Title Here

by

Author Name

Department of Mechanical Engineering
McGill University, Montreal
December, 2019

A thesis submitted to McGill University
in partial fulfillment of the requirements of the degree of
Master of Engineering

Supervisor:

Professor James Richard Forbes

Author Name
first.last@mail.mcgill.ca

To my family for their steadfast support and love.

Abstract

This thesis investigates the invariant extended Kalman filter (IEKF), a recently introduced method for nonlinear state estimation on matrix Lie groups. The IEKF is well suited to a particular class of systems, namely those with group-affine process models and invariant measurement models. In fact, when these conditions are met, the IEKF is a locally asymptotically convergent observer. However, in practice, process models are often not group-affine, and measurement models are often not invariant. The effect of removing these assumptions is investigated in this thesis. In particular, a 3D example is considered, with and without bias estimation. Estimating bias renders the process model not group affine. Then, a non-invariant measurement model is considered. Two different techniques are proposed to incorporate this measurement model into an IEKF, a standard approach using the non-invariant model and a novel approach in which the measurement is preprocessed to force the preprocessed measurement to be invariant. These practical extensions of the IEKF are tested in simulation to determine the effectiveness of the IEKF for more general state estimation problems. Lastly, batch estimation in the invariant framework is formulated. The problem of interest is the simultaneous localization and mapping (SLAM) problem. A general derivation of the SLAM problem on matrix Lie groups is presented. Invariant estimation theory is then leveraged. An inertial navigation example with bias estimation is then presented, with testing done in simulation.

Résumé

Cette thèse étudie le filter de Kalman invariant (IEKF), une méthode récemment introduite pour l'estimation d'état non linéaire sur des groupes de Lie matriciels. L'IEKF est bien adapté à une classe particulière de systèmes, plus précisément ceux dotés de fonctions affines et de fonctions d'observation invariantes. En fait, lorsque ces conditions sont satisfaites, l'IEKF est un observateur localement asymptotiquement convergent. Cependant, en situation pratique, ces conditions ne sont souvent pas satisfaites. Des scénarios ou ces conditions ne sont pas satisfaites sont étudié ici. En particulier, un exemple 3D est considéré, avec et sans estimation de biais dans le gyroscope. L'estimation du biais rend la fonction non-affinée. Ensuite, une fonction d'observation non-invariante est considéré. Deux techniques différentes sont proposées pour incorporer cette fonction d'observation dans un IEKF, une approche standard utilisant la fonction non-invariante et une nouvelle approche dans laquelle la mesure est prétraitée pour la forcer être invariante. Ces extensions pratiques du IEKF sont testées en simulation pour déterminer son efficacités pour des problèmes d'estimation d'état plus généraux. Enfin, une technique d'estimation par lot dans le cadre invariant est formulée. Un intérêt particulier est porté au problème de la localisation et de la cartographie simultanées (SLAM). Une dérivation générale du problème SLAM sur les groupes de Lie matriciels est présentée. La théorie de l'estimation invariante est ensuite mise à profit. Un exemple de navigation inertielle avec estimation du biais est ensuite présenté, avec des tests effectués en simulation.

Acknowledgements

I would like to thank Professor James Richard Forbes for his support over the last two years. His passion and extensive knowledge for his field of research makes him an ideal supervisor, and I was extremely fortunate to work with him during my Masters degree. I would also like to take this opportunity to thank the other members of the DECAR group, current and former. In particular, I would like to thank Alex Walsh for always being available to discuss ideas and answer questions.

I would also like to thank my friends and family for their support, and for always pushing me to new heights.

Preface

The contributions of this thesis that are original to the author's knowledge are as follows.

- Chapter **??**

– Solving the batch SLAM problem using a right-invariant framework while explicitly considering bias states.

All text, plots, figures and results in this thesis are produced by Jonathan Arsenault. The IEKF was originally introduced by Silvère Bonnabel and Axel Barrau in continuous time in [1], and in discrete time in [2].

Table of Contents

List of Figures

Figure

List of Tables

Table

List of Abbreviations

List of Symbols

$\lVert\cdot\rVert$	the Euclidian norm of a physical vector
$\mathbb{R}^n$	the vector space of real n-dimensional vectors
$\mathbb{R}^{m\times n}$	the space of real $m \times n$-dimensional matrices
$(\cdot)^{\mathsf{T}}$	transpose
$(\cdot)^{\times}$	cross operator for $\mathfrak{so}(3)$
$(\cdot)^{\wedge}$	operator mapping an element of $\mathbb{R}^d$ to $\mathfrak{g}$
$(\cdot)^{\vee}$	operator mapping an element of $\mathfrak{g}$ to $\mathbb{R}^d$
$\mathbf{0}$	zero matrix
$\mathbf{1}$	identity matrix
$\operatorname{diag}(\mathbf{M}_1, \ldots, \mathbf{M}_n)$	block diagonal matrix with $\mathbf{M}_1$, $\ldots$, $\mathbf{M}_n$ on diagonals, and zeros elsewhere
$\underrightarrow{\mathcal{F}}_i$	reference frame
$\underrightarrow{r}$	a physical vector
$\underrightarrow{r}^{zw}$	the position of point z relative to point w
$\underrightarrow{r}^{\bullet a}$	the time derivative of $\underrightarrow{r}$ with respect to $\underrightarrow{\mathcal{F}}_a$
$\underrightarrow{r}^{zw^{\bullet a}} = \underrightarrow{v}^{zw/a}$	velocity of point z relative to point w with respect to $\underrightarrow{\mathcal{F}}_a$
$\underrightarrow{\boldsymbol{\mathcal{F}}}_a$	a vectrix, that is a matrix of unit length physical vectors that form a basis for $\underrightarrow{\mathcal{F}}_a$, where $\underrightarrow{\boldsymbol{\mathcal{F}}}_a^{\mathsf{T}} = [\underrightarrow{a}^1 \;\; \underrightarrow{a}^2 \;\; \underrightarrow{a}^3]$
$\mathbf{r}_a$	the physical vector $\underrightarrow{r}$ resolved in $\underrightarrow{\mathcal{F}}_a$
$\mathbf{C}_{ab}$	a DCM parameterizing the attitude of $\underrightarrow{\mathcal{F}}_a$ relative to $\underrightarrow{\mathcal{F}}_b$

$\underrightarrow{\omega}^{ba}$	angular velocity of $\underrightarrow{\mathcal{F}}_b$ relative to $\underrightarrow{\mathcal{F}}_a$

Chapter 1

Introduction

The onboard computers of autonomous robots, such as unmanned aerial vehicles (UAV), mobile robots, or autonomous underwater vehicles (AUV), run navigation, guidance, and control algorithms that enable the robotic system to perform desired tasks. The navigation algorithm is responsible for estimating the states of the robot. The guidance algorithm considers planning the path the robot will take to complete its task. Lastly, the controller computes control inputs, such as forces and torques, to be applied so that the robot follows the desired trajectory. These three modules are of equal importance, and are intrinsically linked.

