as vectors, utilizing one of the previously described parametrizations:

1. Using 3D and Euler angles provides a minimal representation with 6 elements. However, this approach requires renormalization of the angles at each iteration to maintain them within a valid range, and it is susceptible to degenerative gimbal lock, making it a less favorable choice.

2. 3D and Quaternions avoid these issues, but they introduce an excessive 7th degree of freedom (DoF), complicating the search for a solution.

3. A full 4×4 matrix representation comprises 16 elements, resulting in even more excessive DoFs.

Therefore, representing poses as vectors in state estimation may present challenges in achieving an optimal solution. Among the options mentioned, only 3D with Quaternions does not present critical issues and is adequately utilized in some algorithms. However, a more stable approach exists: optimization on manifold.

2.6.2 Optimization on Manifolds

In this case, the optimization problem and its iterative solution for a given pose $\mathbf{T} \in \mathbf{SE}(3)$ can be expressed as

$$\boldsymbol{\xi}^\star \leftarrow \left.\frac{\partial S(\mathrm{Exp}(\boldsymbol{\xi})\mathbf{T})}{\partial \boldsymbol{\xi}}\right|_{\boldsymbol{\xi}=0} = 0, \tag{2.41}$$

$$\mathbf{T} \leftarrow \mathrm{Exp}(\boldsymbol{\xi}^\star)\mathbf{T}. \tag{2.42}$$

This follows a left-hand convention for updating the transformation.

2.7 Differentiation and Valuable Jacobians

The purpose of this section is to derive the derivative of any function with respect to a matrix transformation $\frac{\partial f(\mathbf{T})}{\partial \mathbf{T}}$.

In general, this operation is ill-posed, as the definition of differentiation does not typically consider matrices. To address this, we follow Blanco-Claraco [2021], Ferrer [2021] and change variables between a transformation matrix and its Lie algebra coordinates expressed in the manifold.

Two alternative procedures for differentiating functions with respect to rigid body transformations will be discussed. All Jacobians derived in this section are directly applied in subsequent chapters of the thesis related to mentioned problems of state estimation.

2.7.1 Differentiation: First Order Approximation

We aim to differentiate the function $f(\mathbf{T}) : \mathbf{SE}(3) \to \mathcal{K}$, where the image of this function $\mathcal{K}$ could be a constant, vector, element of $\mathbf{SE}(3)$ or $\mathbf{SO}(3)$. The approach is to define an auxiliary function for the increments in $\Delta\boldsymbol{\xi}$ such that

$$f(\mathrm{Exp}(\Delta\boldsymbol{\xi})\mathbf{T}) = f_{\mathbf{T}}(\Delta\boldsymbol{\xi}),$$

and then apply standard differentiation rules to an input that no longer belongs to $\mathbf{SE}(3)$, but rather to the manifold of rigid body transformations $\mathbb{R}^6$.

We will employ the approximation

$$\mathrm{Exp}(\boldsymbol{\xi}) \simeq \mathbf{I} + \boldsymbol{\xi}^{\wedge}, \tag{2.43}$$

which corresponds to a first-order Taylor expansion. This approximation is sufficiently accurate for small increments near the identity element ($\boldsymbol{\xi} = 0$), as the higher-order terms become negligible in this context. Therefore, we will proceed to calculate the

derivative of the exponential function as

$$\frac{\partial \operatorname{Exp}(\boldsymbol{\xi})}{\partial \boldsymbol{\xi}} \simeq \frac{\partial}{\partial \boldsymbol{\xi}} \left(\mathbf{I} + \boldsymbol{\xi}^{\wedge}\right) = \frac{\partial}{\partial \boldsymbol{\xi}} \begin{bmatrix} 0 & -\theta_3 & \theta_2 & \rho_1 \\ \theta_3 & 0 & -\theta_1 & \rho_2 \\ -\theta_2 & \theta_1 & 0 & \rho_3 \\ 0 & 0 & 0 & 0 \end{bmatrix} = \boldsymbol{G}, \tag{2.44}$$

where $\mathbf{G}$ is a Jacobian matrix, and each component $\boldsymbol{G}(i) = G_i$ represents a Lie algebra matrix generator. The standard method for computing the derivative of matrix expressions is to vectorize them and treat them as vectors via

$$\operatorname{vec}\left(\begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{bmatrix}\right) = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{pmatrix}. \tag{2.45}$$

For some $k \times l$ matrix $\mathbf{A}(\mathbf{B})$ and related $n \times m$ matrix $\mathbf{B}$ we can derive that

$$\frac{\partial \mathbf{A}(\mathbf{B})}{\partial \mathbf{B}} = \frac{\partial vec(\mathbf{A}(\mathbf{B}))}{\partial vec(\mathbf{B})} = \frac{\partial \left[A_{11}A_{21}\dots A_{k1}A_{12}\dots A_{kl-1}A_{1l}\dots A_{kl}\right]}{\partial \left[B_{11}B_{21}\dots B_{n1}B_{12}\dots B_{nm-1}B_{1m}\dots B_{nm}\right]} \tag{2.46}$$

$$= \begin{pmatrix} \frac{\partial A_{11}}{\partial B_{11}} & \frac{\partial A_{11}}{\partial B_{21}} & \dots & \frac{\partial A_{11}}{\partial B_{nm}} \\ \dots & \dots & \dots & \dots \\ \frac{\partial A_{kl}}{\partial B_{11}} & \frac{\partial A_{kl}}{\partial B_{21}} & \dots & \frac{\partial A_{kl}}{\partial B_{nm}} \end{pmatrix}_{kl \times nm}. \tag{2.47}$$

Another useful property is relates vectorization operation and Kronecker product as

$$\operatorname{vec}(\mathbf{ABC}) = \left(\mathbf{C}^{\mathrm{T}} \otimes \mathbf{A}\right) \operatorname{vec}(\mathbf{B}), \tag{2.48}$$

$$\frac{\partial \operatorname{vec}(\mathbf{ABC})}{\partial \operatorname{vec}(\mathbf{B})} = \mathbf{C}^{\mathrm{T}} \otimes \mathbf{A}. \tag{2.49}$$

In this way, to find the derivative of exponent function we first apply approximation

from equation 2.43 similarly as we did in first step of equation 2.44. Afterwards, we vectorize the resulting skew matrix detailed in equation 2.44 and derive the result as 12×6 Jacobian

$$\frac{\partial \operatorname{Exp}(\boldsymbol{\xi})}{\partial \boldsymbol{\xi}} \simeq \frac{\partial}{\partial \boldsymbol{\xi}} \left(\operatorname{vec}(\boldsymbol{\xi}^{\wedge})\right) = \begin{pmatrix} \mathbf{0}_{3\times3} & -\mathbf{e}_1^{\wedge} \\ \mathbf{0}_{3\times3} & -\mathbf{e}_2^{\wedge} \\ \mathbf{0}_{3\times3} & -\mathbf{e}_3^{\wedge} \\ \mathbf{I}_3 & \mathbf{0}_{3\times3} \end{pmatrix}_{12\times6}, \tag{2.50}$$

where $\mathbf{e_1} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^{\top}, \mathbf{e_2} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^{\top}$ and $\mathbf{e_3} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^{\top}$.

2.7.2 Example: Transforming a Vector

To illustrate this concept, let's consider the example of transforming a vector.

The function $f(\mathbf{T}) = \mathbf{Tp}$ represents a standard transformation of a vector $\mathbf{p}$ in homogeneous coordinates.

We are interested in the derivative of $\operatorname{Exp}(\boldsymbol{\xi})\mathbf{Tp}$ with respect to $\boldsymbol{\xi}$. To accomplish this, we first expand the expression using the properties of the chain rule. We then apply the property described in equation 2.49 to the first two factors and use the result obtained in 2.50 resulting in

$$\begin{aligned} \frac{\partial \left(\operatorname{Exp}(\boldsymbol{\xi})\mathbf{T}\right)\mathbf{p}}{\partial \boldsymbol{\xi}}\bigg|_{\boldsymbol{\xi}=0} &= \frac{\partial \left(\operatorname{Exp}(\boldsymbol{\xi})\mathbf{T}\right)\mathbf{p}}{\partial \left(\operatorname{Exp}(\boldsymbol{\xi})\mathbf{T}\right)} \frac{\partial \operatorname{Exp}(\boldsymbol{\xi})\mathbf{T}}{\partial \operatorname{Exp}(\boldsymbol{\xi})} \frac{\partial \operatorname{Exp}(\boldsymbol{\xi})}{\partial \boldsymbol{\xi}}\bigg|_{\boldsymbol{\xi}=0} = \\ &= \left[\begin{bmatrix} \mathbf{p}^{\top} & 1 \end{bmatrix} \otimes \mathbf{I_3}\right]_{3\times12} \left[\mathbf{T}^{\top} \otimes \mathbf{I}_3\right]_{12\times12} \begin{pmatrix} \mathbf{0}_{3\times3} & -\mathbf{e}_1^{\wedge} \\ \mathbf{0}_{3\times3} & -\mathbf{e}_2^{\wedge} \\ \mathbf{0}_{3\times3} & -\mathbf{e}_3^{\wedge} \\ \mathbf{I}_3 & \mathbf{0}_{3\times3} \end{pmatrix}_{12\times6} = \\ &= \begin{pmatrix} \mathbf{I}_3 & -[\mathbf{Tp}]^{\wedge} \end{pmatrix}_{3\times6}. \end{aligned} \tag{2.51}$$

2.7.3 Differentiation: Small Perturbations

Following the structure and the idea presented in Eade [2018], let us consider the function $\boldsymbol{g} : \mathbf{SE}(3) \to \mathbf{SE}(3)$, where both the input and output are rigid body

transformations (RBT).

For any small perturbation in the input of $\boldsymbol{g}$, expressed as

$$\boldsymbol{g}(\mathrm{Exp}(\boldsymbol{\delta})\mathbf{T}) = \mathrm{Exp}(\boldsymbol{\epsilon})\boldsymbol{g}(\mathbf{T}), \tag{2.52}$$

the function results in a corresponding small perturbation in the output. Here, we have adopted a left-hand convention. Expanding the previous term yields

$$\boldsymbol{\epsilon} = \mathrm{Ln}\left(\boldsymbol{g}(\mathrm{Exp}(\boldsymbol{\delta})\mathbf{T})\,\boldsymbol{g}(\mathbf{T})^{-1}\right. . \tag{2.53}$$

Note that we have defined a new function for the perturbation of $\boldsymbol{g}$ expressed in the coordinates of the tangent space. As a result, the Jacobian we obtain with the new input $\Delta\boldsymbol{\xi} \in \mathbb{R}^6$ and the output $\boldsymbol{\epsilon} \in \mathbb{R}^6$ is represented as a 6×6 matrix.

Now, the derivative of the function $\boldsymbol{g}(\cdot)$ with respect to the transformation $\mathbf{T}$ is equivalent to the derivative of $\boldsymbol{\epsilon}$ with respect to $\boldsymbol{\delta}$

$$\frac{\partial \boldsymbol{g}}{\partial \mathbf{T}} = \frac{\partial \boldsymbol{\epsilon}}{\partial \boldsymbol{\delta}} = \frac{\partial}{\partial \boldsymbol{\delta}} \left(\mathrm{Ln}\left[\boldsymbol{g}(\mathrm{Exp}(\boldsymbol{\delta})\mathbf{T})\ \boldsymbol{g}(\mathbf{T})^{-1}\right]\right) |_{\boldsymbol{\delta}=0} = 0. \tag{2.54}$$

Please refer to 2.4.2 in Eade [2018] for more details. We will illustrate how this notation proves to be useful in the following examples.

2.7.4 Example: Direct Observation of a Pose

Consider the function $f(\mathbf{T}) : \mathbf{SE}(3) \rightarrow \mathbf{SE}(3)$ defined as

$$g(\mathbf{T}) = \mathbf{T}_{\mathrm{obs}}\ \mathbf{T}^{-1}. \tag{2.55}$$

This function computes the difference between the transformation $\mathbf{T}$ and the observed transformation $\mathbf{T}_{\mathrm{obs}}$. When these transformations perfectly match, i.e., $\mathbf{T} = \mathbf{T}_{\mathrm{obs}}$, the output will yield the identity element.

Now, applying (2.54) we obtain

$$
\begin{aligned}
\frac{\partial \boldsymbol{g}}{\partial \mathbf{T}} &= \frac{\partial}{\partial \Delta\boldsymbol{\xi}} \left(\operatorname{Ln} \left[\mathbf{T}_{obs} (\operatorname{Exp}(\Delta\boldsymbol{\xi})\mathbf{T})^{-1} \left(\mathbf{T}_{obs}\mathbf{T}^{-1} \right)^{-1} \right] \right) \\
&= \frac{\partial}{\partial \Delta\boldsymbol{\xi}} (\operatorname{Ln}[\underbrace{\mathbf{T}_{obs}\mathbf{T}^{-1} \operatorname{Exp}(-\Delta\boldsymbol{\xi}) \left(\mathbf{T}_{obs}\mathbf{T}^{-1} \right)^{-1}}_{\text{Adjoint (30)}}]) \\
&= \frac{\partial}{\partial \Delta\boldsymbol{\xi}} \left(\operatorname{Ln} \left[\operatorname{Exp} \left(Adj_{\{\mathbf{T}_{obs}\mathbf{T}^{-1}\}}(-\Delta\boldsymbol{\xi}) \right) \right] \right) \\
&= -Adj_{\{\mathbf{T}_{obs}\mathbf{T}^{-1}\}}.
\end{aligned}
\tag{2.56}
$$

This gives us a compact result for the derivative, expressed as a 6×6 matrix.

2.7.5 Example: Two RBT

Consider the function $g(\mathbf{T}_o, \mathbf{T}_t) : \mathbf{SE}(3) \times \mathbf{SE}(3) \to \mathbf{SE}(3)$ be equal to $\boldsymbol{g}(\mathbf{T}_o, \mathbf{T}_t) = \mathbf{T}_o\mathbf{T}_{obs}\mathbf{T}_t^{-1}$. This function quantifies how well our observation aligns with this pair of input poses, such as an odometry pose that matches two consecutive poses within our trajectory. We can then compute the derivative with respect to the origin pose $\mathbf{T}_o$ as

$$
\begin{aligned}
\frac{\partial \boldsymbol{g}}{\partial \mathbf{T}_o} &= \frac{\partial}{\partial \Delta\boldsymbol{\xi}} \left(\operatorname{Ln} \left[\operatorname{Exp}(\Delta\boldsymbol{\xi})\mathbf{T}_o\mathbf{T}_{obs}\mathbf{T}_t^{-1} \left(\mathbf{T}_o\mathbf{T}_{obs}\mathbf{T}_t^{-1} \right)^{-1} \right] \right) \\
&= \frac{\partial}{\partial \Delta\boldsymbol{\xi}} (\operatorname{Ln}[\operatorname{Exp}(\Delta\boldsymbol{\xi})]) = \mathbf{I}.
\end{aligned}
\tag{2.57}
$$

Chapter 3

Jacobian Approximation for Direct Optimization of a Trajectory of Interpolated Poses on SE(3)

In this chapter, we continue our investigation into the challenging scenario of time-continuous state estimation via optimization on manifold concerning time-continuous trajectory representation, utilizing direct linear interpolation on $\mathbf{SE}(3)$. Our approach focuses on a novel Jacobian approximation for a sequence of linearly interpolated poses on $\mathbf{SE}(3)$. This chapter presents a derivation of the proposed approximated Jacobian using retraction mapping, along with an approximation based on the commutativity property of infinitesimal group elements.

We provide extensive evaluations across three distinct optimization problems. In the synthetic point cloud alignment problem, our proposed Jacobian is compared with a numerical alternative. For the synthetic pose graph optimization problem, the proposed Jacobian approximation enables a reduction in state dimensions by a factor of seven, while maintaining a comparable magnitude of resulting error when contrasted with the full discrete-time trajectory. Finally, we demonstrate the validity of our approach in a time-continuous framework for addressing real-world LIDAR odometry challenges.

3.1 Introduction

Pose estimation problems are often formulated using a discrete-time representation of a sequence of 3D poses over time – a trajectory. It is the default representation due to a consolidated theory on state estimation and clear interpretability of the results Dellaert and Kaess [2006],Cunningham et al. [2013]. In addition, many trajectory estimation approaches for robot-sensor systems successfully use a discrete-time formulation since it allows keeping the state size tractable Kaess et al. [2008], Wang et al. [2018]. However, this is unattainable for setups that include different high-rate or asynchronous sensors Huai et al. [2021], Gohard et al. [2019], making discrete-time formulation non-suitable for those scenarios without additional assumptions.

The straightforward solution is to use a time-continuous trajectory representation that enables the addition of information from a high-rate sensor output, preventing to estimate a discrete state for each observation which would produce a high number of state variables to be estimated. Time-continuous trajectories are also capable of fusing multiple asynchronous sensors of different high rates in a natural way. In addition, a time-continuous trajectory representation should allow for a direct calculation of the Jacobian since most modern state estimation techniques are built around optimization.

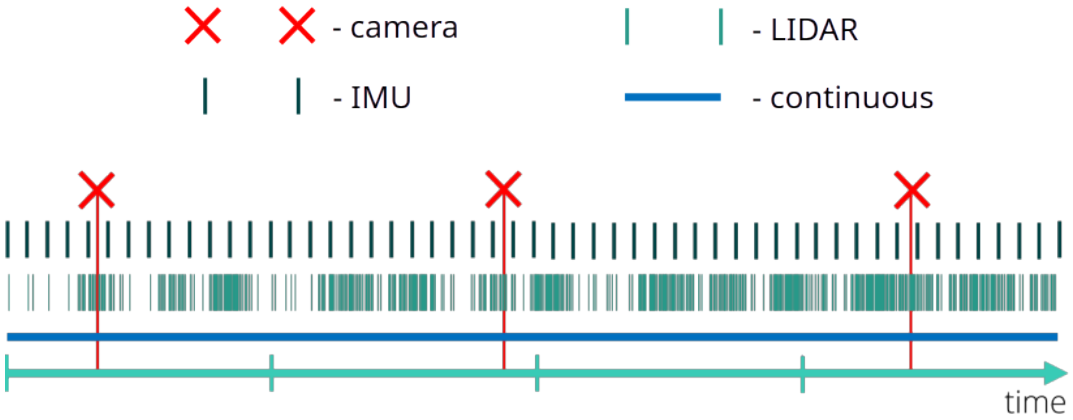


Figure 3-1: Time continuous sensor model compared to discrete

Time-continuity of the trajectory can be ensured using different methods, the most convenient being the Gaussian process used in Dong et al. [2017], Kapushev et al. [2021], B-splines interpolation used for split rotation and translation interpolation

as in Huai et al. [2021], Gohard et al. [2019] or for direct interpolation on Special Euclidean Group $\mathbf{SE}(3)$ as in Tirado and Civera [2022], Mueggler et al. [2018], and finally, linear interpolation of split pose representation as in Dellenbach et al. [2021], Yuan et al. [2022] or for direct interpolation on $\mathbf{SE}(3)$ as in Park et al. [2018],Ceriani et al. [2015], Park et al. [2022].

A comprehensive comparison of different motion interpolation methods is provided in Haarbach et al. [2018]. Despite the $\mathcal{C}^d$ continuity of the B-spline interpolated trajectory, it has a higher computational cost than the linear form and may over-smooth the trajectory by vanishing its high-frequency components. Alternatively, the linear interpolation keeps the raw high-frequency trajectory components and has a much lower computational cost. It is continuous but, at the same time, non smooth on the boundary of the control poses, which is its main disadvantage.

Concerning the optimization, it is a valid question to wonder about the efficiency and correctness of the Jacobian calculation for the corresponding time-continuous representations. Unfortunately, many of the above-mentioned methods keep this issue out of scope by using Jacobians obtained with auto-differentiation functionality of popular non-linear least square problem solvers. Nevertheless, having the analytical form is essential for real-time applications with limited computational resources.

The Jacobians for linear interpolation of split pose representation are introduced in Yuan et al. [2022]. It presents a Jacobian form related to spherical linear interpolation for rotation and linear interpolation for translation and solves the continuous-time LiDAR-Inertial Odometry problem with Sweep Reconstruction. The analytical Jacobian with respect to control poses of direct B-splines interpolation on $\mathbf{SE}(3)$ is derived in Tirado and Civera [2022]. Authors approximate it by applying Baker–Campbell–Hausdorff (BCH) formula to 3D poses, which is previously derived in Barfoot's book Barfoot [2017] for the direct linear pose interpolation on $\mathbf{SE}(3)$.

The contributions described in this chapter are as follows. We derive an analytical form of the Jacobian for linearly interpolated poses on $\mathbf{SE}(3)$ using an approximation by the commutativity property of infinitesimal group elements instead of the BCH formula approximation. We evaluate empirically our approach on several synthetic and real-world datasets and problems, provide a comparison with the formulation

proposed in Barfoot [2017] and show its consistency in robustly achieving accurate results in state estimation of 3D trajectories via optimization on the manifold.

The chapter is organized as follows. Section 3.2 introduces background on Special Euclidean Group $\mathbf{SE}(3)$ and the retraction operation. Section 3.3 describes interpolation in the manifold and the derivation of our proposed Jacobian approximation. Section 3.4 presents the evaluation results for two synthetic problems, the multi-registration of point clouds and pose SLAM, and one real problem from Geiger et al. [2012] LiDAR data. Finally, Section 3.5 concludes the chapter.

3.2 Background

Following the definitions given in the Chapter 2, we operate with poses or Rigid Body Transformations (RBT) that are elements of Special Euclidean group $\mathbf{SE}(3)$ that defined as

$$\mathbf{SE}(3) = \left\{ \mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix} \mid \mathbf{R} \in \mathbf{SO}(3)\,,\, \mathbf{t} \in \mathbb{R}^3 \right\}. \tag{3.1}$$

Rigid body transformations are elements of the matrix group $\mathbf{SE}(3) \subset \mathbb{R}^{4\times4}$, representing a manifold, a lower-dimensional structure. As such, a connection exists between elements of the $\mathbf{SE}(3)$ group with a vector space $\mathbb{R}^6$. We will make use of the definition of a *retraction* $R_T(\boldsymbol{\xi})$, as defined in Absil et al. [2009], which is a smooth mapping from the tangent space around the $\mathbf{T} \in \mathbf{SE}(3)$ element to the manifold defined as

$$R_T(\boldsymbol{\xi}) : \mathbb{R}^6 \to \mathbf{SE}(3)\,. \tag{3.2}$$

Two conditions must be satisfied: i) $R_T(0_T) = \mathbf{T}$ and ii) "local rigidity"[1].

We refer the reader to Chapter 4 in the Absil et al. book Absil et al. [2009] for a complete definition of the general concept of retraction.

Accordingly, the derivative around the element $\mathbf{T}$ becomes a well-defined operation

[1]Conceptually, a retraction R at T, denoted by R_T , is a mapping from the tangent space $\mathcal{T}_T\mathcal{M}$ at the element T to the manifold $\mathcal{M}$ with a *local rigidity* condition that preserves gradients at T. More formally, the local rigidity is defined as $DR_{\mathcal{T}_T}(0_T) = Id_{\mathcal{T}_T\mathcal{M}}$, where Id indicates the identity

now that the retraction allows us to operate in a vector space, where a directional derivative can be expressed as

$$\left.\frac{\partial R_T(\boldsymbol{\xi})}{\partial \boldsymbol{\xi}}\right|_{\boldsymbol{\xi}=0}. \tag{3.3}$$

Note that this directional derivative is indeed a mapping, defined at the zero element.

There are multiple options for choosing a retraction. In our case, the exponential map is the most natural retraction choice: its definition is highly related to the concept of perturbations over $\mathbf{SE}(3)$, which is the main topic since we are interested in calculating the derivatives of linear interpolated 3D poses.

The exponential map defines the retractions R_T in the following way

$$R_T(\boldsymbol{\xi}) = \mathrm{Exp}(\boldsymbol{\xi})\mathbf{T}, \tag{3.4}$$

where we have chosen a left-hand side emplacement of the exponent map as our default convention. A right-hand side convention would require re-doing all further derivations, but the procedure would be identical to what is expanded in the following sections.

3.3 Interpolation in the manifold

Smooth manifolds allow us to import all the tools from calculus and real analysis into manifolds by following simple rules.

We follow Barfoot [2017] and define a linear interpolation of poses $T(\mathbf{T}_1, \mathbf{T}_2, \tau) : \mathbf{SE}(3) \times \mathbf{SE}(3) \times [0, 1] \rightarrow \mathbf{SE}(3)$ as

$$T(\mathbf{T}_1, \mathbf{T}_2, \tau) = (\mathbf{T}_2\mathbf{T}_1^{-1})^{\tau}\mathbf{T}_1. \tag{3.5}$$

This form provides a direct interpolation in $\mathbf{SE}(3)$ and in practice is computed as $\mathrm{Exp}(\tau\,\mathrm{Ln}(\mathbf{T}_2\mathbf{T}_1^{-1})\mathbf{T}_1$. However, in order for us to differentiate for each pose, we require to define a new retraction as the Cartesian product of two manifold elements. In particular, when substituting the retraction (3.4), we obtain a map of the local coordinates of each of the poses as

$$
\begin{aligned}
R_{T_1T_2}(\boldsymbol{\xi}_1, \boldsymbol{\xi}_2) &= T\big(R_{T_1}(\boldsymbol{\xi}_1), R_{T_2}(\boldsymbol{\xi}_2), \tau\big) \\
&= \big(\operatorname{Exp}(\boldsymbol{\xi}_2)\mathbf{T}_2\mathbf{T}_1^{-1}\operatorname{Exp}(-\boldsymbol{\xi}_1)\big)^{\tau}\operatorname{Exp}(\boldsymbol{\xi}_1)\mathbf{T}_1,
\end{aligned} \tag{3.6}
$$

where we have used the property $\operatorname{Exp}(\boldsymbol{\xi})^{-1} = \operatorname{Exp}(-\boldsymbol{\xi})$.

We want to compare the tangent space around both transformations, such that they are equal and to find out the relation of the tangent spaces $\boldsymbol{\xi}_1$, $\boldsymbol{\xi}_2$ and the new joint variable $\boldsymbol{\xi} = [\boldsymbol{\xi}_1, \boldsymbol{\xi}_2]$ composed of the two poses as expressed in the retraction (3.6).

We will expand each of the terms, evaluated at the null element of the tangent vector $\boldsymbol{\xi} = 0$ deriving

$$
\begin{aligned}
R_{T_1T_2}(0, \boldsymbol{\xi}_2)\Big|_{\boldsymbol{\xi}_2=0} &= \big(\operatorname{Exp}(\boldsymbol{\xi}_2)\mathbf{T}_2\mathbf{T}_1^{-1}\big)^{\tau}\mathbf{T}_1\Big|_{\boldsymbol{\xi}_2=0} \\
&\approx \operatorname{Exp}(\tau\boldsymbol{\xi}_2)\big(\mathbf{T}_2\mathbf{T}_1^{-1}\big)^{\tau}\mathbf{T}_1\Big|_{\boldsymbol{\xi}_2=0}.
\end{aligned} \tag{3.7}
$$

We have made use of the property $\operatorname{Exp}(\boldsymbol{\xi})^{\alpha} = \operatorname{Exp}(\alpha\boldsymbol{\xi})$ and the fact that this expression is *evaluated* at the zero element $\boldsymbol{\xi}_2 = 0$. Thereby, the exponent equals the identity in the limit, and the matrix multiplication becomes commutative under this approximation. Accordingly, $\operatorname{Exp}(\boldsymbol{\xi}_2)$ can be moved away from the parenthesis if we consider that this expression is multiplied τ times.

We will take this relaxation to expand the infinitesimal exponent and obtain an analytical solution that is very close to the real analytical gradient, as discussed in the evaluation section. In general, the multiplication of matrix elements infinitesimally close to the identity is a non-commutative operation.

Similarly, the same derivation can be done for $\boldsymbol{\xi}_1$

$$\begin{aligned} R_{T_1T_2}(\boldsymbol{\xi}_1, 0) &= \left(\mathbf{T}_2\mathbf{T}_1^{-1}\operatorname{Exp}(-\boldsymbol{\xi}_1)\right)^{\tau}\operatorname{Exp}(\boldsymbol{\xi}_1)\mathbf{T}_1 \\ &\approx \left(\mathbf{T}_2\mathbf{T}_1^{-1})\right)^{\tau}\operatorname{Exp}(-\boldsymbol{\xi}_1)^{\tau}\operatorname{Exp}(\boldsymbol{\xi}_1)\mathbf{T}_1 \\ &= \Delta\mathbf{T}\operatorname{Exp}((1-\tau)\boldsymbol{\xi}_1)\mathbf{T}_1 \\ &= \operatorname{Exp}\left((1-\tau)Adj_{\Delta\mathbf{T}}\boldsymbol{\xi}_1\right)\cdot\mathbf{T}\Big|_{\boldsymbol{\xi}_1=0}, \end{aligned} \tag{3.8}$$

where $\Delta\mathbf{T} = (\mathbf{T}_2\mathbf{T}_1^{-1})^{\tau}$ and the following property on exponent multiplication holds for the same axis ξ such that $\operatorname{Exp}(\alpha\xi)\operatorname{Exp}(\beta\xi) = \operatorname{Exp}((\alpha+\beta)\xi)$.

Suppose we arrange (3.7) and (3.8) under the condition that both expressions will be evaluated at zero. In that case, we obtain the following compact linear relation that is only valid for calculating derivatives as

$$\boldsymbol{\xi}\Big|_{\boldsymbol{\xi}=0} = (1-\tau)Adj_{\Delta\mathbf{T}}\boldsymbol{\xi}_1 + \tau\boldsymbol{\xi}_2\Big|_{\boldsymbol{\xi}_1=0,\,\boldsymbol{\xi}_2=0}. \tag{3.9}$$

Now, suppose we have a cost function that depends on a linearly interpolated pose on $\mathbf{SE}(3)$. In that case, we can find its derivative with respect to reference poses $\mathbf{T}_1$ and $\mathbf{T}_2$, as defined in (Sec. 2.7.3) in addition to using the retraction mapping in (3.6) and the chain rule Absil et al. [2009] as

$$\frac{\partial h(\mathbf{T})}{\partial\mathbf{T}}\Big|_{\mathbf{T}} = \frac{\partial h_T(\boldsymbol{\xi})}{\partial\boldsymbol{\xi}}\Big|_{\boldsymbol{\xi}=0}, \tag{3.10}$$

$$\begin{aligned} \frac{\partial h_T(\boldsymbol{\xi}(\boldsymbol{\xi}_1,\boldsymbol{\xi}_2))}{\partial\boldsymbol{\xi}_{1,2}}\Big|_{\boldsymbol{\xi}=0} &= \frac{\partial h_T(\boldsymbol{\xi})}{\partial\boldsymbol{\xi}}\frac{\partial\boldsymbol{\xi}}{\partial\boldsymbol{\xi}_{1,2}}\Big|_{\boldsymbol{\xi}=0} \\ &\approx \frac{\partial h_T(\boldsymbol{\xi})}{\partial\boldsymbol{\xi}}\Big|_{\boldsymbol{\xi}=0}\begin{bmatrix}(1-\tau)Adj_{\Delta\mathbf{T}} & \tau\mathbf{I}\end{bmatrix}. \end{aligned} \tag{3.11}$$

Alternatively, we can state it as

$$\frac{\partial T(\mathbf{T}_1,\mathbf{T}_2,\tau)}{\partial\mathbf{T}_1}\Big|_{\boldsymbol{\xi}=0} \approx (1-\tau)Adj_{\Delta\mathbf{T}}, \tag{3.12}$$

$$\left.\frac{\partial T(\mathbf{T}_1, \mathbf{T}_2, \tau)}{\partial \mathbf{T}_2}\right|_{\boldsymbol{\xi}=0} \approx \tau \mathbf{I}. \tag{3.13}$$

The results (3.11), (3.12) and (3.13) are our proposed Jacobian approximation that we imply to use for optimization of trajectories of interpolated poses on $\mathbf{SE}(3)$ directly on manifold. Analyzing it more closely, we can see that the resulting Jacobian form is in direct ratio with the interpolation time τ in the way that gradually moving from $\mathbf{T}_1$ to the $\mathbf{T}_2$ is coupled with the increase or decrease of the corresponding parts of the resulting Jacobian. Besides that, one can see that computation of this Jacobian is relatively lightweight since it is just a single matrix-scalar multiplication.

In the following sections, we evaluate it in three problems showing its consistency. For that, we first evaluate it alongside another form that is introduced in Barfoot [2017] by approximation with Baker–Campbell–Hausdorff (BCH) formula resulting in

$$\begin{aligned} \delta\boldsymbol{\xi} &= (\mathbf{1} - \mathcal{A}(\tau, \boldsymbol{\xi}_{21}))\, \delta\boldsymbol{\xi}_1 + \mathcal{A}(\tau, \boldsymbol{\xi}_{21})\, \delta\boldsymbol{\xi}_2, \\ A(\tau, \boldsymbol{\xi}) &= \tau \mathcal{J}(\tau\boldsymbol{\xi}) \mathcal{J}(\boldsymbol{\xi})^{-1}, \end{aligned} \tag{3.14}$$

where $\boldsymbol{\xi}_{21}$ defined as $R_T(\boldsymbol{\xi}_{21}) = \mathrm{Exp}(\boldsymbol{\xi}_{21})\mathbf{T}_2\mathbf{T}_1^{-1}$, and $\mathcal{J}$ is a left Jacobian of $\mathbf{SE}(3)$ (see Barfoot [2017]).

3.4 Evaluation

The main purpose of the experiments is to show the applicability of the proposed approximated gradient form to be used in various problems that can apply time-continuous trajectory representation and optimization.

For that, we first conduct an experiment on time-continuous point-cloud alignment problem (multi-view registration) with synthetic randomly generated data. In this task we compare results achieved using the proposed gradient form (3.11), Barfoot's Barfoot [2017] formulation (3.14) and the numerically estimated gradient obtained using the finite difference method. We imply numerical gradient as the most accurate one to compare and use its results as a baseline.

In the second experiment we investigate how the proposed gradient formulation

is suitable for solving time-continuous Pose Simultaneous Localization and Mapping (SLAM) problem on a synthetic benchmark - Sphere dataset Kümmerle et al. [2011].

And finally, we qualitatively describe the applicability of the proposed gradient form for real data implementing it inside the Continuous-Time Iterative Closest Point (CT-ICP) Dellenbach et al. [2021] algorithm and achieving trajectory consistency for short subsamples of raw uncorrected KITTI dataset Geiger et al. [2012] scenes. We obtain results for all three experiments using Intel i7-9750H CPU.