This thesis is focused on the navigation problem, also commonly called the state estimation problem. State estimation is the process of estimating the states of a system given noisy and biased sensor data. For example, an UAV must typically maintain a robust and accurate estimate of its position, velocity, and attitude in order to perform precision tasks, such as parcel delivery or surveillance. However, the sensors onboard UAVs are often of lower quality, to minimize the cost of the system, necessitating a state estimation algorithm that can reliably estimate the position, velocity, and attitude of the UAV given low-qulity senor data.

Several different state estimation techniques exist, each with their advantages and disadvantages. Roughly speaking, they can be separated into batch algorithms, which typically run offline, and sequential algorithms, which typically run in real time. Batch algorithms use sensor data over the entire trajectory to in turn provide an estimate of the states over the entire trajectory. Traditional batch algorithms include the (nonlinear) least-squares formulation [3, Sec. 4.3] and the forward-backward smoother [3, Sec. 3.2.2] and Rauch-Tung-Striebel smoother [3, Sec. 3.2.3]. Batch algorithms are especially useful when reconstructing scenes for metrology or photogrammetry applications, for example. In addition, simultaneous localization and mapping (SLAM) algorithms are often batch algorithms that do not run in real time.

In real-time applications, sequential state estimation methods are often preferred. The most commonly used algorithms for real-time state estimation are approximations of the Bayes filter [4], such as the Kalman filter, extended Kalman filter (EKF), or sigma-point Kalman filter. Other real-time state estimation methods that leverage concepts from the batch formulation, such as using a bundle of sensor data or iteration, include the sliding window filter [5], iterative extended Kalman filter [3, Sec. 4.2.5], and iterative sigma-point Kalman filter [3, Sec. 4.2.10].

In industry, the EKF is often the algorithm of choice, due to its relative simplicity and its track record of effectiveness. However, it does have its deficiencies. In this thesis, a variant of the EKF, the invariant extended Kalman filter (IEKF) is considered. For a review of the IEKF, see Chapter **??**. The main idea behind the invariant filtering framework is that certain problems (i.e., so-called "left-invariant" problems) do not explicitly depend on a particular inertial frame, and others (i.e., so-called "right-invariant" problems) do not explicitly depend on a particular body-fixed frame. Not all estimation problems fit the invariant filtering framework, but when an estimation problem does, extremely appealing properties appear.

1.1 Thesis Objective

The objective of this thesis is to determine how the invariant filtering framework can be used to improve existing state estimation methods. In particular, the contribution of this thesis is an overview of practical considerations of the IEKF and an extension of the invariant estimation theory to the SLAM problem posed in a batch framework.

Another contribution of this thesis is to thoroughly summarize the theory behind the IEKF. This includes some proofs that are missing from the literature. A major contribution of this thesis is to compare the various error definitions that can be used in solving the SLAM problem. This includes a general formulation for performing SLAM when the state can be formulated as an element of a matrix Lie group. Modifying the error definitions leads to Jacobians that may depend less, or not at all, on the state estimate. Lastly, this thesis provides a thorough analysis of the practical implications of the IEKF. The theory behind the IEKF is sound, but the assumptions made often do not hold in practice. Of note, a novel method of using the IEKF in conjunction with a stereo camera is presented.

1.2 Thesis Overview

This thesis is structured as follows.

Chapter 2 summarizes mathematical concepts and notation that are used throughout this thesis.

Chapter **??** outlines the IEKF. The relevant theorems and proofs are presented in continuous and discrete-time. The left-invariant extended Kalman filter and right-invariant extended Kalman filter are then detailed.

In Chapter **??**, several examples of the IEKF are presented to illustrate how to practically implement an IEKF and to compare its performance to that of a standard multiplicative extended Kalman filter (MEKF).

In Chapter **??**, a solution to the SLAM problem in the invariant framework is presented. Simulation results are shown comparing the novel formulation to more traditional batch-based solutions to the SLAM problem.

This thesis is concluded in Chapter 3, where a summary of the findings are presented, along with recommended future work.

Chapter 2

Preliminaries

2.1 Matrix Lie Groups

In many robotics applications, the states are an element of a matrix Lie groups. Any navigation, guidance, or control algorithm should explicitly accommodate the matrix Lie group nature of the states. As such, matrix Lie group properties and theory will be introduced next.

2.1.1 Overview

This summary of matrix Lie group theory is based on Section 2 of [6]. A matrix Lie group $\mathcal{G}$, is composed of invertible $n \times n$ matrices that is closed under matrix multiplication. The matrix Lie algebra associated with $\mathcal{G}$, denoted $\mathfrak{g}$, is the tangent space around the identity of $\mathcal{G}$, denoted $T_{\mathbf{1}}\mathcal{G}$. The tangent space of $\mathcal{G}$ at any $\mathbf{X} \in \mathcal{G}$ is denoted $T_{\mathbf{X}}\mathcal{G}$. The matrix Lie algebra is a vector space closed under the operation of the matrix Lie bracket defined $[\mathbf{A}, \mathbf{B}] = \mathbf{AB} - \mathbf{BA},\ \forall \mathbf{A}, \mathbf{B} \in \mathfrak{g}$. Furthermore, $\mathbf{XAX}^{-1} \in \mathfrak{g},\ \forall \mathbf{X} \in \mathcal{G},\ \forall \mathbf{A} \in \mathfrak{g}$. Any $\mathbf{A} \in \mathfrak{g}$ can be written as $\mathbf{A} = \boldsymbol{\xi}^\wedge = \sum_{i=1}^{n} \xi_i \mathbf{B}_i$, where $\{\mathbf{B}_1, \ldots, \mathbf{B}_n\}$ is a basis for $\mathfrak{g}$, also known as the generators, and $\mathbf{A} = [\xi_1, \ldots, \xi_n]^\mathsf{T} \in \mathbb{R}^n$ is the column matrix of coefficients associated with $\mathbf{A}$. Alternatively, $\mathbf{A}^\vee = \boldsymbol{\xi}$.

The exponential map takes elements in the Lie algebra and maps them to the Lie group. For matrix Lie groups, the exponential map is simply the matrix exponential. The inverse of the matrix exponential, the matrix logarithm, is also defined and maps elements of the matrix Lie group to the matrix Lie algebra. In more detail,

$$\mathbf{X} = \exp\left(\boldsymbol{\xi}^\wedge\right)$$

and

$$\boldsymbol{\xi}^{\wedge} = \log(\mathbf{X})$$

where $\mathbf{X} \in \mathcal{G}$ and $\boldsymbol{\xi}^{\wedge} \in \mathfrak{g}$.