3.4.1 Synthetic Time-Continuous Point Cloud Alignment

We start with testing the applicability of the proposed gradient formulation (3.11) for a multi-view point cloud alignment problem using randomly generated synthetic data.

Problem description

Point cloud alignment task implies estimating a relative transform $\mathbf{T}_{1\to 2}$ between two poses $\{\mathbf{T}_1, \mathbf{T}_2\} \in \mathbf{SE}(3)$ from which we observe a shared cloud of points.

In our case, we randomly generate point cloud $\mathcal{P}\{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_N | \mathbf{p}_i \in \mathbb{R}^3\}$ of size N. After that, we generate two random poses $\{\mathbf{T}_1, \mathbf{T}_2\} \in \mathbf{SE}(3)$. We treat these poses as the interpolation interval's initial and final ends. We uniformly split this interval with K intermediate timestamps $\tau_k \in [0, 1]$. Using these timestamps, we interpolate initial and final poses, getting corresponding intermediate poses $\mathcal{T}\{\mathbf{T}_{\tau_1}, \mathbf{T}_{\tau_2}, ..., \mathbf{T}_{\tau_K} | \mathbf{T}_{\tau_k} = T(\mathbf{T}_1, \mathbf{T}_2, \tau_k)\}$.

For each interpolated pose we generate corresponding point cloud observations $\mathbf{p}_i^{\tau_k}$ by adding noise $\alpha_i^{\tau_k}$ randomly sampled from normal distribution $\mathcal{N}(0, \mathbf{\Sigma}_i^{\tau_k})$ to each point obtaining K different sets of points $\mathcal{P}_k$. Here $\mathbf{\Sigma}_i^{\tau_k} = \boldsymbol{I}_3 \alpha_i^{\tau_k}$ where $\alpha_i^{\tau_k} \in \mathbb{R}^3$ sampled from one more distribution $\alpha_i^{\tau_k} \backsim \mathcal{N}(0, \sigma \mathbf{I}_3)$. By that, we can specify the noise magnitude of the points observations by varying parameter σ and thus investigate the performance of the gradient forms during optimization with different challenging conditions.

Furthermore, the last important parameter related to our synthetic data generation approach is a scale parameter s. Changing this parameter, we can scale by s

factor translation components $\mathbf{t} \in \mathbb{R}^3$ of all poses and points, keeping the rotation parts $\mathbf{R} \in \mathbf{SO}(3)$ unchanged. One can think about this as stretching or shrinking the scene without changing its internal structure.

Having all these data generated, we seek to find optimal initial and final poses $\mathbf{T}_1$ and $\mathbf{T}_2$ to achieve a points observation consistency for all poses in $\mathcal{T}$. For that we minimize the following cost function $h(\mathbf{T}_1, \mathbf{T}_2)$ with respect to initial or final pose as

$$h(\mathbf{T}_1, \mathbf{T}_2) = \frac{1}{2} \sum_{i=0}^{N-1} \sum_{\substack{\tau_j \\ \tau_j \neq \tau_k}} \sum_{\tau_k} \left\| \mathbf{T}_\tau(\mathbf{T}_1, \mathbf{T}_2, \tau_j) \mathbf{p}_i^{\tau_j} - \mathbf{T}_\tau(\mathbf{T}_1, \mathbf{T}_2, \tau_k) \mathbf{p}_i^{\tau_k} \right\|^2_{\mathbf{\Sigma}_i^{\tau_j} + \mathbf{\Sigma}_i^{\tau_k}}. \quad (3.15)$$

This cost function implies pair-wise all-vs-all comparison of i-th point observation for all poses in $\mathcal{T}$.

Denoting the expression under the norm as a residual $r_i(\mathbf{T}_1, \mathbf{T}_2, \tau_j, \tau_k)$ that describes an observation point relation between some two interpolated poses and introducing $\mathbf{\Sigma}_i = \mathbf{\Sigma}_i^{\tau_j} + \mathbf{\Sigma}_i^{\tau_k}$ we rewrite (3.15) in a shorter form as

$$h(\mathbf{T}_1, \mathbf{T}_2) = \frac{1}{2} \sum_{i=0}^{N-1} \sum_{\substack{\tau_j \\ \tau_j \neq \tau_k}} \sum_{\tau_k} \left\| r_i(\mathbf{T}_1, \mathbf{T}_2, \tau_j, \tau_k) \right\|^2_{\mathbf{\Sigma}_i}. \quad (3.16)$$

In order to separately testify parts of the gradient related to the initial or final pose, we run two experiments. In the first case, we optimize only initial pose $\mathbf{T}_1$, fixing final pose $\mathbf{T}_2$ as an identity transform. In the second case, we do the vice-versa and optimize $\mathbf{T}_2$ keeping $\mathbf{T}_1$ fixed. As a result, our goal is to solve the following optimization problem

$$\min_{\substack{\mathbf{T}_1, \mathbf{T}_2 = \boldsymbol{I} \\ \text{or} \\ \mathbf{T}_2, \mathbf{T}_1 = \boldsymbol{I}}} h(\mathbf{T}_1, \mathbf{T}_2). \quad (3.17)$$

Using the following gradient expression

$$\nabla_{\mathbf{T}_{1,2}} h = \sum_{i=0}^{N-1} \sum_{\substack{\tau_j \\ \tau_j \neq \tau_k}} \sum_{\tau_k} r_i(\mathbf{T}_{1,2})^\top \mathbf{\Sigma}_i^{-1} \frac{\partial r_i}{\partial \mathbf{T}_{1,2}}, \quad (3.18)$$

where $\mathbf{T}_{1,2} \in \{\mathbf{T}_1, \mathbf{T}_2\}$ and

$$\frac{\partial r_i}{\partial \mathbf{T}_{1,2}} = \left[\frac{\partial r_i}{\partial \mathbf{T}_{\tau_j}} \frac{\partial \mathbf{T}_{\tau_j}}{\partial \mathbf{T}_{1,2}} - \frac{\partial r_i}{\partial \mathbf{T}_{\tau_k}} \frac{\partial \mathbf{T}_{\tau_k}}{\partial \mathbf{T}_{1,2}} \right] \tag{3.19}$$

is a gradient of the residual r_i with respect to optimization pose and $\frac{\partial \mathbf{T}_\tau}{\partial \mathbf{T}_{1,2}}$ - is a gradient of the interpolation pose computed using our proposed method or Barfoot's form.

Experiment results

In our experiments for that task, we test different setups of the problem by checking different combinations of the parameters defined in the synthetic data generation description above. To be more precise, we solve this task using the Gauss-Newton method for all combinations of the following parameters: point cloud size $N \in [20, 40, 80, 100]$, number of interpolation timestamps $K \in [5, 20, 40, 80, 100]$, scale factors $s \in [1, 10, 100]$ and noise magnitude $\sigma = 0.01$.

We optimize these tasks using the proposed gradient, Barfoot's BCH approximation, and numerical form. We initialize optimized $\mathbf{T}_{1,2}$ as identity transform and compare results with a ground truth $\mathbf{T}_{gt}$ using a distance metric which we denote as a norm of the Lie algebra coordinates vector as

$$d = \left\| Ln(\mathbf{T}_{1,2}\mathbf{T}_{gt}^{-1}) \right\|. \tag{3.20}$$

These experiments allow us to comprehensively test each Jacobian formulation performance and limitations. We provide main insights on results in Figs.3-2-Fig.3-6. The left graphs - results for the initial pose $\mathbf{T}_1$ optimization (first case), and the right ones - results for the final pose $\mathbf{T}_2$ (second case). In Fig.3-2, we show that with an increase in the number of the interpolated states K used, the accuracy of the final result also increases for both cases. The same improvement also occurs with the increase of the number of points in a point cloud N as shown in Fig.3-3. Direct comparison of Jacobian magnitudes presented in Fig.3-4 shows the close similarity of all three methods.

Summing the final results for all experiments, we see that all three approaches

show similar results and achieve a feasible solution for both cases. Barfoot's method reproduces the numerical results up to the numerical resolution. Our proposed method closely follows the others providing similar or better final optimization results. Although our method does not exactly match the numerical Jacobian, it still results in almost similar number of optimization steps to converge as shown in Fig.3-5 and, more importantly, being computationally lighter, it requires noticeably less time to obtain the result. As shown in Fig.3-6, our method reaches the solution ~ 2 times faster than the BCH form and ~ 3 times faster than the numerical solution. More precisely, our method requires 0.02 ms to compute $\frac{\partial \mathbf{T}_\tau}{\partial \mathbf{T}_{1,2}}$ in (3.19), while Barfoot's form - 0.16 ms.

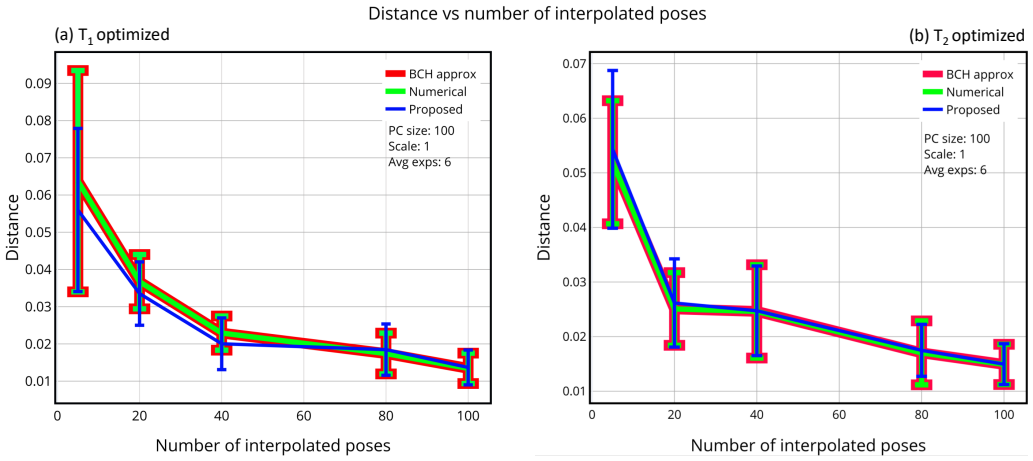


Figure 3-2: Optimization results vs different numbers of interpolated poses K

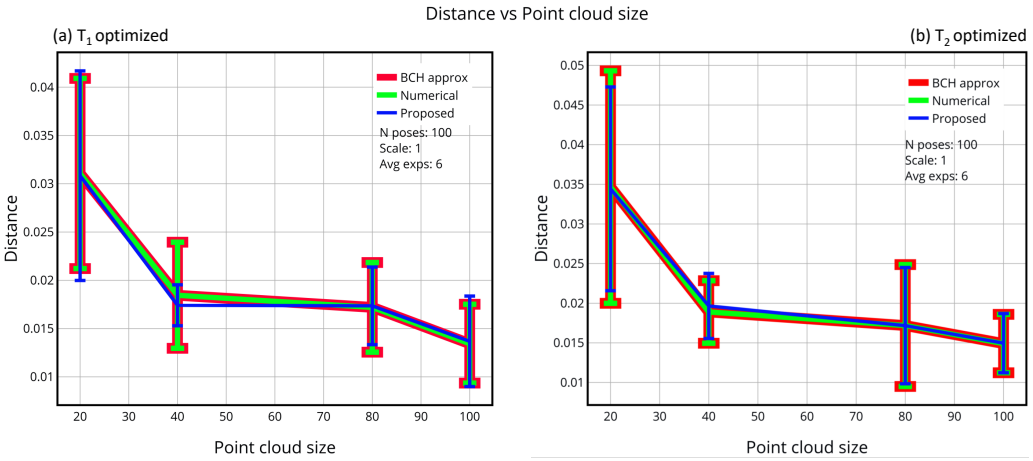


Figure 3-3: Optimization results vs different numbers N of points in point cloud

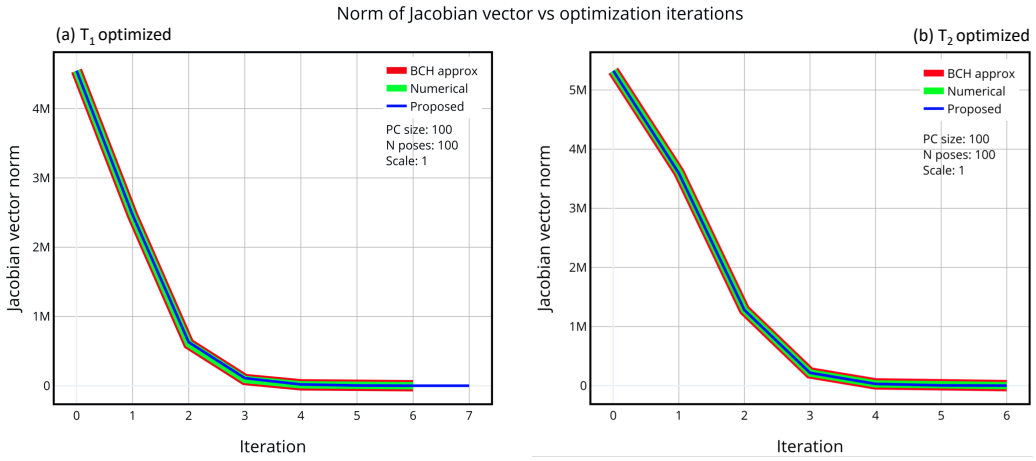


Figure 3-4: Magnitudes of Jacobians

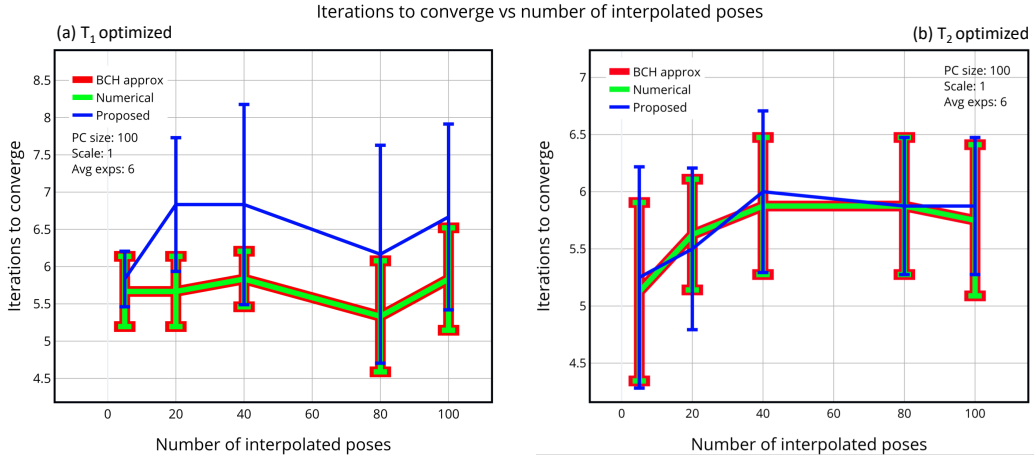


Figure 3-5: The number of iterations to converge vs number of interpolated poses K

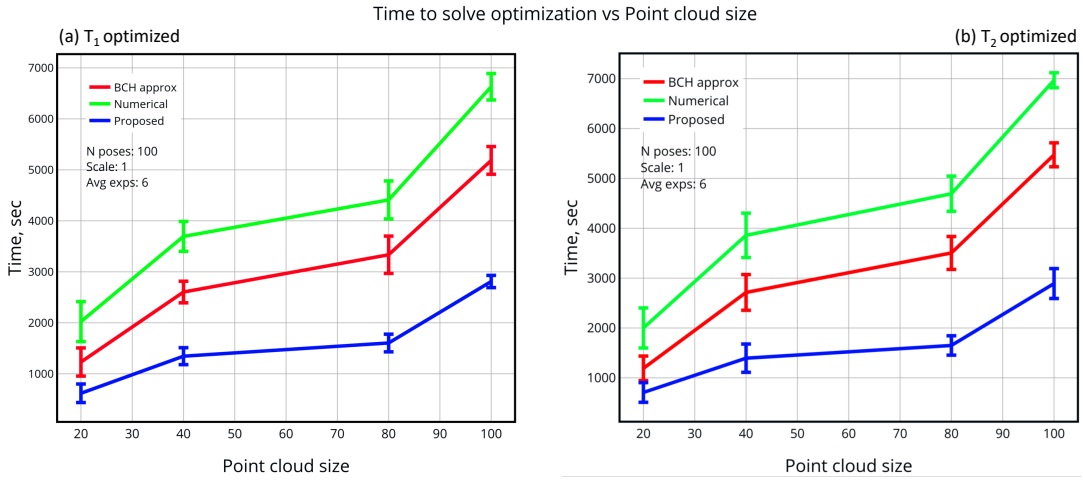


Figure 3-6: Time to converge vs number N of points in point cloud

This evaluation experiment shows that our gradient approximation for the linearly interpolated rigid body transform (RBT) has no usage limitations and allows us to expect it to be sufficient for other problems requiring on manifold interpolated pose optimization.

3.4.2 Synthetic Pose-SLAM

We proceed by testing the applicability of the proposed Jacobian approximation for the Pose SLAM problem.

Problem description

The Pose SLAM is a nonlinear pose graph optimization problem that estimates N robot poses $\mathbf{T}_i \in \mathbf{SE}(3)$ using M relative pose observations $\mathbf{T}_{ij}^{obs} \in \mathbf{SE}(3)$. This problem can be seen as a directed graph: robot poses $\mathbf{T}_i$ are graph nodes that we want to estimate, while relative pose observations are graph edges $\mathcal{E}_{ij}$ that encodes a relative pose transformation measurement between two poses $\mathbf{T}_i$ and $\mathbf{T}_j$.

The classic Pose SLAM graph optimization estimates robot poses by solving the following optimization problem

$$\min_{\{\mathbf{T}_i\} \in \mathbf{SE}(3)} \sum_{\{\mathcal{E}_{ij}\}} g(\mathbf{T}_i, \mathbf{T}_j, \mathbf{T}_{ij}^{obs}), \tag{3.21}$$

$$g(\mathbf{T}_i, \mathbf{T}_j, \mathbf{T}_{ij}^{obs}) = \mathbf{T}_i \mathbf{T}_{ij}^{obs} \mathbf{T}_j^{-1}. \tag{3.22}$$

Function $g(\mathbf{T}_i, \mathbf{T}_j, \mathbf{T}_{ij}^{obs})$ measures how well our observation matches pair (i, j) of input nodes and they are "closed" chains of transformations, ideally equaling the identity transformation.

We use this approach as a baseline for comparison. In order to check the performance of our Jacobian of interpolated RBT, we modify the classic Pose SLAM problem by decimating the trajectory in the factor graph using linear interpolation among them. The decimation factor is expressed by the δ parameter. Regarding the graphical model, the number of nodes is reduced δ times and the number of factors (observations) remains unaltered.

With that, it is possible to achieve almost similar final results with less computations per optimization step by solving a smaller optimization task. We further refer to those nodes we keep as *base*-nodes and those we replace with interpolation as *inter*-nodes.

Accordingly, we can modify our Pose SLAM optimization problem (3.21) and cost function (3.22) as

$$\min_{\{\hat{\mathbf{T}}_i\}\in\mathbf{SE}(3)} \sum_{\{\mathcal{E}_{ij}\}} \hat{g}(\mathbf{T}_{\tau_i}, \mathbf{T}_{\tau_j}, \mathbf{T}_{ij}^{obs}), \tag{3.23}$$

$$\hat{g}(\mathbf{T}_{\tau_i}, \mathbf{T}_{\tau_j}, \mathbf{T}_{ij}^{obs}) = \mathbf{T}_{\tau_i}\mathbf{T}_{ij}^{obs}\mathbf{T}_{\tau_j}^{-1}, \tag{3.24}$$

$$\mathbf{T}_{\tau_i}(\hat{\mathbf{T}}_a, \hat{\mathbf{T}}_b, \tau_i) = (\hat{\mathbf{T}}_b\hat{\mathbf{T}}_a^{-1})^{\tau_i}\hat{\mathbf{T}}_a, \tag{3.25}$$

where $\hat{g}$ is our new interpolated cost function, $\{\hat{\mathbf{T}}_i\}$ - our novel sparse set of *base*-nodes to estimate, $\mathbf{T}_{\tau_i}$ - pose of the i-th *inter*-node, its corresponding previous $\hat{\mathbf{T}}_a$ and next $\hat{\mathbf{T}}_b$ *base*-nodes used for interpolation with timestamp τ_i. Following the derivations given in the Section 2.8 we derive the chain rule gradients and experimentally evaluate the performance in the next section.

Table 3.1: RPE before and after optimization of pose graphs with different sparsity factors

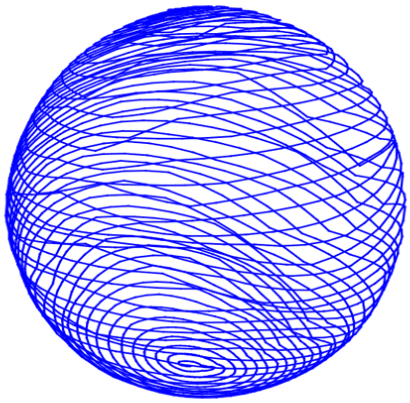
Sparsity factor	Num factors	Num nodes	Time per opt step, sec	Base nodes only			Full trajectory			
				RPE Before	RPE Interp. posegraph	RPE Classic	Interpolated posegraph		Classic	
							RPE Before	RPE After	RPE Before	RPE After
x1 (classic)	9788	2500	0.51	73.13	—	4.13	—	—	73.13	4.13
x4		625	0.19	73.09	10.71	4.13	84.94	11.46		
x8		313	0.16	73.02	12.35	4.16	84.95	14.22		
x16		157	0.11	72.96	63.71	4.20	84.96	73.64		
x32		79	0.07	73.00	85.35	4.18	85.73	85.16		

Experiment results

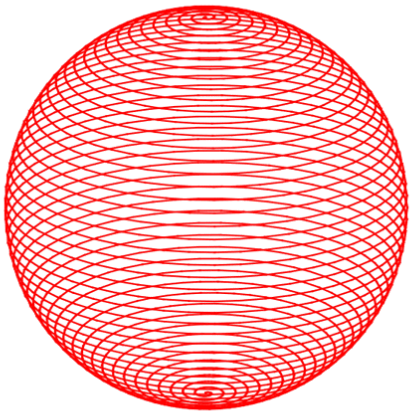
We use the sphere benchmark Kümmerle et al. [2011] as a data source. It consists of 2500 poses and 9788 observations. We first solve the classic Pose SLAM problem (3.21) and use it as a baseline for comparison. After that, we solve Interpolated Pose SLAM (3.23) using our approximated Jacobian form, comparing its results for different sparsity factors with a baseline.

We use the mrob C++ library as a framework for optimization mrob. We provide visualization for the Interpolated Pose SLAM for sparsity factor $\delta = 4$ at Fig. 3-7

(a) Before optimization



(b) Interpolated Pose SLAM results



(c) Classic Pose SLAM results

Figure 3-7: Visual representation of the full trajectory Interpolated Pose SLAM results with sparsity factor $\delta = 4$ on Sphere data. Here (a) - depicts an initial state of the graph, (b) - Interpolated Pose SLAM results, (c) - Classic Pose SLAM results

where (a) depicts an initial state of *base*-nodes and (b) represents the same nodes after optimization. One can see the visible improvement obtained by keeping only 625 of the original 2500 poses of the original dataset.

Tab 3.1 provides more comprehensive results and comparison with a classic approach for different sparsity factors. We use Relative Pose Error (RPE) as a metric for comparison. We define it as an average pair-wise all-vs-all distance (3.20) between relative transforms of poses inside the estimated trajectory and the ground

truth one. Base nodes only comparison performed using only nodes that we use as *base*-nodes in our interpolated pose graph. Full trajectory comparison implies that we compare whole trajectories of 2500 original nodes. We obtain missing *inter*-nodes for interpolated pose graph using linear interpolation with corresponding *base*-nodes.

As we see from Tab 3.1, our Jacobian form ensures a feasible solution with sparsity factors of 4 and 8, decreasing the number of optimization state variables to 625 and 313 nodes out of the initial 2500. Further decimation with sparsity factors of 16 and 32 leads to unusable results due to the far distance between interpolation base nodes and the inability to restore the trajectory's circularity using linear interpolation in such conditions.

3.4.3 Time-Continuous LIDAR Odometry

In the last experiment, we perform a qualitative performance estimation of our Jacobian formulation for Lidar-only odometry problem solving it using a modified version of the Continuous-Time Iterative Closest Point (CT-ICP) algorithm presented in Dellenbach et al. [2021] for short sequences of KITTI-raw dataset Geiger et al. [2012].

Problem description

The main feature of method Dellenbach et al. [2021] is a combination of continuity in the scan matching and discontinuity between scans.

They use split rotation and translation representation and achieve time continuity for the scan matching by using spherical linear interpolation (slerp) for rotations and standard linear interpolation for translations.

We can very shortly describe their time-continuous scan matching procedure (for more details, please, refer to original paper Dellenbach et al. [2021]) as minimizing the following point-to-plane residual function with respect to starting and ending scan poses $\mathbf{T} = (\mathbf{T}_b, \mathbf{T}_e) \in \mathbf{SE}(3)^2$ for each i-th combination of sample point $\boldsymbol{p}_i^L$, map point $\boldsymbol{q}_i^W$ and related neighborhood normal n_i as

$$r_i[\mathbf{T}] = \left(\boldsymbol{p}_i^W[\mathbf{T}] - \boldsymbol{q}_i^W\right) \cdot \boldsymbol{n}_i, \tag{3.26}$$

$$\boldsymbol{p}_i^W[\mathbf{T}] = \mathbf{T}^{\tau_i}[\mathbf{T}]\boldsymbol{p}_i^L, \tag{3.27}$$

where $\mathbf{T}^{\tau_i}$ - interpolated pose for τ_i timestamp of scanning.

While the original work performs interpolation using split spherical linear interpolation (slerp) for rotations and linear interpolation for translations, we replace it by direct interpolation in manifold and evaluate the performance of our proposed approximated Jacobian on short sequences of ~ 100 frames from KITTI-raw Geiger et al. [2012] dataset.

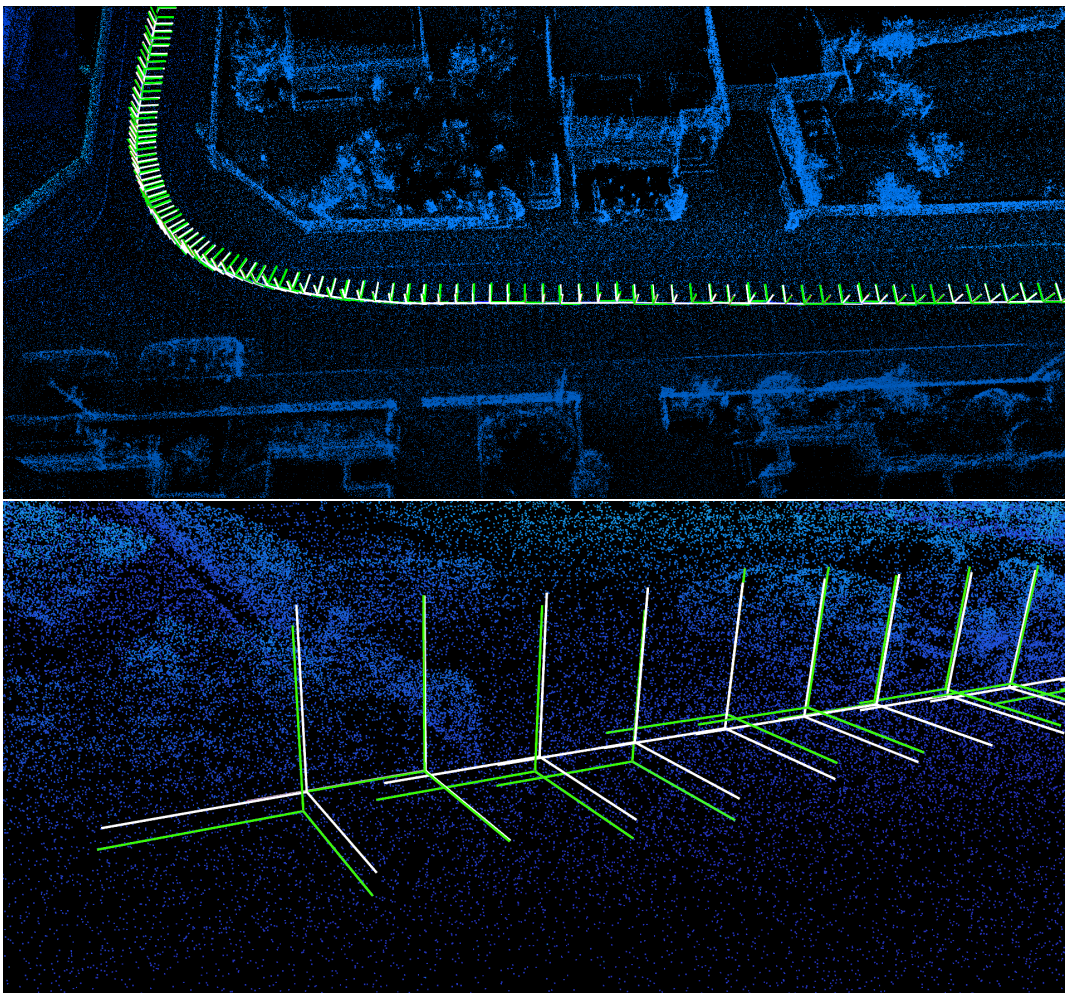


Figure 3-8: Lidar odometry results obtained using our proposed Jacobian approximation (green) and original CT-ICP method (white) on short sequences of KITTI-raw dataset

Experiment results

We use a mrob C++ library mrob as our optimization backbone providing it with approximated Jacobian and replacing the parts of the original pipeline that uses Ceres-solver Agarwal et al. [2022] framework alongside Jacobians obtained using its auto differentiation module.

We provide the qualitative visual results for short ~ 100 frames long subsequences from KITTI-raw Geiger et al. [2012] dataset scenes at Fig. 3-8. The green trajectory depicts the result obtained with our Jacobian formulation, while the white one describes the original CT-ICP result. The proposed Jacobian approximation results in a trajectory that is generally very close to the original result (top image) but does not replicate it exactly (bottom image).

Our Jacobian approximation results in stable trajectory and consistent maps for short sequences. Testing it with longer trajectory subsamples or reaching a closer match with the original CT-ICP results implies implementing additional constraints described in the original paper that ensure discontinuity consistency between sequential frames, which is out of the scope of the current work.

3.5 Summary

We have presented a Jacobian approximation of interpolated poses on $\mathbf{SE}(3)$ that can be used for tasks involving the direct optimization on manifold of trajectories in 3D. We provide its derivation and report its performance on several synthetic and real-world datasets and problems, showing its consistency in achieving correct results.

Chapter 4

Analytical Jacobian for Camera Pose Estimation with Splatting Based Rendering

In this chapter, we explore the complex topic of dense visual-only camera pose estimation through rendering. We introduce GSLoc, a novel visual localization method that performs dense camera alignment on manifold using 3D Gaussian SplattinVisual localization, the process of determining the camera pose using a visual representation of a known scene, plays an important role in various applications related to robot navigation, self-driving cars, and augmented/virtual reality Lynen et al. [2015], Heng et al. [2019]. In particular, the main objective of visual localization via camera alignment is, provided an input query image, to determine the 6 degrees of freedom (dof) camera pose (position and orientation) in a 3D environment with a known map representation. The map representation of a known scene, which is a core part of every localization method, can be of different forms. The most developed and commonly used ones are sparse map representations Collet et al. [2009], Li et al. [2018], which rely on a set of 2D-3D feature-landmark correspondences typically estimated using structure-from-motion (SfM) techniques Schonberger and Frahm [2016]. Despite their effectiveness in various localization scenarios, sparse map representations provide limited scene comprehension, falling short in empty spaces or textureless environments with no distinct features. Dense mapping is an alternative

family of representations that aim to utilize information from entire images but may require capturing depth, ensuring continuity of the input frames Engel et al. [2014, 2017], Forster et al. [2014], Zubizarreta et al. [2020]. Other methods may operate on dense image descriptors Taira et al. [2018], Arandjelović et al. [2016], usually extracted with convolutional neural networks (CNN). Methods of this category have proven their efficiency in large-scale scenarios and image retrieval tasks but have limited accuracy and produce only an approximated pose of the query camera. We focus in this chapter on utilizing the 3D Gaussian Splatting rendering technique as a map representation for visual localization tasks and aims to overcome the existing challenges described above. Our study includes an investigation of the viability of the 3DGS method as a map representation, a comprehensive convergence analysis for various camera initialization scenarios, an exploration of convergence limitations arising from the highly non-convex nature of the problem, and the proposal of a coarse-to-fine optimization strategy to mitigate such limitations. The main contributions are summarized as follows:g as the map representation of the scene.

GSLoc backpropagates pose gradients through the rendering pipeline to align the rendered and target images. It adopts a coarse-to-fine strategy that employs blurring kernels to address the non-convex nature of the problem by reformulating it into a sequence of subproblems that are easier to solve, resulting in improved convergence. The results demonstrate that our approach effectively achieves visual localization under challenging conditions, particularly in scenarios with relatively small overlap between the initial and target frames in textureless environments, where state-of-the-art neural sparse methods yield inferior results.

Additionally, leveraging the byproduct of realistic rendering from the 3DGS map representation, we illustrate how to enhance localization outcomes by combining a set of observed and virtual reference keyframes when addressing the image retrieval problem.

We evaluate our method using both synthetic and real-world data, highlighting its advantages and potential applications.

4.1 Introduction

Visual localization, the process of determining the camera pose using a visual representation of a known scene, plays an important role in various applications related to robot navigation, self-driving cars, and augmented/virtual reality Lynen et al. [2015], Heng et al. [2019]. In particular, the main objective of visual localization via camera alignment is, provided an input query image, to determine the 6 degrees of freedom (dof) camera pose (position and orientation) in a 3D environment with a known map representation.

The map representation of a known scene, which is a core part of every localization method, can be of different forms. The most developed and commonly used ones are sparse map representations Collet et al. [2009], Li et al. [2018], which rely on a set of 2D-3D feature-landmark correspondences typically estimated using structure-from-motion (SfM) techniques Schonberger and Frahm [2016]. Despite their effectiveness in various localization scenarios, sparse map representations provide limited scene comprehension, falling short in empty spaces or textureless environments with no distinct features. Dense mapping is an alternative family of representations that aim to utilize information from entire images but may require capturing depth, ensuring continuity of the input frames Engel et al. [2014, 2017], Forster et al. [2014], Zubizarreta et al. [2020]. Other methods may operate on dense image descriptors Taira et al. [2018], Arandjelović et al. [2016], usually extracted with convolutional neural networks (CNN). Methods of this category have proven their efficiency in large-scale scenarios and image retrieval tasks but have limited accuracy and produce only an approximated pose of the query camera.