The matrix representation of the adjoint operator is used throughout this thesis. It is not unique, as it depends on the parametrization. Denoting the adjoint representation of $\mathbf{X}$ as $\mathrm{Ad}(\mathbf{X})$, then $(\mathrm{Ad}(\mathbf{X})\boldsymbol{\xi})^{\wedge} = \mathbf{X}\boldsymbol{\xi}^{\wedge}\mathbf{X}^{-1}$. This leads to the useful identity

$$\mathbf{X}\exp(\boldsymbol{\xi}^{\wedge})\mathbf{X}^{-1} = \exp\left((\mathrm{Ad}(\mathbf{X})\boldsymbol{\xi})^{\wedge}\right). \tag{2.1}$$

The adjoint representation of an element of the matrix Lie algebra can also be defined [1, 7]. Given $\boldsymbol{\xi}^{\wedge}, \boldsymbol{\zeta}^{\wedge} \in \mathfrak{g}$, the adjoint matrix satisfies $\mathrm{ad}(\boldsymbol{\zeta})\boldsymbol{\xi} = -\mathrm{ad}(\boldsymbol{\xi})\boldsymbol{\zeta}$ and

$$\boldsymbol{\xi}^{\wedge}\boldsymbol{\zeta}^{\wedge} - \boldsymbol{\zeta}^{\wedge}\boldsymbol{\xi}^{\wedge} = (-\mathrm{ad}(\boldsymbol{\zeta})\boldsymbol{\xi})^{\wedge}. \tag{2.2}$$

2.1.2 Uncertainty Representations

In standard linear vector spaces, uncertainty is simply additive, such that $\mathbf{x} = \bar{\mathbf{x}} + \delta\mathbf{x}$, where $\delta\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$. However, this is not applicable to matrix Lie groups, as they are not closed under addition. Rather, a multiplicative uncertainty must be used [8]. This leads to two distinct options, namely

$$\mathbf{X} = \bar{\mathbf{X}}\exp\left(\delta\boldsymbol{\xi}^{\wedge}\right), \tag{2.3}$$

$$\mathbf{X} = \exp\left(\delta\boldsymbol{\xi}^{\wedge}\right)\bar{\mathbf{X}}, \tag{2.4}$$

where $\delta\boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$. Note that $\mathbf{X}$ is not normally distributed. Two additional uncertainty definitions can also be defined. They are

$$\mathbf{X} = \bar{\mathbf{X}}\exp\left(-\delta\boldsymbol{\xi}^{\wedge}\right), \tag{2.5}$$

$$\mathbf{X} = \exp\left(-\delta\boldsymbol{\xi}^{\wedge}\right)\bar{\mathbf{X}}, \tag{2.6}$$

defined as the left-invariant and right-invariant uncertainty representations, respectively. They are named as such as they are consistent with left and right-invariant error definitions, which are introduced in Chapter **??**.

2.1.3 The Baker-Campbell-Hausdorff Formula

The Baker-Campbell-Hausdorff (BCH) formula is the solution to the equation [3]

$$\mathbf{z}^\wedge = \log\left(\exp\left(\mathbf{a}^\wedge\right)\exp\left(\mathbf{b}^\wedge\right)\right).$$

The detailed solution is available in [3, pp. 230-232]. Herein only a first-order approximation is needed, that being

$$\log\left(\exp\left(\mathbf{a}^\wedge\right)\exp\left(\mathbf{b}^\wedge\right)\right) = \mathbf{a}^\wedge + \mathbf{b}^\wedge.$$

This is exact in the case that $[\mathbf{a}^\wedge, \mathbf{b}^\wedge] = \mathbf{0}$.

2.1.4 Linearization

Any element of a matrix Lie group can be expressed using the exponential map, which is in fact the matrix exponential,

$$\mathbf{X} = \exp\left(\boldsymbol{\xi}^\wedge\right).$$

The matrix exponential itself is defined by a power series,

$$\begin{aligned}\exp\left(\boldsymbol{\xi}^\wedge\right) &= \sum_{k=0}^{\infty}\frac{1}{k!}(\boldsymbol{\xi}^\wedge)^k \\ &= \mathbf{1} + \boldsymbol{\xi}^\wedge + \frac{(\boldsymbol{\xi}^\wedge)^2}{2} + \frac{(\boldsymbol{\xi}^\wedge)^3}{6} + \dots\end{aligned}$$

Now, consider the case where $\boldsymbol{\xi}$ can be considered small. This small element of $\mathbb{R}^d$ is denoted $\delta\boldsymbol{\xi}$ and $\delta\mathbf{X} = \exp\left(\delta\boldsymbol{\xi}^\wedge\right)$. As $\delta\boldsymbol{\xi}$ is already considered small, it is common to assume that terms of order $\mathcal{O}(\|\delta\boldsymbol{\xi}\|^2)$ can be neglected, leading to the approximation

$$\delta\mathbf{X} \approx \mathbf{1} + \delta\boldsymbol{\xi}^\wedge.$$

Thus, the uncertainty representations (2.3) and (2.4) can be approximated as

$$\begin{aligned}\mathbf{X} &= \bar{\mathbf{X}}(\mathbf{1} + \delta\boldsymbol{\xi}^\wedge), \\ \mathbf{X} &= (\mathbf{1} + \delta\boldsymbol{\xi}^\wedge)\bar{\mathbf{X}},\end{aligned}$$

respectively. Similarly for (2.5) and (2.6),

$$\begin{aligned}\mathbf{X} &= \bar{\mathbf{X}}(\mathbf{1} - \delta\boldsymbol{\xi}^\wedge), \\ \mathbf{X} &= (\mathbf{1} - \delta\boldsymbol{\xi}^\wedge)\bar{\mathbf{X}}.\end{aligned}$$

2.2 The Special Orthogonal Group $SO(3)$

The properties of $SO(3)$ are from [3, Ch. 7]. Three dimensional rotations can be represented by the special orthogonal group $SO(3)$,

$$SO(3) = \left\{ \mathbf{C} \in \mathbb{R}^{3\times 3} \mid \mathbf{C}^{\mathsf{T}}\mathbf{C} = \mathbf{1}, \det \mathbf{C} = +1 \right\}.$$

The matrix $\mathbf{C}$ is known as a direction cosine matrix (DCM). $SO(3)$ has three degrees of freedom for rotation. The Lie algebra associated with $SO(3)$ is

$$\mathfrak{so}(3) = \left\{ \boldsymbol{\phi}^{\times} \in \mathbb{R}^{3\times 3} \mid \boldsymbol{\phi} \in \mathbb{R}^3 \right\},$$

where $\boldsymbol{\phi}^{\times}$ is the skew-symmetric representation of $\boldsymbol{\phi}$,

$$\boldsymbol{\phi}^{\times} = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{bmatrix}^{\times} = \begin{bmatrix} 0 & -\phi_3 & \phi_2 \\ \phi_3 & 0 & -\phi_1 \\ -\phi_2 & \phi_1 & 0 \end{bmatrix}.$$

The adjoint representation of an element of $SO(3)$ is identically that element, $\mathrm{Ad}(\mathbf{C}) = \mathbf{C}$. Similarly, the adjoint representation of an element of $\mathfrak{so}(3)$ is identical to that element, $\mathrm{ad}(\boldsymbol{\phi}) = \boldsymbol{\phi}^{\times}$. The closed form of the exponential map from $\mathfrak{so}(3)$ to $SO(3)$ is known as the Rodrigues formula,

$$\exp(\boldsymbol{\phi}^{\times}) = \cos\phi \mathbf{1} + (1 - \cos\phi)\mathbf{a}\mathbf{a}^{\mathsf{T}} + \sin\phi \mathbf{a}^{\times},$$

where $\phi = \|\boldsymbol{\phi}\|$ and $\mathbf{a} = \boldsymbol{\phi}/\phi$. The logarithmic map from $SO(3)$ to $\mathfrak{so}(3)$ is

$$\log(\mathbf{C}) = (\mathbf{a}\phi)^{\times},$$

where the angle ϕ is given by

$$\phi = \cos^{-1}\left(\frac{\mathrm{tr}(\mathbf{C}) - 1}{2}\right) + 2\pi m$$

and the axis $\mathbf{a}$ is

$$\mathbf{a} = \frac{1}{2\sin(\phi)} \begin{bmatrix} \mathbf{C}_{2,3} - \mathbf{C}_{3,2} \\ \mathbf{C}_{3,1} - \mathbf{C}_{1,3} \\ \mathbf{C}_{1,2} - \mathbf{C}_{2,1} \end{bmatrix}.$$