Differentiable mesh-based rendering algorithms have also been employed for the visual localization task, leading to a family of dense map representation methods that can achieve impressive results. However, this comes at the significant cost of requiring a detailed 3D model of the environment Chen et al. [2020], Park et al. [2020]. This drawback has been mitigated with the introduction of Neural Radiance Field (NeRF) Mildenhall et al. [2021] models that can be trained using only a set of posed images. NeRFs can achieve photo-realistic rendering quality by implicitly

learning via 2D supervision the 3D scene as a function of a continuous radiance field. While NeRFs were originally introduced to deal with novel-view synthesis, their learned map representation has been recently used in the design of novel pose estimation methods. Started with a simple idea presented in iNeRF Yen-Chen et al. [2021], it continued with other sophisticated pose estimation approaches Maggio et al. [2023], Sucar et al. [2021]. However, despite their initial promising results, such methods still face a limited applicability since they suffer from the same drawbacks of NeRF models, that is extremely long training and rendering times due to the expensive backward mapping ray-casting procedure.

Recently, 3D Gaussian Splatting (3DGS) Kerbl et al. [2023] has been introduced and achieves high-quality real-time novel view synthesis at full HD resolution. This is an alternative learning-based approach that unlike NERF-based methods is based on a forward mapping/rasterization strategy. Specifically, 3DGS represents the 3D scene with a collection of 3D anisotropic Gaussians, which play the role of rendering primitives and whose parameters are directly optimized from a set of available posed images during training. The type of operations required by a 3DGS rasterizer are better suited for GPUs resulting in a very efficient and interactive novel view rendering process.

3DGS introduces a novel and distinctive map representation of the environment, which shows promise for effectively addressing the challenges associated with camera pose estimation and visual localization. The 3DGS strategy is computationally efficient and fully differentiable, facilitating the generation of highly realistic images in arbitrary views. Importantly, it allows for the direct flow of parameter gradients for any given camera pose, enabling real-time dense camera alignment, a capability not offered by other localization methods. Furthermore, it establishes a unique and fully-differentiable rendering-pose relation, enabling the generation of rendering images for any given camera and facilitating gradient-based optimization to refine its pose by minimizing the discrepancy between rendered and query images.

Nevertheless, there are still two challenges associated with this novel approach. The first one is that the accuracy of the initial camera pose used during training can have a significant impact on the success of the method. Secondly, the utilized

objective loss, which is based on the photometric difference, is highly non-convex due to the presence of high-frequency details in the images. The non-convex nature of the loss poses difficulties in its optimization, as it can lead to the entrapment of the optimization process to one of the numerous local minima, resulting in suboptimal solutions.

This work focuses on utilizing the 3D Gaussian Splatting rendering technique as a map representation for visual localization tasks and aims to overcome the existing challenges described above. Our study includes an investigation of the viability of the 3DGS method as a map representation, a comprehensive convergence analysis for various camera initialization scenarios, an exploration of convergence limitations arising from the highly non-convex nature of the problem, and the proposal of a coarse-to-fine optimization strategy to mitigate such limitations. The main contributions of this work are summarized as follows:

- We analytically derive the gradients of the 3DGS renderings with respect to to camera poses for pose optimization on manifold and implement a 3DGS-based visual localization pipeline. We also solve this task utilizing 3D+Quat pose parametrization and show advantages of optimization on manifold.

- We propose a coarse-to-fine optimization strategy where we apply gradually fading Gaussian blur on the query and rendered images that allows us to overcome the problem of suboptimal convergence for high-frequency image details.

- We propose an effective way of improving the localization results by enhancing camera initialization obtained via image retrieval by extending its image base with rendered camera frames.

- We evaluate our approach on indoor synthetic and real scenes, provide a comprehensive quantitative analysis of camera pose convergence based on various initial camera pose priors and parameterizations, and compare it with a sparse feature-based localization baseline.

- We have made the source code of the proposed method and its evaluation

publicly available at Botashev [2024] to ensure reproducibility and to provide transparency regarding all technical details.

4.2 Related work

4.2.1 Visual Localization methods

Classic sparse feature-based localization focuses on detecting and matching a sparse set of distinctive features or keypoints in the camera images Collet et al. [2009], Li et al. [2018]. The initial approach to feature matching involved manual design of keypoint detection algorithms that can identify visually salient image details: points, edges, and corners Bay et al. [2008]. However, the recent progress in the field, which has been driven by the introduction of dedicated neural networks for feature extraction Sarlin et al. [2020], has led to a revision of the feature extraction and matching stages. Indeed, these network architectures have demonstrated exceptional performance, achieving precise and robust feature matching results.

Consequently, the map representation in sparse feature-based localization can be constructed using network-extracted keypoints alongside their related 3D positions or descriptors. At the localization stage, the query image is processed to extract keypoints that are afterwards matched against the map to estimate the 6-DoF camera pose. Sparse feature-based methods are computationally efficient and have demonstrated robustness in a variety of applications. However, they cannot be used for tasks that require scene understanding while they also disregard useful volumetric context and, thus, can fail in featureless or empty environments.

On the other hand, dense visual localization methods aim to utilize an entire image dense map representation by matching visual information across the entire images using dense descriptors, such as pixel-level descriptors or dense feature maps. This type of representations encodes information about the appearance, texture, or semantic context of the scene Arandjelović et al. [2016].

4.2.2 Rendering-based Pose Estimation

The introduction of Neural Radiance Fields (NeRF) Mildenhall et al. [2021] and the large array of follow-up workYen-Chen et al. [2021], Maggio et al. [2023], Sucar et al. [2021] has brought a new paradigm to novel view synthesis and subsequently to camera pose estimation. In particular, NeRF represents the scene as a continuous 3D volume and learns the radiance field properties, resulting in a more accurate and realistic map representation of static scenes with intricate geometry and complex lighting. While NeRF-based camera pose estimation methods have certain advantages and can achieve competitive results, they also face a limited applicability due to the principal drawbacks of the NeRF model itself, including long model training and inference time due to the computationally expensive utilized backward mapping/ray-casting procedure.

Meanwhile, recent studies indicate that 3DGS-based camera pose estimation methods effectively circumvent these drawbacks and hold significant promise for future applications Matsuki et al. [2024], Sun et al. [2023].

4.3 Rendering with 3D Gaussian Splatting

This section starts with a further elaboration on the theory behind the 3DGS given in Zwicker et al. [2002]. We start with introducing the concept of ray space.

We denote a point in ray space by a column vector of three coordinates $\mathbf{x} = (x_0, x_1, x_2)^T$. The coordinates x_0, x_1 specify a point on the projection plane. The third coordinate x_2 specifies the Euclidean distance from the center of projection to a point on the viewing ray passing trough (x_0, x_1).

We proceed with the volume rendering equation and its low-albedo approximation, which is the main basis for all volume rendering approaches. We write volume rendering equation as

$$\hat{I}_\lambda(\mathbf{x}) = \int_0^L c_\lambda(\mathbf{x}, \xi) f_c(\mathbf{x}, \xi) e^{-\int_0^\xi f_c(\mathbf{x}, \mu) d\mu} d\xi \tag{4.1}$$

It describes $\hat{I}_\lambda(x)$ - the intensity of light that is projected along the ray x with

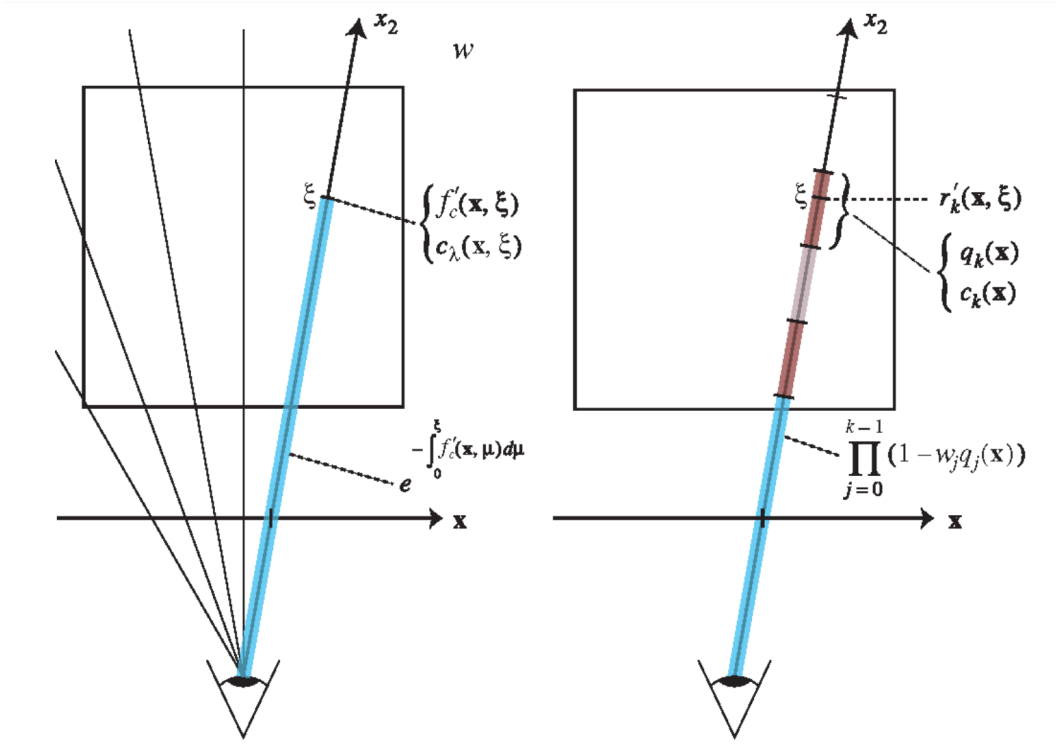


Figure 4-1: Volume rendering. Left: Illustrating the volume rendering equation Right: Approximation in typical splatting algorithms. Source: Zwicker et al. [2002]

length L and has wavelength λ. Here, $c_\lambda(\mathbf{x})$ is an emission coefficient, and $f_c(\mathbf{x})$ is the extinction function that describes the rate of light occlusion along a ray. Most of rendering approaches of all kinds (regardless of being forward or backward) are based on different approximations of this equation being simplified by some physical assumptions.

Splatting algorithms in general are no exceptions and obey the following physical model assumptions see Fig.4-1. We assume that volume is built with irregularly spaced individual particles that absorb and emit light. The position and shape of the particles are defined in some coordinate space by reconstruction kernels r_i and some constant opacity weights σ_k as

$$f_c(\mathbf{x}) = \sum_i \sigma_i r_i(\mathbf{x}) \tag{4.2}$$

Substituting eq. (4.2) into eq. (4.1) and using linearity of integration, we introduce additional simplifying assumptions and further approximate rendering equation. First, we impose reconstruction kernels $r_i(\mathbf{x})$ to have non-overlapping local support along

a ray in front-to-back order. Second, we assume that the emission coefficient is constant in each reconstruction kernel local support along a ray and that there is no self-occlusion. Finally, exploiting the mentioned assumptions, expanding the exponential term in the rendering integral, we can derive the following rendering equation approximation

$$\hat{I}(\mathbf{x}) = \sum_i \sigma_i c_i(\mathbf{x}) q_i(\mathbf{x}) \prod_{j=0}^{i-1} (1 - \sigma_j q_j(\mathbf{x})) \tag{4.3}$$

where $q_k(\mathbf{x})$ is an integrated reconstruction kernel defined as

$$q_i(\mathbf{x}) = \int_{\mathbb{R}} r_i\left(\mathbf{x}, x_2\right) dx_2 \tag{4.4}$$

The result eq. (4.3) is the foundation for all splatting algorithms. Also known as α-blending, it is widely used in other volumetric rendering methods such as NeRF.

The choice of a suitable function for the reconstruction kernel r_k is a central topic of discussion in splatting algorithms. EWA Splatting Zwicker et al. [2002] and 3DGS show that 3D Gaussians are very convenient reconstruction kernels due to the useful properties of the Gaussian functions.

We define a n-dimensional multivariate elliptical Gaussian centered at point $\boldsymbol{\mu} \in \mathbb{R}^n$ with covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{n \times n}$ as:

$$\mathcal{G}^n_{\boldsymbol{\Sigma}}(\mathbf{x} - \boldsymbol{\mu}) = \frac{1}{(2\pi)^{n/2} |\boldsymbol{\Sigma}|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x}-\boldsymbol{\mu})} \tag{4.5}$$

For a given set of $\mathcal{N}$ RGB images $\{I_n\}_{n=1}^{\mathcal{N}}$ and corresponding camera poses $T^c_{wn} \in \mathbf{SE}(3)$, 3DGS can learn a 3D scene representation that enables photo-realistic novel view rendering for an arbitrary camera pose. This is achieved by modeling the scene using a collection of $\mathcal{M}$ 3D Gaussians, which are defined in a world coordinate frame w as

$$\mathbf{G} = \{\mathcal{G}^3_{\boldsymbol{\Sigma}^{\mathbf{w}}_{\mathbf{i}}}(\mathbf{x} - \boldsymbol{\mu}^w_i), \sigma_i, \boldsymbol{c}_i\}_{i=1}^{\mathcal{M}}. \tag{4.6}$$

These Gaussians serve as rendering primitives and are fully described by their centers $\boldsymbol{\mu}^w_i \in \mathbb{R}^3$, covariance $\boldsymbol{\Sigma}^w_i \in \mathbb{R}^{3\times3}$, opacity $\sigma_i \in \mathbb{R}$ and view-dependent color $\boldsymbol{c}_i \in \mathbb{R}^3$.

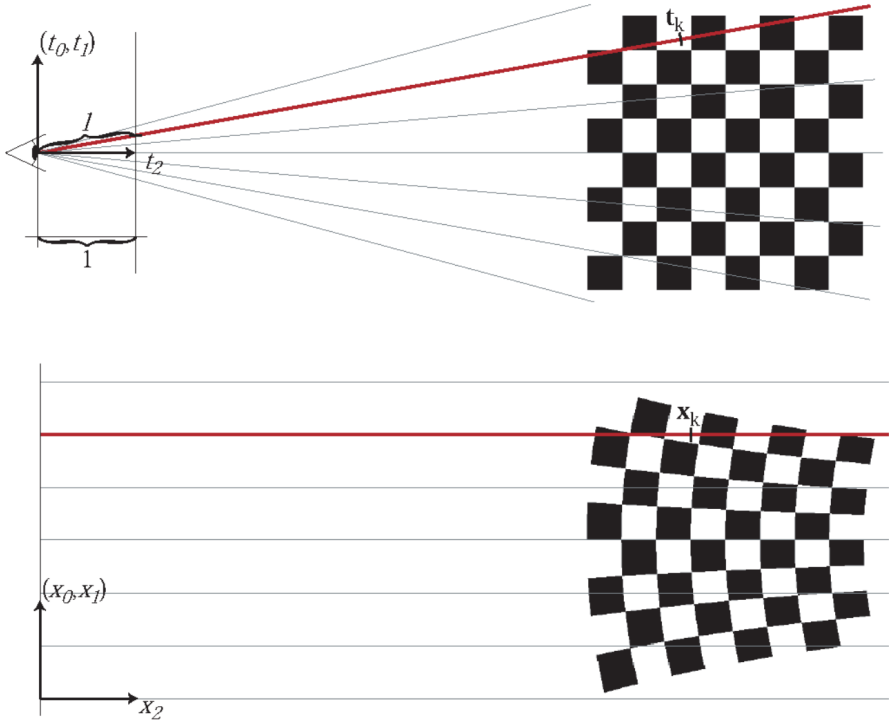


Figure 4-2: Transforming the volume from camera to image plane in ray space. Top: camera space. Bottom: ray space. Source: Zwicker et al. [2002]

Having the 3D Gaussian scene representation $\mathbf{G}$ at hand, rendering the image for a novel view, which is determined by a camera pose defined using a world w to camera c rigid body transformation $T_w^c = \{\mathbf{R}_w^c \in \mathbf{SO}(3), \mathbf{t}_w^c \in \mathbb{R}^3\} \in \mathbf{SE}(3)$, proceeds by perspectively projecting the Gaussians G_i^w to the image plane $\mathcal{I}$ in ray space. To do so, we first express the Gaussians w.r.t the camera frame with affine mapping φ_i, which leads to

$$\boldsymbol{\mu}_i^c = \varphi_i(\boldsymbol{\mu}_i^w) = \mathbf{R}_w^c \boldsymbol{\mu}_i^w + \mathbf{t}_w^c; \quad \boldsymbol{\Sigma}_i^c = \mathbf{R}_w^c \boldsymbol{\Sigma}_i^w \mathbf{R}_w^{c\intercal} \tag{4.7}$$

Then, all the Gaussians are perspectively converted to ray space and projected to the image plane (see Fig. 4-2) using an affine Taylor expansion approximation π_i of the projective non-linear transformation π that involves its Jacobian $\mathbf{J}_i$. This approximation leads to a mapping m_i of the initial 3D Gaussians to 2D Gaussians whose centers and covariances are expressed as

$$\boldsymbol{\mu}_i^{\mathcal{I}} = m_i(\boldsymbol{\mu}_i^w) = \pi_i(\boldsymbol{\mu}_i^c) = \pi_i(\varphi_i(\boldsymbol{\mu}_i^w)) \tag{4.8}$$

where $\pi_i(\boldsymbol{\mu}) = \pi(\boldsymbol{\mu}_i^c) + \mathbf{J}_k \cdot (\boldsymbol{\mu} - \boldsymbol{\mu}_i^c)$

$$\boldsymbol{\Sigma}_i^{\mathcal{I}} = \mathbf{J}_i \boldsymbol{\Sigma}_i^c \mathbf{J}_i^\intercal = \mathbf{J}_i \mathbf{R}_w^c \boldsymbol{\Sigma}_i^w \mathbf{R}_w^{c\,\intercal} \mathbf{J}_i^\intercal \tag{4.9}$$

The corresponding Jacobian $\mathbf{J}_k$ is given by the partial derivatives of π at the point $\boldsymbol{\mu}_i^c$ as

$$\mathbf{J}_i = \frac{\partial \pi\left(\boldsymbol{\mu}_{\boldsymbol{i}}^{\boldsymbol{c}}\right)}{\partial \boldsymbol{\mu}} = \begin{pmatrix} 1/\mu_{i,2}^c & 0 & -\mu_{i,0}^c/{\mu_{i,2}^c}^2 \\ 0 & 1/\mu_{i,2}^c & -\mu_{i,1}^c/{\mu_{i,2}^c}^2 \\ \mu_{i,0}^c/l' & \mu_{i,1}^c/l' & \mu_{i,2}^c/l' \end{pmatrix}, \tag{4.10}$$

where $l' = \left\| \left(\mu_{i,0}^c, \mu_{i,1}^c, \mu_{i,2}^c\right)^T \right\|$.