2.3 The Special Euclidean Group $SE(3)$

Poses can be represented by the special euclidean group $SE(3)$ [3, Ch. 7],

$$SE(3) = \left\{ \mathbf{T} = \begin{bmatrix} \mathbf{C} & \mathbf{r} \\ \mathbf{0} & 1 \end{bmatrix} \in \mathbb{R}^{4\times4} \;\middle|\; \mathbf{C} \in SO(3), \mathbf{r} \in \mathbb{R}^3 \right\}.$$

$SE(3)$ has three degrees of freedom for rotation and three for translation, for a total of six. The inverse of $\mathbf{T}$ is defined as

$$\mathbf{T}^{-1} = \begin{bmatrix} \mathbf{C}^\mathsf{T} & -\mathbf{C}^\mathsf{T}\mathbf{r} \\ \mathbf{0} & 1 \end{bmatrix}.$$

The matrix Lie algebra associated with $SE(3)$ is

$$\mathfrak{se}(3) = \left\{ \boldsymbol{\Xi} = \boldsymbol{\xi}^\wedge \in \mathbb{R}^{4\times4} \mid \boldsymbol{\xi} \in \mathbb{R}^6 \right\},$$

where

$$\boldsymbol{\xi}^\wedge = \begin{bmatrix} \boldsymbol{\xi}^\phi \\ \boldsymbol{\xi}^\mathrm{r} \end{bmatrix}^\wedge = \begin{bmatrix} \boldsymbol{\xi}^{\phi\times} & \boldsymbol{\xi}^\mathrm{r} \\ \mathbf{0} & 0 \end{bmatrix} \in \mathbb{R}^{4\times4}, \;\; \boldsymbol{\xi}^\phi, \boldsymbol{\xi}^\mathrm{r} \in \mathbb{R}^3.$$

Note that, in [3], $\boldsymbol{\xi}$ is defined in the opposite order, such that $\boldsymbol{\xi}^\phi$ is below $\boldsymbol{\xi}^\mathrm{r}$. The convention used here is adopted from [1]. The exponential map from $\mathfrak{se}(3)$ to $SE(3)$ is

$$\exp\left(\boldsymbol{\xi}^\wedge\right) = \begin{bmatrix} \exp_{SO(3)}\left(\boldsymbol{\xi}^{\phi\times}\right) & \mathbf{J}\boldsymbol{\xi}^\mathrm{r} \\ \mathbf{0} & 1 \end{bmatrix},$$

where

$$\mathbf{J} = \frac{\sin\phi}{\phi}\mathbf{1} + \left(1 - \frac{\sin\phi}{\phi}\right)\mathbf{a}\mathbf{a}^\mathsf{T} + \frac{1-\cos\phi}{\phi}\mathbf{a}^\times, \tag{2.7}$$

where $\phi = \left\|\boldsymbol{\xi}^\phi\right\|$ and $\mathbf{a} = \boldsymbol{\xi}^\phi/\phi$. The logarithmic map from $SE(3)$ to $\mathfrak{se}(3)$ is

$$\log(\mathbf{T}) = \begin{bmatrix} \log_{SO(3)}(\mathbf{C}) & \mathbf{J}^{-1}\mathbf{r} \\ \mathbf{0} & 0 \end{bmatrix},$$

where

$$\mathbf{J}^{-1} = \frac{\phi}{2}\cot\frac{\phi}{2}\mathbf{1} + \left(1 - \frac{\phi}{2}\cot\frac{\phi}{2}\right)\mathbf{a}\mathbf{a}^\mathsf{T} - \frac{\phi}{2}\mathbf{a}^\times. \tag{2.8}$$

The adjoint representation of an element of $SE(3)$ is

$$\mathrm{Ad}(\mathbf{T}) = \begin{bmatrix} \mathbf{C} & \mathbf{0} \\ \mathbf{r}^\times\mathbf{C} & \mathbf{C} \end{bmatrix} \in \mathbb{R}^{6\times6}.$$

The inverse of $\mathrm{Ad}(\mathbf{T})$ is

$$(\mathrm{Ad}(\mathbf{T}))^{-1} = \mathrm{Ad}\left(\mathbf{T}^{-1}\right) = \begin{bmatrix} \mathbf{C}^{\mathsf{T}} & \mathbf{0} \\ -\mathbf{C}^{\mathsf{T}}\mathbf{r}^{\times} & \mathbf{C}^{\mathsf{T}} \end{bmatrix} \in \mathbb{R}^{6\times 6}.$$

The adjoint representation of an element of $\mathfrak{se}(3)$ is

$$\mathrm{ad}(\boldsymbol{\xi}) = \begin{bmatrix} \boldsymbol{\xi}^{\phi\times} & \mathbf{0} \\ \boldsymbol{\xi}^{\mathrm{r}\times} & \boldsymbol{\xi}^{\phi\times} \end{bmatrix}.$$

2.4 The Group of Double Direct Isometries $SE_2(3)$

Introduced in [9] and explored in detail in [10], $SE_2(3)$, the group of double direct isometries, is

$$SE_2(3) = \left\{ \mathbf{T} = \begin{bmatrix} \mathbf{C} & \mathbf{v} & \mathbf{r} \\ \mathbf{0} & 1 & 0 \\ \mathbf{0} & 0 & 1 \end{bmatrix} \in \mathbb{R}^{5\times 5} \;\middle|\; \mathbf{C} \in SO(3), \mathbf{v}, \mathbf{r} \in \mathbb{R}^3 \right\}.$$

The matrix Lie algebra associated with $SE_2(3)$ is

$$\mathfrak{se}_2(3) = \left\{ \boldsymbol{\Xi} = \boldsymbol{\xi}^{\wedge} \in \mathbb{R}^{5\times 5} \mid \boldsymbol{\xi} \in \mathbb{R}^9 \right\},$$

where

$$\boldsymbol{\xi}^{\wedge} = \begin{bmatrix} \boldsymbol{\xi}^{\phi} \\ \boldsymbol{\xi}^{\mathrm{v}} \\ \boldsymbol{\xi}^{\mathrm{r}} \end{bmatrix}^{\wedge} = \begin{bmatrix} \boldsymbol{\xi}^{\phi\times} & \boldsymbol{\xi}^{\mathrm{v}} & \boldsymbol{\xi}^{\mathrm{r}} \\ \mathbf{0} & 0 & 0 \\ \mathbf{0} & 0 & 0 \end{bmatrix}.$$