We choose the resulting 3D Gaussian functions $\mathcal{G}^3_{\boldsymbol{\Sigma}_{\mathbf{i}}^{\mathcal{I}}}(\mathbf{x} - \boldsymbol{\mu}_i^{\mathcal{I}})$ as a reconstruction kernel r_i in (4.4) which results in

$$q_i(\mathbf{x}) \simeq \int_{\mathbb{R}} \mathcal{G}^3_{\boldsymbol{\Sigma}_i^{\mathcal{I}}}\left(\mathbf{x} - \boldsymbol{\mu}_i^{\mathcal{I}}, x_2 - \boldsymbol{\mu}_{i2}^{\mathcal{I}}\right) dx_2 = \mathcal{G}^2_{\hat{\boldsymbol{\Sigma}}_i^{\mathcal{I}}}\left(\mathbf{x} - \boldsymbol{\mu}_i^{\mathcal{I}}\right), \tag{4.11}$$

where $\hat{\boldsymbol{\Sigma}}_i^{\mathcal{I}} \in \mathbb{R}^{(n-1)\times(n-1)}$ is left upper part of $\boldsymbol{\Sigma}_i^{\mathcal{I}} \in \mathbb{R}^{n\times n}$ constructed by excluding the last n-th row and the column from it.

Finally, using this result the image intensity $\hat{I}$ is computed with (4.3) via depth-ordered α-blending of the projected Gaussians following

$$\hat{I} = \sum_{i \in \mathcal{M}} \mathbf{c}_i(\mathbf{d}_i) \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j), \tag{4.12}$$

where the density α_i is computed as a multiplication of opacity σ_i and evaluated reconstruction kernel, $\mathbf{c}_i(\mathbf{d}_i)$ is the view-dependent color of the 3D Gaussians defined with spherical harmonics and computed based on the view-direction vector $\mathbf{d}_i = (\boldsymbol{\mu}_i^w - \mathbf{t}_c^w)/\|(\boldsymbol{\mu}_i^w - \mathbf{t}_c^w)\|$.

Starting with some sparse SfM Schonberger and Frahm [2016] point cloud initialization and following the above described fully-differentiable rendering procedure, 3DGS gradually optimizes the 3D Gaussian parameters with gradient descent by minimizing loss $\mathcal{L}$

$$\min_{\mathbf{G}} \sum_n \mathcal{L}(\hat{I}_n(G_n), I_n) \tag{4.13}$$

equal to weighted combination of L_1 and D-SSIM losses between the rendered image $\hat{I}_n(T^c_{wn}, \mathbf{G})$ and the ground-truth posed image I_n in dataset.

4.4 Method

The learned 3DGS scene model $\mathbf{G}$ serves as a novel and distinctive map representation of the 3D environment and is potentially highly suitable for being utilized in the visual localization task. Specifically, for a query image $\tilde{I}$ the corresponding pose $\tilde{T}^c_w = \tilde{T} \in \mathbf{SE}(3)$ can be found by minimizing the discrepancy between the rendered $\hat{\tilde{I}}$ and query images defined as

$$\tilde{T} = \underset{T \in \mathbf{SE}(3)}{\arg\min}\, \mathcal{L}_1(\hat{\tilde{I}}(T, \mathbf{G}), \tilde{I}). \tag{4.14}$$

Ostensibly, the solution of this task seems to be straightforward thanks to the photo-realistic real-time rendering capabilities of 3DGS. However, there are several aspects that require specific attention and which we address next.

4.4.1 Camera Pose Gradients

To achieve real-time rendering performance, 3DGS utilizes GPU capabilities and its official implementation of the rasterization step is based in CUDA. This precludes out-of-the-box automatic differentiation and instead requires the derivation of the explicit form of gradients for all the parameters to be optimized. To enable camera pose optimization, it is also required to express analytically all the gradients related to the poses parameters.

Since in the original work of 3DGS the authors did not optimize the camera poses we need to derive all the gradients related to the pose parameters that we wish to estimate. We follow our general approach explained in Chapter 2 and implement the camera pose optimization on the manifold and use Lie algebra to derive the camera pose Jacobians for the terms in Eq. (4.8) and Eq. (4.12) via the chain rule as

$$\frac{\partial \boldsymbol{\mu}^{\mathcal{I}}_i}{\partial T^c_w} = \frac{\partial \boldsymbol{\mu}^{\mathcal{I}}_i}{\partial \boldsymbol{\mu}^c_i} \frac{\partial \boldsymbol{\mu}^c_i}{\partial T^c_w} \tag{4.15}$$

$$\frac{\partial \boldsymbol{\Sigma}_i^{\mathcal{I}}}{\partial T_w^c} = \frac{\partial \boldsymbol{\Sigma}_i^{\mathcal{I}}}{\partial \mathbf{J}} \frac{\partial \mathbf{J}}{\partial \boldsymbol{\mu}_i^c} \frac{\partial \boldsymbol{\mu}_i^c}{\partial T_w^c} + \frac{\partial \boldsymbol{\Sigma}_i^{\mathcal{I}}}{\partial \mathbf{R}_w^c} \frac{\partial \mathbf{R}_w^c}{\partial T_w^c} \tag{4.16}$$

$$\frac{\partial \mathbf{c}_i}{\partial T_w^c} = \frac{\partial \mathbf{c}_i}{\partial \mathbf{d}_i} \frac{\partial \mathbf{d}_i}{\partial \mathbf{t}_c^w} \frac{\partial \mathbf{t}_c^w}{\partial T_w^c}. \tag{4.17}$$

We compute the derivatives on the manifold using the results obtained in Section 2.8 as

$$\frac{\partial \boldsymbol{\mu}_i^c}{\partial T_w^c} = \begin{bmatrix} \mathbf{I} & -\boldsymbol{\mu}_i^{c\wedge} \end{bmatrix}; \; \frac{\partial \mathbf{t}_c^w}{\partial T_w^c} = \begin{bmatrix} \mathbf{R}_w^{c\intercal} & \mathbf{0} \end{bmatrix} \tag{4.18}$$

$$\frac{\partial \mathbf{R}_w^c}{\partial T_w^c} = \begin{bmatrix} \mathbf{0} & -\mathbf{r}_{c1}^{\wedge} \\ \mathbf{0} & -\mathbf{r}_{c2}^{\wedge} \\ \mathbf{0} & -\mathbf{r}_{c3}^{\wedge} \end{bmatrix} \tag{4.19}$$

where $^\wedge$ denotes the skew symmetric matrix constructed from the corresponding input vector and $\mathbf{r}_{cj}$ denotes the j-th column of the rotation matrix $\mathbf{R}_w^c$.

Using the equations (4.15)-(4.19) it is possible to propagate all the necessary gradients for the camera pose optimization task. We iteratively solve the optimization problem of (4.14) for 2000 steps or until convergence using the first-order Adam Kingma and Ba [2015] optimizer with exponentially decaying learning rate. More details are provided in the following sections.

Manifold-derived pose Jacobians have a minimal 6 DoF representation and lead to a better convergence compared to alternative parametrizations. In particular, according to our ablation study, the manifold optimization achieves better results and shows clear advantages compared to the common alternative parametrization utilizing quaternions for rotation and 3D vectors for translation.

4.4.2 Impact of Initial Camera Pose Proximity

Among the most important factors that affect the final result of visual localization is the proper initialization of the camera poses. Finding an initial camera pose that exhibits a sufficiently large overlap with the query camera frame is crucial and can be one of the main factors of success or failure. This problem, which is also known as the Image Retrieval task, is a separate long-standing computer vision problem that requires special attention on its own. While, a 3DGS-based solution of this task

Figure 4-3: Visual explanation of 3D Intersection over Union (IoU) metric used for camera frames proximity estimation. Computed with voxels of the scene, this metric naturally describes both proximity of the camera poses and the visual similarity of their image frames. Here for the visualized frames the 3D IoU is equal to 0.15.

is an intriguing possible research direction, here we focus on solving the exact visual localization task. As a result, finding the best possible existing Image Retrieval algorithm for the camera pose initialization for GSLoc is out of scope of this work.

Instead, we seek to perform a comprehensive analysis of our method. We aim to estimate the dependency between obtaining the correct result with GSLoc and the proximity of the initial camera frame to the target one. In other words, we want to answer the following questions: 1) How close/far our initial guess of the camera pose need to be so that our method leads to the correct solution? and 2) What are the chances of this convergence?

To answer these questions, we propose to measure the camera frames proximity using the 3D Intersection over Union (IoU) metric that is computed using voxels of the scene. As visualized in Fig. 4-4, this metric naturally describes both proximity of the camera poses and the visual similarity of their corresponding image frames and enable us to quantitatively assess their impact on the final result. For our particular task the 3D IoU is more informative and intuitive compared to simple rotation and translation distances.

4.4.3 Extending image retrieval database with renderings

Besides the 3D IoU criterion, we also evaluate the results of the visual localization for initial poses corresponding to the closest dense image descriptors. Following a widely adopted approach, we extract global image descriptors using NetVLAD Arandjelović et al. [2016] and for each query image we find the most similar ones in the pool of images used for 3DGS training. Next, we solve the visual localization task by initializing the camera pose with these closest matches.

This is a very common approach widely adopted as a first step for sparse feature based visual localization. Its results directly depend on the number and diversity of images used as a map for comparison. However, 3DGS-based map representation allows us to overcome this limitation by extending the original set of images used both for 3DGS training and image retrieval step with any number of posed photorealistic scene renderings produced with the optimized 3DGS scene. Starting with a limited set of images, 3DGS allows us to extend our image base used for Image Retrieval by creating arbitrary novel-view renderings. This increases the probability of obtaining a good initial pose and as a result increases our chances of converging to the correct solution. Further, we show the effectiveness of this proposed technique both for synthetic and real scenes in evaluation.

4.4.4 Coarse-to-fine Rendering Scheduling

The highly non-convex nature of the photometric $\mathcal{L}_1$ loss w.r.t the 6 DoF space of camera poses in $\mathbf{SE}(3)$ poses a significant challenge related to its optimization. This non-convexity is caused by high-frequency image details and can lead to the entrapment of the optimization process to a bad local minima, which in turn can lead to a sub-optimal solution.

The visual representation of one such case is depicted in Fig. 4-4(a)-(b). Iteratively minimizing the objective function in (4.14) in a standard way leads to the convergence of the first-order method to a sub-optimal solution. Indeed, it is clearly visible that during standard optimization using Adam Kingma and Ba [2015] does not manage to escape the local minima caused by the sub-optimal overlap between the intermediate

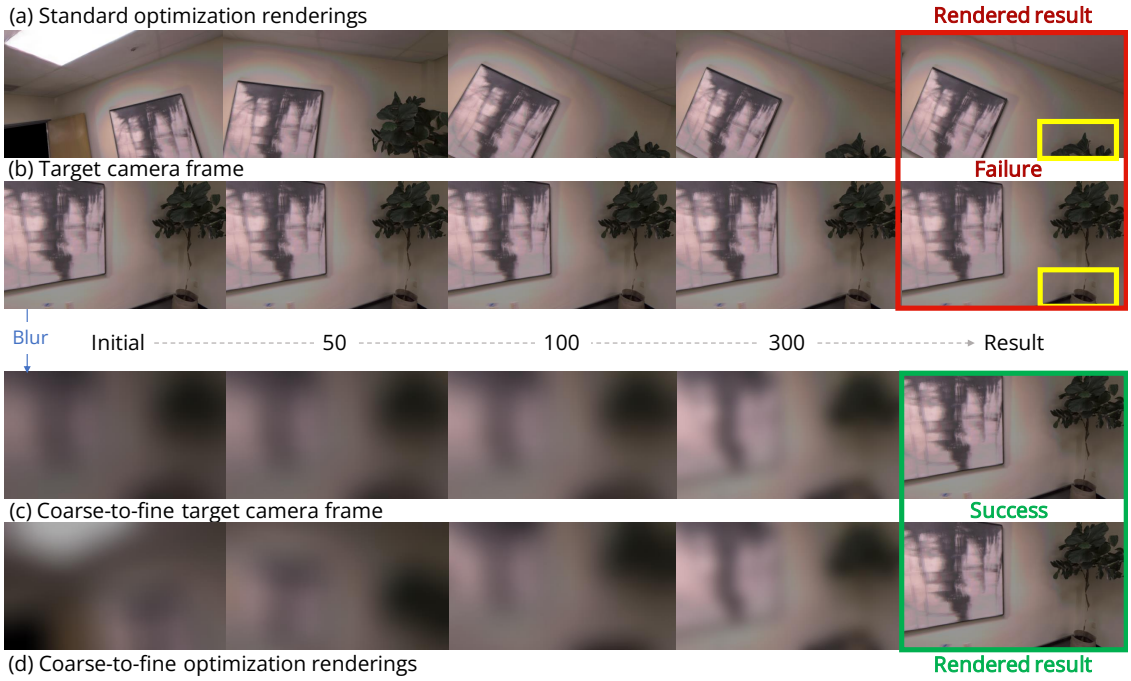


Figure 4-4: Visualization of the camera pose alignment process induced by iterative optimization of photometric loss between intermediate renderings and target images for standard (a)-(b) and coarse-to-fine (c)-(d) strategies. Standard optimization decsribed with (a)-(b) leads to convergence to a sub-optimal solution: it does not manage to escape the local minima caused by the sub-optimal overlap between the intermediate rendering and the target query image (highlighted with yellow) resulting to an unsuccessful image alignment. On the contrary, smoothing the image gradients with our coarse-to-fine approach (c)-(d) allows us to avoid being trapped in local minima and converge to the correct camera pose.

rendering and the target query image. This results to an unsuccessful image alignment (highlighted with yellow).

To overcome this problem we propose a simple yet effective coarse-to-fine strategy of applying a progressively decaying Gaussian blur both on the rendered and target images. Specifically, we convolve both the target query image and the intermediate rendering with a 2D Gaussian kernel $\mathcal{N}_{2d}(\delta_j) \in \mathbb{R}^{L \times L}$ of fixed size L while gradually decreasing its covariance δ_j. This results in a modified objective function $\mathcal{L}_1(\mathcal{N}_{2d} * \hat{\tilde{I}}(T, \mathbf{G}), \mathcal{N}_{2d} * \tilde{I})$ with smoothed gradients and a stabilized camera pose estimation as depicted in Fig. 4-4(c)-(d). Smoothing the image gradients allows us to avoid being trapped in local minima and converge to the correct camera pose. We have also found it to be effective the strategy of running several passes of coarse-to-fine optimization, restarting each new pass with the result from the previous one. Based on the above,

we have concluded that the highest efficiency is achieved by the following two-step GSLoc algorithm: 1) In the first step a standard camera pose optimization takes place. 2) If the first step does not recover the correct pose (the photometric loss between the rendered image and the query image exceeds a user-defined threshold), then we restart the entire process and apply the described coarse-to-fine optimization strategy.

4.5 Evaluation

We assess the performance of our proposed method by performing extensive experiments on 5 synthetic scenes from the Replica Straub et al. [2019] dataset. Our motivation for using this data in our evaluation stems from several reasons that we discuss next. The first reason is that the use of synthetic data ensures that the 3DGS map representation $\mathbf{G}$ is adequately learned. This can be achieved by exploiting the available accurate ground truth poses and depth information during the 3DGS training. In turn, this allows us to neglect any possible negative effects of the incorrect map representation on the visual localization results and validate the efficiency of the proposed method without worrying about data-related inaccuracies. Another reason is that we have access to the ground truth poses which allows us to accurately compute the localization errors and perform a precise evaluation of the proposed method. Finally, in order to conduct a comprehensive study of the effect of pose initialization based on the camera proximity according to the 3D voxel-based IoU, we need access to the detailed 3D voxel model of the scene, which we can accurately extract from such synthetic data. Nevertheless, we also evaluate our method on 2 real scenes from the Deep Blending dataset Hedman et al. [2018] and show the coherence of the obtained real-data results with the synthetic ones.