The exponential map from $\mathfrak{se}_2(3)$ to $SE_2(3)$ is

$$\exp\left(\boldsymbol{\xi}^{\wedge}\right) = \begin{bmatrix} \exp_{SO(3)}\left(\boldsymbol{\xi}^{\phi\times}\right) & \mathbf{J}\boldsymbol{\xi}^{\mathrm{v}} & \mathbf{J}\boldsymbol{\xi}^{\mathrm{r}} \\ \mathbf{0} & 1 & 0 \\ \mathbf{0} & 0 & 1 \end{bmatrix},$$

where $\mathbf{J}$ is given by (2.7). The logarithmic map from $SE_2(3)$ to $\mathfrak{se}_2(3)$ is

$$\log(\mathbf{T}) = \begin{bmatrix} \log_{SO(3)}(\mathbf{C}) & \mathbf{J}^{-1}\mathbf{v} & \mathbf{J}^{-1}\mathbf{r} \\ \mathbf{0} & 0 & 0 \\ \mathbf{0} & 0 & 0 \end{bmatrix},$$

where $\mathbf{J}^{-1}$ is given by (2.8). The adjoint representation of an element of $SE_2(3)$ is

$$\mathrm{Ad}(\mathbf{T}) = \begin{bmatrix} \mathbf{C} & \mathbf{0} & \mathbf{0} \\ \mathbf{v}^{\times}\mathbf{C} & \mathbf{C} & \mathbf{0} \\ \mathbf{r}^{\times}\mathbf{C} & \mathbf{0} & \mathbf{C} \end{bmatrix}.$$

The adjoint representation of an element of $\mathfrak{se}_2(3)$ is

$$\mathrm{ad}(\boldsymbol{\xi}) = \begin{bmatrix} \boldsymbol{\xi}^{\phi^{\times}} & \mathbf{0} & \mathbf{0} \\ \boldsymbol{\xi}^{\mathrm{v}^{\times}} & \boldsymbol{\xi}^{\phi^{\times}} & \mathbf{0} \\ \boldsymbol{\xi}^{\mathrm{r}^{\times}} & \mathbf{0} & \boldsymbol{\xi}^{\phi^{\times}} \end{bmatrix}.$$

2.5 Geometry

A reference frame $\underrightarrow{\mathcal{F}}_a$ is defined by three physical basis vectors $\underrightarrow{a}^1$, $\underrightarrow{a}^2$, and $\underrightarrow{a}^3$. In particular, the vectrix $\underrightarrow{\boldsymbol{\mathcal{F}}}_a$ can be defined as [11]

$$\underrightarrow{\boldsymbol{\mathcal{F}}}_a = \begin{bmatrix} \underrightarrow{a}^1 \\ \underrightarrow{a}^2 \\ \underrightarrow{a}^3 \end{bmatrix}.$$

A physical vector $\underrightarrow{u}$ can then be written as

$$\underrightarrow{u} = \underrightarrow{\boldsymbol{\mathcal{F}}}_a^{\mathsf{T}} \mathbf{u}_a,$$

where $\underrightarrow{u} \in \mathbb{P}$ and $\mathbf{u}_a \in \mathbb{R}^3$ is the physical vector $\underrightarrow{u}$ resolved in $\underrightarrow{\mathcal{F}}_a$. The orientation of $\underrightarrow{\mathcal{F}}_b$ relative to $\underrightarrow{\mathcal{F}}_a$ is given by a DCM $\mathbf{C}_{ab} \in SO(3)$. The relationship between $\underrightarrow{u}$ resolved in $\mathcal{F}_a$, $\mathbf{u}_a$, and $\underrightarrow{u}$ resolved in $\mathcal{F}_b$, $\mathbf{u}_b$, is $\mathbf{u}_a = \mathbf{C}_{ab}\mathbf{u}_b$.

2.6 Kinematics

The position of point z relative to point w is described by the physical vector $\underrightarrow{r}^{zw}$. The rate of change of $\underrightarrow{r}^{zw}$ with respect to $\underrightarrow{\mathcal{F}}_a$ is denoted $\underrightarrow{r}^{zw^{\bullet a}} = \underrightarrow{v}^{zw/a}$. Similarly, $\underrightarrow{r}^{zw^{\bullet a \bullet a}} = \underrightarrow{v}^{zw/a^{\bullet a}} = \underrightarrow{a}^{zw/a/a}$. These physical vectors can all then be resolved in a frame, as appropriate. Poisson's equation is

$$\dot{\mathbf{C}}_{ab} = \mathbf{C}_{ab}\boldsymbol{\omega}_b^{ba^{\times}},$$

where $\boldsymbol{\omega}_b^{ba}$ is the angular velocity of $\underrightarrow{\mathcal{F}}_b$ relative to $\underrightarrow{\mathcal{F}}_a$ resolved in $\underrightarrow{\mathcal{F}}_b$. When discretized, Poisson's equation becomes

$$\mathbf{C}_{ab_k} = \mathbf{C}_{ab_{k-1}} \exp\left(\left(T\boldsymbol{\omega}_{b_{k-1}}^{b_{k-1}a}\right)^{\times}\right),$$

where $T = t_k - t_{k-1}$.

2.7 Optimization

Consider a standard optimization problem

$$\mathbf{x}^{\star} = \operatorname*{argmin}_{\mathbf{x}\in\mathbb{R}^n} J(\mathbf{x}),$$

where $\mathbf{x}^{\star}$ is the minimizing solution of the cost function $J(\mathbf{x})$. Two types of optimization problems are of interest here, namely linear and nonlinear least squares.

2.7.1 Linear Least Squares

Consider a linear system that can be written

$$\mathbf{A}\mathbf{x} = \mathbf{b},$$

where $\mathbf{A} \in \mathbb{R}^{m\times n}$, $\mathbf{x} \in \mathbb{R}^n$, and $\mathbf{b} \in \mathbb{R}^m$. Define the error to be $\boldsymbol{\rho}(\mathbf{x}) = \mathbf{A}\mathbf{x} - \mathbf{b}$. The objective function to be minimized is

$$J(\mathbf{x}) = \frac{1}{2}\boldsymbol{\rho}(\mathbf{x})^{\mathsf{T}}\boldsymbol{\rho}(\mathbf{x}). \tag{2.9}$$

The solution minimizing (2.9) is

$$\mathbf{x}^{\star} = \left(\mathbf{A}^{\mathsf{T}}\mathbf{A}\right)^{-1}\left(\mathbf{A}^{\mathsf{T}}\mathbf{b}\right).$$

Explicitly computing the inverse of $\mathbf{A}^{\mathsf{T}}\mathbf{A}$ can be computationally costly. To alleviate these costs, a Cholesky factorization can be used to decompose $\mathbf{A}^{\mathsf{T}}\mathbf{A}$. This leads to

$$\mathbf{A}^{\mathsf{T}}\mathbf{A} = \mathbf{L}\mathbf{L}^{\mathsf{T}},$$

where $\mathbf{L}$ is lower triangular. The linear least squares problem can then be rewritten as

$$\mathbf{L}\mathbf{L}^{\mathsf{T}}\mathbf{x}^{\star} = \mathbf{A}^{\mathsf{T}}\mathbf{b}.$$

Letting $\mathbf{L}^{\mathsf{T}}\mathbf{x}^{\star} = \mathbf{z}$, it is possible to solve

$$\mathbf{L}^{\mathsf{T}}\mathbf{z} = \mathbf{A}^{\mathsf{T}}\mathbf{b}$$

for $\mathbf{z}$ via forwards substitution. The minimizing solution $\mathbf{x}^{\star}$ is then found by solving $\mathbf{L}\mathbf{x}^{\star} = \mathbf{z}$ using backward substitution.