4.5.1 Synthetic Data and Initial Camera Analysis

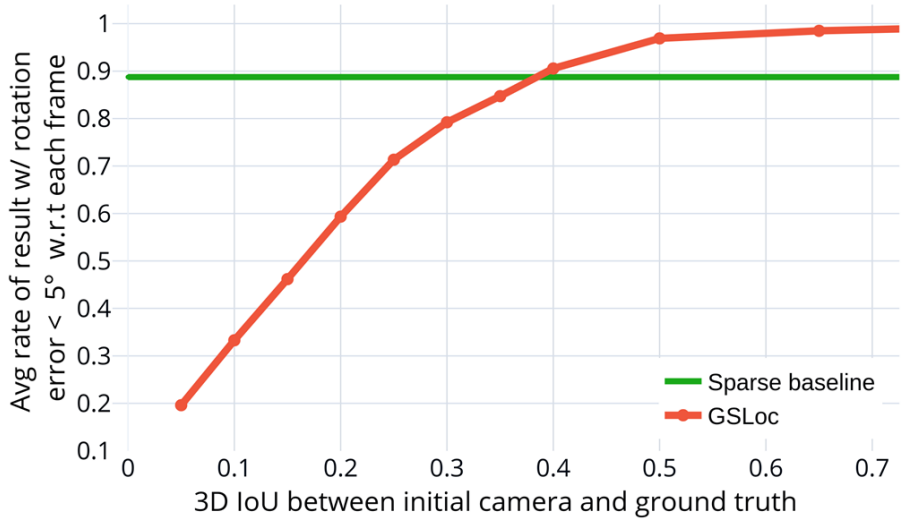
Setting

We perform our study using 5 scenes from the Replica Straub et al. [2019] dataset. For each scene we assume that the camera intrinsics are known and utilize a base trajectory of approximately 200 frames that describe in detail the environment. These frames are accompanied by a depth estimated point cloud and camera poses.

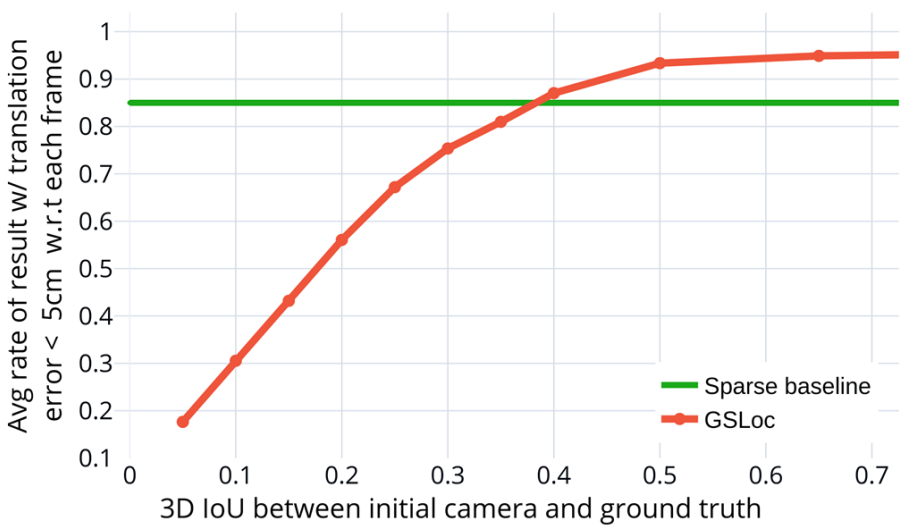
Next, for each scene we capture a diverse set of 32 test query frames that we then use for the visual localization evaluation. For each of these query frames we randomly generate 16 different pose initializations related to each one of 10 different 3D IoU levels lying in the range between 0.05 and 0.65. Overall, we utilize 5 scenes - 32 query frames - 10 IoU levels - 16 different initalization poses. In total this amounts to approximately 25k different localization tasks, which allow us to thoroughly validate the performance of GSLoc, its strengths and possible limitations.

In order to not significantly deviate from a realistic setup, we process the base images using SfM to estimate the camera poses and the 3D points, with Sarlin et al. [2019] serving as the SfM reconstruction method. Specifically, we first extract local feature descriptors with SuperPoint DeTone et al. [2018] and global image descriptors with NetVLAD Arandjelović et al. [2016]. Based on the similarity of NetVLAD decriptors we then estimate the top 5 neighbors for each image. After that we use SuperGlue to perform feature matching of each image with it's top 5 neighbors. The rest of the reconstruction is performed with COLMAP's Schonberger and Frahm [2016] incremental mapping.

This SfM reconstruction is a priori not perfect due to the inherent flaws of such methods. Therefore, in order to minimize the impact of these map reconstruction errors on visual localization, we replace all the poses and 3D points in the SfM reconstruction with the ground truth ones. One can interpret this choice as if we had a flawless SfM reconstruction. With this strategy, we manage to avoid overly refined experiments setup, keeping the evaluation clean but still realistic. After this, we use this clean SfM reconstruction both for learning the 3DGS scene map representation and as a database for the sparse matching based visual localization baseline.



(a) Rotation results



(b) Translation results

Figure 4-5: Quantitative results of GSLoc on synthetic scenes from Replica Straub et al. [2019] dataset compared with sparse feature-matching baseline. Provided results show the dependency between obtaining the correct pose with GSLoc and the proximity of the initial camera frame to the target one. With the increase of the frames' proximity, GSLoc first reaches and then surpasses the baseline. We report the results separately for rotation (a) and translation (b) pose components.

The localization baseline that we use for comparison is based on HLoc Sarlin et al. [2019] and consists of the following steps. For the query image I_q we perform feature extraction with SuperPoint DeTone et al. [2018]. Then we extract its NetVLAD Arandjelović et al. [2016] global descriptor, find the top 5 neighbors from the database and perform feature matching with SuperGlue. This gives us 2D-3D correspondences

that we use in PnP RANSAC to estimate the absolute pose. The final pose of the query image is obtained after the non-linear refinement with the Levenberg-Marquardt algorithm.

Results

For all the 5 scenes, we have in total 160 test query frames to localize. Following Yen-Chen et al. [2021], we identify the localization as successful if the resulting pose error is less than 5 degrees for rotation and 5cm for translation. While for the baseline the result is binary: either success or failure, for GSLoc method we separately evaluate each of 10 IoU proximity levels trying to localize each frame with 16 different initializations. Hence, for each IoU level the result of localizing each of the 160 test frames is not binary but the 0-1 ratio describing the percentage of successful results for 16 different initialization per query frame. By averaging this ratios w.r.t to all query frames, we evaluate the efficiency of our method and present the results along with the baseline comparison in Fig. 4-5.

These results indicate that with an increased proximity of the initial camera pose the 3DGS based visual localization improves its results getting close and outperforming the sparse feature based localization after reaching the threshold of ~ 0.4 3D IoU. Although, it may seem that the baseline results are not perfect, it has in fact managed to solve most of the localization tasks, failing only for extreme featureless images. While such cases are almost impossible to solve with sparse methods, GSLoc handles them well due to its dense alignment nature. Based on the above, we conclude that the 3DGS based map representation used by GSLoc is suitable for solving visual localization tasks and can lead to competitive results. We also show that as any other visual localization methods, GSLoc requires a certain level of proximity of the initial camera pose used for optimization. We elaborate on this aspect of the problem in the next section.

4.5.2 Enhanced Camera Initialization with Image Retrival on Extended Image Base

The camera initialization for visual localization problems is typically obtained by solving the Image Retrieval task. A common way to do this is by finding the closest image descriptors in an image database. For instance, such approach corresponds to the preliminary step used in our sparse baseline. In this section we investigate whether such resulting poses are suitable enough for being used as initialization within GSLoc.

Setting

We follow the same setup as the one used in the previous experiment. The only major difference here is that instead of investigating different 3D IoU levels for initial camera poses, we simply follow our sparse baseline and initialize the pose with those that have 5 closest NetVlad Arandjelović et al. [2016] descriptors among images in our base trajectory. Here we switch on the binary result classification, assuming the localization of the query to be successful if at least one out of five camera initialization lead GSLoc to the correct final solution.

Further, we utilize the learned 3DGS scenes to extend our base trajectory image databases used for the image retrieval task. We carefully capture an additional set of camera frames extending the existing database and increasing its diversity and scene comprehension.

Results

We report the results for the experimental setup described above in Fig. 4-6. We describe the percentage of the successfully localized frames for each individual room of Replica Straub et al. [2019] dataset. Detailing the previously reported results, the baseline method initialized with closest descriptors manages to achieve the correct result for most of the cases, failing only for frames that are dominated by empty space as in the “office 2" scene.

In contrast, GSLoc initialized with the original base map dense matches, exhibits

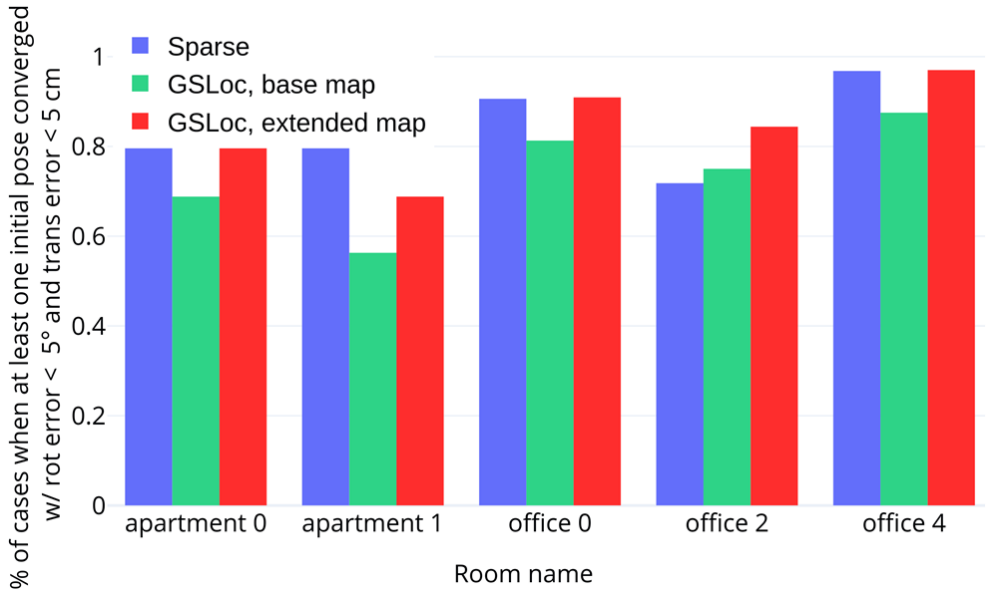


Figure 4-6: Quantitive results on the synthetic scenes from Replica Straub et al. [2019] dataset. Enhancing the GSLoc camera initializations obtained by the image retrieval with the rendering-extended imagebase leads to consistent success rate improvement proving the efficiency of the proposed method.

a slightly worse performance, on average having less successfully localized results. What is interesting is that extending the image base with the renderings actually helps to improve its performance and on average increases the GSLoc success rate by 10% bringing it closer to the baseline. Investigating the reasons for this improvement, we figured out that the median 3D IoU for the camera initializations estimated with dense matching on the original database is ~ 0.3 and increases to ~ 0.4 with the database rendering extension. This is an indirect confirmation of the results previously reported in Fig. 4-5. This experiment proves the efficiency of the proposed base map extension technique and reveals a potential of utilizing 3DGS-based methods for successfully solving the image retrieval task.

Concluding the discussion of the results obtained on synthetic data, we report the average numerical pose estimation errors of successfully localized frames for all previously reported experiments in Tab. 4.1. We show that the successful localization with GSLoc leads to comparable or even smaller pose errors compared with the sparse baseline method. Besides that, we also show that successfully localized frames also achieve higher visual metrics, which directly relate to the quality of the pose estimation and therefore might be used as a criterion for deciding if localization has

been successful.

Method	Rotation error, deg		Translation error, cm		Mean PSNR, dB
	Mean	Median	Mean	Median	
Sparse baseline	0.098	0.058	0.498	0.295	-
GSLoc, init with 5 desc.	0.091	0.054	**0.487**	0.286	**35.52**
GSLoc, avg for all IoUs	**0.078**	**0.035**	0.534	**0.264**	35.36

Table 4.1: Average pose estimation errors for successfully localized frames. GSLoc leads to comparable or even smaller pose errors compared with the sparse baseline method.

4.5.3 Real Data Results

In the last experiment, we show that the results obtained for the synthetic data remain valid for the real-world datasets. To do so, we conducted similar experiments with camera initialization with NetVLAD descriptors and image base map rendering extension for 2 real indoor scenes from the Deep Blending dataset Hedman et al. [2018]. Each scene is represented with a comprehensive set of arbitrary posed photos.

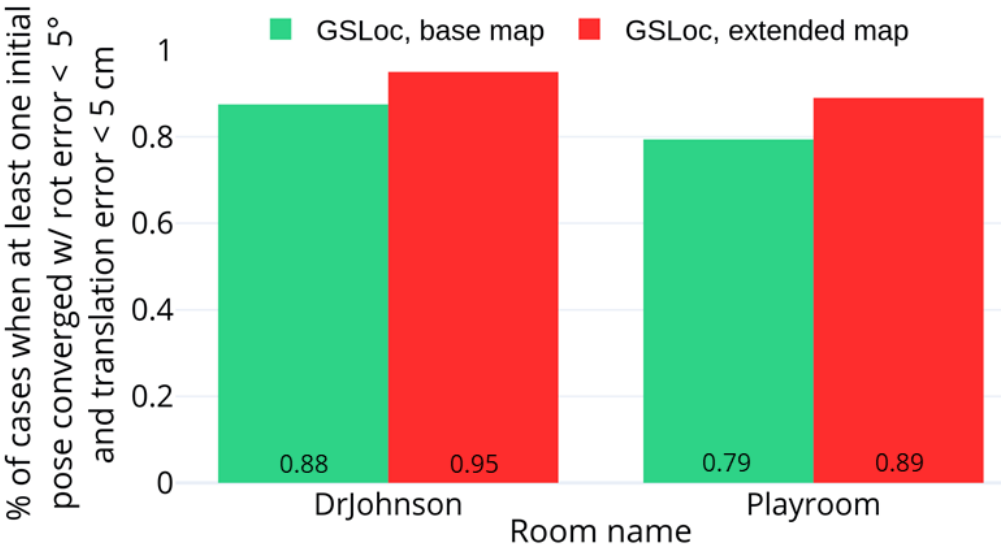


Figure 4-7: Quantitive results on the real scenes from Deep Blending Hedman et al. [2018] dataset. Enhancing the GSLoc camera initializations obtained by the image retrieval with the rendering-extended imagebase leads up to 10 % success rate improvement matching the observations obtained with synthetic data.

We form a test query set by taking each 8^{th} image of the set. We use the rest of the frames as the image base firstly for the SfM reconstruction and 3DGS training and secondly for the initial pose estimation with closest descriptors. Following the same experimental design, we again extend the image base with 3DGS renderings and showcase that the effectiveness of this technique remains valid also for the real scenes. We report our results in Fig. 4-7. We observe that the results achieved for the real scenes are consistent with those of the synthetic scenes. This is a strong indication of the suitability of GSLoc for real-world visual localization.

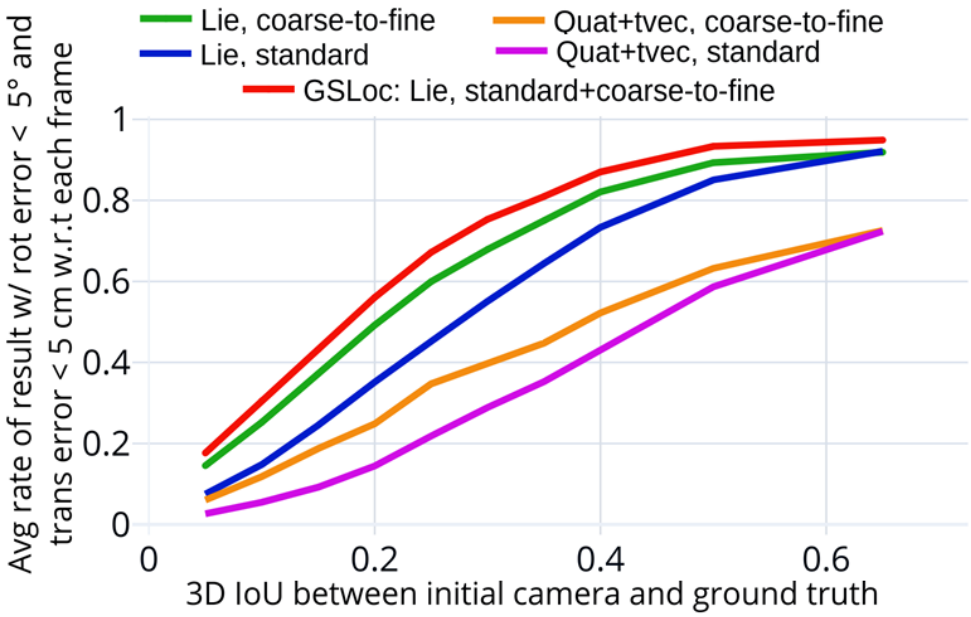
4.6 Ablation Study

We perform a ablation of our GSLoc method by re-running the experiment described in the previous section utilizing different optimization strategies and pose parametrizations. We summarize the results in Fig. 4-8(a) showing the advantage of the proposed two-step standard+coarse-to-fine GSLoc optimization on manifold over other methods.