2.7.2 Nonlinear Least Squares

This section is based on [3]. In practice, the error function $\boldsymbol{\rho}(\mathbf{x})$ is often not linear, and is instead some nonlinear function of $\mathbf{x}$. In this case, a Taylor series expansion of the cost function is used to linearize the problem. Consider a function $f(\cdot) : \mathbb{R}^n \to \mathbb{R}^m$. A Taylor series expansion of f is

$$f(\mathbf{x}^{\mathrm{op}} + \delta\mathbf{x}) = f(\mathbf{x}^{\mathrm{op}}) + \left[\left. \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} \right|_{\mathbf{x}=\mathbf{x}^{\mathrm{op}}} \right] \delta\mathbf{x} + \frac{1}{2}\delta\mathbf{x}^{\mathsf{T}} \left[\left. \frac{\partial^2 f(\mathbf{x})}{\partial \mathbf{x} \partial \mathbf{x}^{\mathsf{T}}} \right|_{\mathbf{x}=\mathbf{x}^{\mathrm{op}}} \right] \delta\mathbf{x} + \mathcal{O}\left(\left\| \delta\mathbf{x}^3 \right\| \right),$$

where $\mathbf{x}^{\mathrm{op}}$ is the operating point. The Jacobian of f is

$$\nabla f(\mathbf{x}^{\mathrm{op}}) = \left. \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} \right|_{\mathbf{x}=\mathbf{x}^{\mathrm{op}}}$$

and the Hessian of f is

$$\nabla^2 f(\mathbf{x}^{\mathrm{op}}) = \left. \frac{\partial^2 f(\mathbf{x})}{\partial \mathbf{x} \partial \mathbf{x}^{\mathsf{T}}} \right|_{\mathbf{x}=\mathbf{x}^{\mathrm{op}}}.$$

2.7.2.1 Newton's Method [3]

Taking a second-order Taylor series expansion of the cost function $J(\mathbf{x})$ yields

$$J(\mathbf{x}^{\mathrm{op}} + \delta\mathbf{x}) = J(\mathbf{x}^{\mathrm{op}}) + \nabla J(\mathbf{x}^{\mathrm{op}})\delta\mathbf{x} + \frac{1}{2}\delta\mathbf{x}^{\mathsf{T}}\nabla^2 J(\mathbf{x}^{\mathrm{op}})\delta\mathbf{x}. \tag{2.10}$$

To minimize the cost function, the derivative of (2.10) with respect to $\delta\mathbf{x}$ is computed and set to zero,

$$\begin{aligned} \nabla J(\mathbf{x}^{\mathrm{op}}) + \delta\mathbf{x}^{\star\mathsf{T}}\nabla^2 J(\mathbf{x}^{\mathrm{op}}) &= 0, \\ \nabla^2 J(\mathbf{x}^{\mathrm{op}})\delta\mathbf{x}^{\star} &= -\nabla J(\mathbf{x}^{\mathrm{op}}). \end{aligned} \tag{2.11}$$

Assuming the Hessian is positive definite and therefore invertible, (2.11) can be solved. The operating point is then updated,

$$\mathbf{x}^{\mathrm{op}} \leftarrow \mathbf{x}^{\mathrm{op}} + \delta\mathbf{x}^{\star}.$$

To improve the performance of Newton's method, it is common to multiply the update by a step length $\alpha > 0$, leading to

$$\mathbf{x}^{\text{op}} \leftarrow \mathbf{x}^{\text{op}} + \alpha \delta \mathbf{x}^{\star}.$$

A backtracking procedure is used to find the step length [12, p. 37]. This method is summarized in Algorithm 2.1. The parameter c is typically chosen to be small (approximately 10^{-4}) and ρ is tuned to a desired convergence rate. In Newton methods, $\bar{\alpha} = 1$.

Algorithm 2.1

Select $\bar{\alpha} > 0$, $\rho \in (0, 1)$, $c \in (0, 1)$
Set $\alpha \leftarrow \bar{\alpha}$
while $J(\mathbf{x}^{\text{op}} + \alpha \delta \mathbf{x}^{\star}) > J(\mathbf{x}^{\text{op}}) + c\alpha \nabla J(\mathbf{x}^{\text{op}})^{\mathsf{T}} \delta \mathbf{x}^{\star}$ **do**
    $\alpha \leftarrow \rho \alpha$
end while

2.7.2.2 Gauss-Newton Method

The Gauss-Newton method [3] is applicable when the cost function is a nonlinear least squares cost function,

$$J(\mathbf{x}) = \frac{1}{2} \boldsymbol{\rho}(\mathbf{x})^{\mathsf{T}} \boldsymbol{\rho}(\mathbf{x}).$$

A first-order Taylor series expansion yields

$$\begin{aligned} J(\mathbf{x}^{\text{op}} + \delta \mathbf{x}) &= \frac{1}{2} \boldsymbol{\rho}(\mathbf{x}^{\text{op}} + \delta \mathbf{x})^{\mathsf{T}} \boldsymbol{\rho}(\mathbf{x}^{\text{op}} + \delta \mathbf{x}), \\ 2J(\mathbf{x}^{\text{op}} + \delta \mathbf{x}) &= \left(\boldsymbol{\rho}(\mathbf{x}^{\text{op}}) + \nabla J(\mathbf{x}^{\text{op}}) \delta \mathbf{x}\right)^{\mathsf{T}} \left(\boldsymbol{\rho}(\mathbf{x}^{\text{op}}) + \nabla J(\mathbf{x}^{\text{op}}) \delta \mathbf{x}\right) \\ &= \boldsymbol{\rho}(\mathbf{x}^{\text{op}})^{\mathsf{T}} \boldsymbol{\rho}(\mathbf{x}^{\text{op}}) + \boldsymbol{\rho}(\mathbf{x}^{\text{op}})^{\mathsf{T}} \nabla J(\mathbf{x}^{\text{op}}) \delta \mathbf{x} + \left(\nabla J(\mathbf{x}^{\text{op}}) \delta \mathbf{x}\right)^{\mathsf{T}} \boldsymbol{\rho}(\mathbf{x}^{\text{op}}) + \\ &\quad \left(\nabla J(\mathbf{x}^{\text{op}}) \delta \mathbf{x}\right)^{\mathsf{T}} \nabla J(\mathbf{x}^{\text{op}}) \delta \mathbf{x}. \end{aligned} \tag{2.12}$$

Once again, the derivative of (2.12) is taken with respect to $\delta \mathbf{x}$ and set to zero, yielding

$$2 \frac{\partial J(\mathbf{x}^{\text{op}} + \delta \mathbf{x})}{\partial \delta \mathbf{x}} = 2 \boldsymbol{\rho}(\mathbf{x}^{\text{op}})^{\mathsf{T}} \nabla J(\mathbf{x}^{\text{op}}) + 2 \delta \mathbf{x}^{\mathsf{T}} \nabla J(\mathbf{x}^{\text{op}})^{\mathsf{T}} \nabla J(\mathbf{x}^{\text{op}}) = 0,$$

leading to

$$\nabla J(\mathbf{x}^{\text{op}})^{\mathsf{T}} \nabla J(\mathbf{x}^{\text{op}}) \delta \mathbf{x} = -\nabla J(\mathbf{x}^{\text{op}})^{\mathsf{T}} \boldsymbol{\rho}(\mathbf{x}^{\text{op}}).$$

This is often written

$$\left(\mathbf{H}\mathbf{H}^{\mathsf{T}}\right) \delta \mathbf{x} = -\mathbf{H}^{\mathsf{T}} \boldsymbol{\rho}(\mathbf{x}^{\text{op}}),$$

where

$$\mathbf{H} = \nabla J(\mathbf{x}^{\mathrm{op}}).$$

This is iterated until convergence using the technique described in Section 2.7.2.1 .

2.7.2.3 Levenberg-Marquardt Method

A small modification to the Gauss-Newton method leads to the Levenberg-Marquardt method [13–15],

$$\left(\mathbf{H}\mathbf{H}^{\mathsf{T}} + \lambda \mathrm{diag}(\mathbf{H}\mathbf{H}^{\mathsf{T}})\right) \delta\mathbf{x} = -\mathbf{H}^{\mathsf{T}} \boldsymbol{\rho}(\mathbf{x}^{\mathrm{op}}),$$

where $\lambda \geq 0$ is a damping factor. This damping factor allows the condition of the Hessian to be improved. Multiplying by the diagonal elements of the Hessian allows for the scaling of the problem to be preserved. The parameter λ can be found using a technique such as the one shown in Algortihm 2.2. The value of λ is chosen such that the computed update will result in a decrease of the objective function. If the initial λ is too low, it is increased.

Algorithm 2.2

1: Set $\lambda = 0$
2: Compute $\delta\mathbf{x}^{\star}$
3: Set $\lambda \geq 0$
4: **while** $J(\mathbf{x}^{\mathrm{op}} + \delta\mathbf{x}^{\star}) > J(\mathbf{x}^{\mathrm{op}})$ **do**
5: $\quad\lambda \leftarrow 10\lambda$
6: $\quad$Recompute $\delta\mathbf{x}^{\star}$
7: **end while**

2.8 State Estimation

A robotic system can be described by a set of states. These states often include position, attitude, and any other quantity that can help describe the motion of the body. Two methods of state estimation are considered herein, namely the Kalman filter and batch estimation.

2.8.1 Extended Kalman Filtering

The Kalman filter and its nonlinear variant, the extended Kalman filter (EKF), are two of the most common state estimation algorithms used today. Only the EKF is presented here.

Consider nonlinear process and measurement models given by

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{w}), \tag{2.13}$$

$$\mathbf{y}_k = \mathbf{g}_k(\mathbf{x}_k, \mathbf{v}_k), \tag{2.14}$$

where $\mathbf{x} \in \mathbb{R}^{n_x}$ is the state, $\mathbf{u} \in \mathbb{R}^{n_u}$ is the interoceptive measurement, $\mathbf{w} \in \mathbb{R}^{n_u}$ is the interoceptive measurement, or process, noise, $\mathbf{y}_k \in \mathbb{R}^{n_y}$ is the exteroceptive measurement at time t_k and $\mathbf{v}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k)$ is the exteroceptive measurement noise. The process noise is white and band-limited, and, when discretized, is normally distributed with zero mean and covariance $\mathbf{Q}_k$.

To implement an EKF, the process and measurement models must be linearized about an operating point. A first-order Taylor series expansion of (2.13) yields

$$\delta\dot{\mathbf{x}} = \mathbf{A}\delta\mathbf{x} + \mathbf{L}\delta\mathbf{w},$$

where

$$\mathbf{A} = \left.\frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{w})}{\partial \mathbf{x}}\right|_{\bar{\mathbf{x}}, \mathbf{u}, \bar{\mathbf{w}}},$$

$$\mathbf{L} = \left.\frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{w})}{\partial \mathbf{w}}\right|_{\bar{\mathbf{x}}, \mathbf{u}, \bar{\mathbf{w}}},$$

are the process model Jacobians evaluated at the nominal solution. Similarly, a first-order Taylor series expansion of (2.14) yields

$$\delta\mathbf{y}_k = \mathbf{H}_k\delta\mathbf{x}_k + \mathbf{L}_k\delta\mathbf{v}_k,$$

where

$$\mathbf{H}_k = \left.\frac{\partial \mathbf{g}_k(\mathbf{x}_k, \mathbf{v}_k)}{\partial \mathbf{x}_k}\right|_{\bar{\mathbf{x}}_k, \bar{\mathbf{v}}_k},$$

$$\mathbf{M}_k = \left.\frac{\partial \mathbf{g}_k(\mathbf{x}_k, \mathbf{v}_k)}{\partial \mathbf{v}_k}\right|_{\bar{\mathbf{x}}_k, \bar{\mathbf{v}}_k},$$

are the measurement model Jacobians evaluated at the nominal solution. In an EKF, the nominal noise values are assumed to be $\bar{\mathbf{w}} = \mathbf{0}$ and $\bar{\mathbf{v}}_k = \mathbf{0}$, and the nominal value of the state is the best estimate provided by the filter.

As interoceptive measurements are available, the state estimate is predicted by integrating

(2.13),

$$\check{\mathbf{x}}_k = \hat{\mathbf{x}}_{k-1} + \int_{t_{k-1}}^{t_k} \mathbf{f}(\hat{\mathbf{x}}, \mathbf{u}, \mathbf{0}) \mathrm{d}t,$$

where $(\hat{\cdot})$ is used to denote the corrected state and $(\check{\cdot})$ is used to denote the predicted state. The covariance is predicted using

$$\check{\mathbf{P}}_k = \mathbf{A}_{k-1}\mathbf{P}_{k-1}\mathbf{A}_{k-1}^\mathsf{T} + \mathbf{L}_{k-1}\mathbf{Q}_{k-1}\mathbf{L}_{k-1}^\mathsf{T}.$$

When exteroceptive measurements are available, the state estimate is corrected. The correction equations are

$$\begin{aligned}
\mathbf{K}_k &= \check{\mathbf{P}}_k\mathbf{H}_k^\mathsf{T}(\mathbf{H}_k\check{\mathbf{P}}_k\mathbf{H}_k^\mathsf{T} + \mathbf{M}_k\check{\mathbf{R}}_k\mathbf{M}_k^\mathsf{T})^{-1}, \\
\hat{\mathbf{x}}_k &= \check{\mathbf{x}}_k + \mathbf{K}_k(\mathbf{y}_k - \mathbf{g}(\check{\mathbf{x}}_k, \mathbf{0})), \\
\mathbf{P}_k &= (\mathbf{1} - \mathbf{K}_k\mathbf{H}_k)\check{\mathbf{P}}_k(\mathbf{1} - \mathbf{K}_k\mathbf{H}_k)^\mathsf{T} + \mathbf{K}_k\mathbf{M}_k\mathbf{R}_k\mathbf{M}_k^\mathsf{T}\mathbf{K}_k^\mathsf{T},
\end{aligned}$$

where $\mathbf{K}_k$ is the Kalman gain.