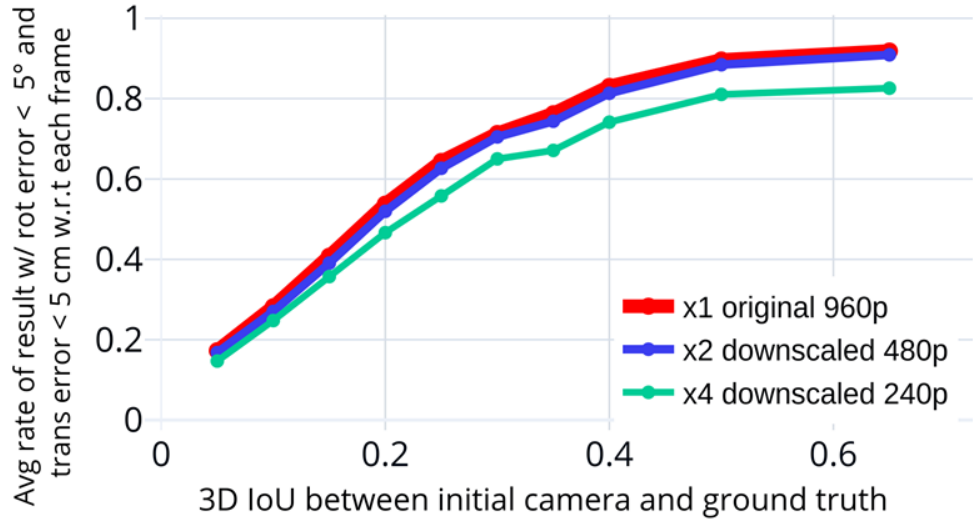
Furthermore, we estimate how the rendering resolution affects the GSLoc localization results, describing the details in Fig. 4-8(b). While we run all our experiments on an image resolution of 960x540 pixels we see that the decrease of image size results in almost identical results for 2x downscaling and starts suffering localization degradation only at a 4x image downscaling.

4.7 Limitations and Future Work

While GSLoc shows promising results for visual localization, there are a few issues and limitations that we did not address in this work. An important one is the running-time efficiency of the method, which has not been optimized. Both, the multi-step sequential optimization as well as the use of a first order gradient descent method can lead to an increased execution time. In the future, we plan to improve the time efficiency of GSLoc and mitigate the current limitations by utilizing second order optimization algorithms.



(a) Method and state representation ablation



(b) Resolution ablation

Figure 4-8: Ablation study on optimization strategy and pose parametrization (a) and rendering resolution (b). Proposed two step standard+coarse-to-fine GSLoc optimization on manifold outperforms other methods and allows running on 2x downscaled images without results degradation.

4.8 Implementation and Runtime Details

We modify the original CUDA-based implementation Kerbl et al. [2023] of the differentiable renderer enabling camera pose-related gradients. Please, refer to Botashev [2024] for the source code of our implementation. We solve optimization using Adam Kingma and Ba [2015] optimizer for 2000 steps or until convergence

when the loss change is smaller than 10^{-5} for 3 consecutive iterations. On average, for cases that achieve a successful outcome the number of necessary iterations may vary between 100-300 iterations and take around 5-15 seconds to converge on a modern GPU. The optimization learning rate starts with 10^{-2} and exponentially decays with iterations to 10^{-5}. The Gaussian blur is applied for the first 1000 iterations, its kernel covariance δ_j decays linearly from 10^{-1} to 10^{-4}, its kernel size L is 200 pixels for standard resolution experiments and is decreased for resolution ablation according to the image downscale factors. For the coarse-to-fine optimization strategy we decide that the localization successfully converged and should not be restarted if the rendered image PSNR has reached 25 dBs.

4.9 Summary

We have investigated a problem of dense visual camera pose on manifold estimation with rendering and presented the GSLoc - a novel visual localization technique based on 3D Gaussian Splatting environment map representation. We have demonstrated both on synthetic and real data that our method is capable of performing accurate camera pose estimation. We have confirmed it via a comprehensive convergence analysis of various camera initializations and parametrizations. We have thoroughly explored the convergence limitations due to non-convexity of the photometric loss and proposed a coarse-to-fine strategy to mitigate this issue. Finally, we have proposed an effective way to improve the localization results by enhancing the GSLoc camera initialization, which is obtained by image retrieval with a refined image base that is extended with 3DGS-rendered camera frames.

Chapter 5

Closing remarks

This dissertation has explored relevant challenges in the area of state estimation within robotics and computer vision, focusing primarily on time-continuous trajectory representation and visual localization using advanced rendering techniques. By combining theoretical frameworks with empirical evaluations, the research aims to enhance the accuracy and efficiency of robotic systems.

5.1 Time-Continuous State Estimation

The first part of this work concentrated on time-continuous state estimation and on-manifold optimization, particularly in relation to trajectory representation via direct linear interpolation on $\mathbf{SE}(3)$. A contribution of this research is the derivation of an analytical form of the Jacobian for linearly interpolated poses on $\mathbf{SE}(3)$, utilizing an approximation based on the commutativity property of infinitesimal group elements.

Empirical evaluations were conducted using various synthetic and real-world datasets, demonstrating the effectiveness of this analytical approach in performing on-manifold optimization for 3D trajectories. The experiments assessed the applicability of the proposed approximated gradient across different scenarios that utilize time-continuous trajectory representation and optimization.

In the initial experiment, the framework was tested on a time-continuous point-cloud alignment problem with synthetically generated data. The results of the proposed gradient formulation were compared to Barfoot's formulation and a numer-

ically estimated gradient obtained through the finite difference method. The findings indicated that all three methods produced comparable outcomes, with Barfoot's approach successfully reproducing numerical results, while our method achieved similar optimization results with improved computational efficiency, requiring less time to reach convergence.

Subsequently, the proposed gradient formulation was shown to be effective in addressing the time-continuous Pose Simultaneous Localization and Mapping (SLAM) problem using the synthetic Sphere dataset. Additionally, the applicability of the proposed gradient was illustrated in the Continuous-Time Iterative Closest Point (CT-ICP) algorithm, achieving stability in trajectories and mapping for short sequences from the uncorrected KITTI dataset.

5.2 Visual Localization with 3D Gaussian Splatting

The latter half of this dissertation focused on dense visual camera pose estimation, employing the 3D Gaussian Splatting (3DGS) method as a map representation. This section outlined the method's effectiveness for visual localization tasks, supported by convergence analyses that examined various initial camera pose priors and parameterizations, compared with a sparse feature-based localization baseline.

The research demonstrated that the proposed method was capable of accurate camera pose estimation, supported by evaluations on both synthetic and real scenes. Limitations related to convergence due to the non-convexity of the photometric loss were examined, leading to the introduction of a coarse-to-fine optimization strategy to address these challenges.

Additionally, an approach to improve localization outcomes was proposed by refining camera initialization through image retrieval techniques. By extending the image base with 3DGS-rendered camera frames, the overall performance of the localization pipeline was enhanced.

5.3 Final Remarks

In summary, this dissertation provides insights into state estimation in robotics, presenting methodologies that contribute to both theoretical understanding and practical application. The integration of time-continuous state estimation and visual localization techniques establishes a framework for future research in more complex environments. Further investigation into combining multiple sensor modalities may also enhance state estimation processes.

The findings aim to improve the performance and reliability of robotic systems, contributing to ongoing developments in the field of robotics.

Bibliography

P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.

Sameer Agarwal, Keir Mierle, and The Ceres Solver Team. Ceres Solver, 3 2022. URL `https://github.com/ceres-solver/ceres-solver`.

R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

Timothy D Barfoot. *State Estimation for Robotics*. Cambridge University Press, 2017.

Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.

José Luis Blanco-Claraco. A tutorial on $\mathbf{SE}(3)$ transformation parameterizations and on-manifold optimization. *CoRR*, abs/2103.15980, 2021. URL `https://arxiv.org/abs/2103.15980`.

Kazii Botashev. Source code of gsloc for chapter 4 of phd thesis of kazii botashev. `https://github.com/KaziiBotashev/Botashev_PhD_Thesis_GSLoc`, 2024. Accessed: 2024-12-04.

Simone Ceriani, Carlos Sánchez, Pierluigi Taddei, Erik Wolfart, and Vítor Sequeira. Pose interpolation slam for large maps using moving 3d sensors. In *IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 750–757, 2015.

Xu Chen, Zijian Dong, Jie Song, Andreas Geiger, and Otmar Hilliges. Category level object pose estimation via neural analysis-by-synthesis. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVI 16*, pages 139–156. Springer, 2020.

Alvaro Collet, Dmitry Berenson, Siddhartha S Srinivasa, and Dave Ferguson. Object recognition and full pose registration from a single image for robotic manipulation. In *2009 IEEE International Conference on Robotics and Automation*, pages 48–55. IEEE, 2009.

Alexander G. Cunningham, Vadim Indelman, and Frank Dellaert. Ddf-sam 2.0: Consistent distributed smoothing and mapping. *2013 IEEE International Conference on Robotics and Automation*, pages 5220–5227, 2013.

Frank Dellaert and Michael Kaess. Square root sam: Simultaneous localization and mapping via square root information smoothing. *Int. J. Rob. Res.*, 25(12): 1181–1203, dec 2006. ISSN 0278-3649. doi: 10.1177/0278364906072768. URL `https://doi.org/10.1177/0278364906072768`.

Pierre Dellenbach, Jean-Emmanuel Deschaud, Bastien Jacquet, and Franccois Goulette. Ct-icp: Real-time elastic lidar odometry with loop closure, 2021.

Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 224–236, 2018.

Jing Dong, Byron Boots, and Frank Dellaert. Sparse gaussian processes for continuous-time trajectory estimation on matrix lie groups, 2017. URL `https://arxiv.org/abs/1705.06020`.

Ethan Eade. Lie groups for computer vision. `https://ethaneade.com/lie_groups.pdf`, 2014. Accessed: 2024-12-04.

Ethan Eade. Lie groups for 2d and 3d transformations. `https://ethaneade.com/lie.pdf`, 2018. Accessed: 2024-12-04.

Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *European conference on computer vision*, pages 834–849. Springer, 2014.

Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE transactions on pattern analysis and machine intelligence*, 40(3):611–625, 2017.

Juan-Antonio Fernández-Madrigal and José Luis Blanco-Claraco. *Simultaneous Localization and Mapping for Mobile Robots: Introduction and Methods.* IGI Global, 2013. doi: 10.4018/978-1-4666-2104-6.

Gonzalo Ferrer. Introduction for robotics to rigid body transformations and differentiation over se(3). `https://github.com/g-ferrer/Perception-in-Robotics-2023/blob/master/L12/RBT.pdf`, 2021. Accessed: 2024-12-04.

Christian Forster, Matia Pizzoli, and Davide Scaramuzza. Svo: Fast semi-direct monocular visual odometry. In *2014 IEEE international conference on robotics and automation (ICRA)*, pages 15–22. IEEE, 2014.

Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361, 2012. doi: 10.1109/CVPR.2012.6248074.

Philippe-Antoine Gohard, Bertrand Vandeportaele, and Michel Devy. Spatiotemporal optimization for rolling shutter camera pose interpolation. In Ana Paula Cláudio, Dominique Bechmann, Paul Richard, Takehiko Yamaguchi, Lars Linsen, Alexandru Telea, Francisco Imai, and Alain Tremeau, editors, *Computer Vision, Imaging and Computer Graphics – Theory and Applications*, pages 154–175, Cham, 2019. Springer International Publishing. ISBN 978-3-030-12209-6.

Adrian Haarbach, Tolga Birdal, and Slobodan Ilic. Survey of higher order rigid body motion interpolation methods for keyframe animation and continuous-time trajectory estimation. In *2018 International Conference on 3D Vision (3DV)*, pages 381–389, 2018. doi: 10.1109/3DV.2018.00051.

Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Transactions on Graphics (ToG)*, 37(6):1–15, 2018.

Lionel Heng, Benjamin Choi, Zhaopeng Cui, Marcel Geppert, Sixing Hu, Benson Kuan, Peidong Liu, Rang Nguyen, Ye Chuan Yeo, Andreas Geiger, et al. Project autovision: Localization and 3d scene perception for an autonomous vehicle with a multi-camera system. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 4695–4702. IEEE, 2019.

Jianzhu Huai, Y. Zhuang, Qicheng Yuan, and Yukai Lin. Continuous-time spatiotemporal calibration of a rolling shutter camera-imu system. *IEEE Sensors Journal*, 22:7920–7930, 2021.

Michael Kaess, Ananth Ranganathan, and Frank Dellaert. isam: Incremental smoothing and mapping. *IEEE Transactions on Robotics*, 24(6):1365–1378, 2008. doi: 10.1109/TRO.2008.2006706.

Yermek Kapushev, Anastasia Kishkun, Gonzalo Ferrer, and Evgeny Burnaev. Random fourier features based slam. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6597–6602, 2021. doi: 10.1109/IROS51168.2021.9636819.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4):1–14, 2023.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, San Diego, CA, USA, 2015.

Rainer Kümmerle, Giorgio Grisetti, Hauke Strasdat, Kurt Konolige, and Wolfram Burgard. G2o: A general framework for graph optimization. In *2011 IEEE International Conference on Robotics and Automation*, pages 3607–3613, 2011. doi: 10.1109/ICRA.2011.5979949.

Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. Deepim: Deep iterative matching for 6d pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 683–698, 2018.

Simon Lynen, Torsten Sattler, Michael Bosse, Joel A Hesch, Marc Pollefeys, and Roland Siegwart. Get out of my lab: Large-scale, real-time visual-inertial localization. In *Robotics: Science and Systems*, volume 1, page 1, 2015.

Dominic Maggio, Marcus Abate, Jingnan Shi, Courtney Mario, and Luca Carlone. Loc-nerf: Monte carlo localization using neural radiance fields. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4018–4025. IEEE, 2023.

Hidenobu Matsuki, Riku Murai, Paul H. J. Kelly, and Andrew J. Davison. Gaussian Splatting SLAM. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

mrob. Mrob: Mobile robotics library. `https://github.com/g-ferrer/mrob`. Accessed: 2023-02-01.

Elias Mueggler, Guillermo Gallego, Henri Rebecq, and Davide Scaramuzza. Continuous-time visual-inertial odometry for event cameras. *IEEE Transactions on Robotics*, PP:1–16, 08 2018. doi: 10.1109/TRO.2018.2858287.

Chanoh Park, Peyman Moghadam, Soohwan Kim, Alberto Elfes, Clinton Fookes, and Sridha Sridharan. Elastic lidar fusion: Dense map-centric continuous-time slam. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1206–1213, 2018.

Chanoh Park, Peyman Moghadam, Jason L. Williams, Soohwan Kim, Sridha Sridharan, and Clinton Fookes. Elasticity meets continuous-time: Map-centric dense 3d lidar slam. *IEEE Transactions on Robotics*, 38(2):978–997, 2022. doi: 10.1109/TRO.2021.3096650.

Keunhong Park, Arsalan Mousavian, Yu Xiang, and Dieter Fox. Latentfusion: End-to-end differentiable reconstruction and rendering for unseen object pose estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10710–10719, 2020.

Olinde Rodrigues. Des lois géométriques qui régissent les déplacements d'un système solide dans l'espace, et de la variation des coordonnées provenant de ces déplacements considérés indépendants des causes qui peuvent les produire. *Journal de Mathématiques Pures et Appliquées*, 1840(5):380–440, 1840.

Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *CVPR*, 2019.

Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4938–4947, 2020.

Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016.

Joan Sola, Jeremie Deray, and Dinesh Atchuthan. A micro lie theory for state estimation in robotics. *arXiv preprint arXiv:1812.01537*, 2018.

Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, et al. The replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019.

Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J Davison. imap: Implicit mapping and positioning in real-time. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6229–6238, 2021.

Yuan Sun, Xuan Wang, Yunfan Zhang, Jie Zhang, Caigui Jiang, Yu Guo, and Fei Wang. icomma: Inverting 3d gaussians splatting for camera pose estimation via comparing and matching. *arXiv preprint arXiv:2312.09031*, 2023.

Hajime Taira, Masatoshi Okutomi, Torsten Sattler, Mircea Cimpoi, Marc Pollefeys, Josef Sivic, Tomas Pajdla, and Akihiko Torii. Inloc: Indoor visual localization with dense matching and view synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7199–7209, 2018.

Javier Tirado and Javier Civera. Jacobian computation for cumulative b-splines on se(3) and application to continuous-time object tracking. *IEEE Robotics and Automation Letters*, 7:1–8, 07 2022. doi: 10.1109/LRA.2022.3180427.

Xipeng Wang, Ryan J. Marcotte, Gonzalo Ferrer, and Edwin Olson. Apriisam: Real-time smoothing and mapping. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2486–2493, 2018.

Lin Yen-Chen, Pete Florence, Jonathan T Barron, Alberto Rodriguez, Phillip Isola, and Tsung-Yi Lin. inerf: Inverting neural radiance fields for pose estimation. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1323–1330. IEEE, 2021.

Zikang Yuan, Fengtian Lang, and Xin Yang. Sr-lio: Lidar-inertial odometry with sweep reconstruction, 2022. URL `https://arxiv.org/abs/2210.10424`.

Jon Zubizarreta, Iker Aguinaga, and Jose Maria Martinez Montiel. Direct sparse mapping. *IEEE Transactions on Robotics*, 36(4):1363–1370, 2020.

M. Zwicker, H. Pfister, J. van Baar, and M. Gross. Ewa splatting. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):223–238, 2002. doi: 10.1109/TVCG.2002.1021576.