2.8.2 Batch Estimation [3, pp.127-143]

Batch estimation is useful in situations when the estimation algorithm does not need to run in real time. Once again, only the nonlinear variant of batch estimation is presented. The state at each time step composes the trajectory

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_n \end{bmatrix},$$

where $t \in [t_0, t_n]$. The discrete-time kinematics are given by

$$\mathbf{x}_k = \mathbf{f}_{k-1}\left(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1}\right),$$

where $\mathbf{u}_{k-1}$ are the interoceptive measurements, or inputs, and $\mathbf{w}_{k-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{k-1})$ is zero-mean white noise. The exteroceptive measurements are modelled as

$$\mathbf{y}_k = \mathbf{g}_k(\mathbf{x}_k) + \mathbf{v}_k$$

where $\mathbf{v}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k)$. A batch maximum *a posteriori* method is be used to solve the batch estimation problem. The input errors are

$$\mathbf{e}_{u,0}(\mathbf{x}) = \check{\mathbf{x}}_0 - \mathbf{x}_0.$$
$$\mathbf{e}_{u,k}(\mathbf{x}) = \mathbf{f}_{k-1}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{0}) - \mathbf{x}_k, \quad k = 1, \ldots, n,$$

where $\check{\mathbf{x}}_0$ is the initial state estimate, $\check{\mathbf{x}}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{P}_0)$. The errors in the exteroceptive measurements are

$$\mathbf{e}_{y,k}(\mathbf{x}) = \mathbf{y}_k - \mathbf{g}_k(\mathbf{x}_k), \quad k = 1, \ldots, n.$$

The objective function to minimize is

$$J(\mathbf{x}) = \frac{1}{2}\mathbf{e}_{u,0}(\mathbf{x})^\mathsf{T}\mathbf{W}_{u,0}^{-1}\mathbf{e}_{u,0}(\mathbf{x}) + \frac{1}{2}\sum_{k=1}^{n}\mathbf{e}_{u,k}(\mathbf{x})^\mathsf{T}\mathbf{W}_{u,k}^{-1}\mathbf{e}_{u,k}(\mathbf{x}) + \frac{1}{2}\sum_{k=1}^{n}\mathbf{e}_{y,k}(\mathbf{x})^\mathsf{T}\mathbf{W}_{y,k}^{-1}\mathbf{e}_{y,k}(\mathbf{x}),$$

where $\mathbf{W}_{u,0}$, $\mathbf{W}_{u,k}$, and $\mathbf{W}_{y,k}$ are weighting matrices related to the error distribution. The error is deemed Gaussian, as it is assumed that it arises only due to the presence of Gaussian noise in the measurements. Further defining

$$\mathbf{e}(\mathbf{x}) = \left[\begin{array}{c} \mathbf{e}_{u,0}(\mathbf{x}) \\ \vdots \\ \mathbf{e}_{u,n}(\mathbf{x}) \\ \hline \mathbf{e}_{y,1}(\mathbf{x}) \\ \vdots \\ \mathbf{e}_{y,n}(\mathbf{x}) \end{array}\right],$$

and $\mathbf{W}_u = \operatorname{diag}(\mathbf{W}_{u,0}, \mathbf{W}_{u,1}, \ldots, \mathbf{W}_{u,n})$, $\mathbf{W}_y = \operatorname{diag}(\mathbf{W}_{y,1}, \ldots, \mathbf{W}_{y,n})$ and $\mathbf{W} = \operatorname{diag}(\mathbf{W}_u, \mathbf{W}_y)$, the objective function is rewritten as

$$J(\mathbf{x}) = \frac{1}{2}\mathbf{e}(\mathbf{x})^\mathsf{T}\mathbf{W}^{-1}\mathbf{e}(\mathbf{x}).$$

To minimize this objective function, linearize the errors about an operating point $\mathbf{x}^{\text{op}}$. The linearized prior error has the form

$$\mathbf{e}_{u,0}(\mathbf{x}) = \mathbf{e}_{u,0}(\mathbf{x}^{\text{op}}) - \mathbf{F}_0^2\delta\boldsymbol{\epsilon}_0,$$

where $\delta\boldsymbol{\epsilon}_k$ is the error between the operating and truth trajectory. The linearized input error has the form

$$\mathbf{e}_{u,k}(\mathbf{x}) = \mathbf{e}_{u,k}(\mathbf{x}^{\text{op}}) + \mathbf{F}_k^1 \delta\boldsymbol{\epsilon}_{k-1} - \mathbf{F}_k^2 \delta\boldsymbol{\epsilon}_k.$$

The linearized measurement error has the form

$$\mathbf{e}_{y,k}(\mathbf{x}) = \mathbf{e}_{y,k}(\mathbf{x}^{\text{op}}) - \mathbf{H}_k \delta\boldsymbol{\epsilon}_k.$$

By stacking the estimation errors

$$\delta\mathbf{x} = \begin{bmatrix} \delta\boldsymbol{\epsilon}_0 \\ \vdots \\ \delta\boldsymbol{\epsilon}_n \end{bmatrix},$$

the linearized system can then be written

$$\mathbf{e}(\mathbf{x}) = \mathbf{e}(\mathbf{x}^{\text{op}}) - \boldsymbol{\Gamma}\delta\mathbf{x},$$

where

$$\boldsymbol{\Gamma} = \begin{bmatrix} \mathbf{A}^{-1} \\ \mathbf{H} \end{bmatrix},$$

where

$$\mathbf{A}^{-1} = \begin{bmatrix} \mathbf{F}_0^2 & & & \\ \mathbf{F}_1^1 & \mathbf{F}_1^2 & & \\ & \ddots & \ddots & \\ & & \mathbf{F}_{n-1}^1 & \mathbf{F}_{n-1}^2 \end{bmatrix},$$

and

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_1 & & \\ & \ddots & \\ & & \mathbf{H}_n \end{bmatrix}.$$

The linearized objective function is

$$J(\mathbf{x} + \mathbf{x}^{\text{op}}) = (\mathbf{e}(\mathbf{x}^{\text{op}}) + \boldsymbol{\Gamma}\delta\mathbf{x})^{\mathsf{T}} \mathbf{W}^{-1} (\mathbf{e}(\mathbf{x}^{\text{op}}) + \boldsymbol{\Gamma}\delta\mathbf{x}) \tag{2.15}$$

Minimizing (2.15) with respect to $\delta\mathbf{x}$ yields the Gauss-Newton update,

$$\left(\boldsymbol{\Gamma}^{\mathsf{T}}\mathbf{W}^{-1}\boldsymbol{\Gamma}\right)\delta\mathbf{x} = \boldsymbol{\Gamma}^{\mathsf{T}}\mathbf{W}^{-1}\mathbf{e}(\mathbf{x}^{\text{op}}),$$

which can be iteratively solved for the minimizing solution $\delta\mathbf{x}^{\star}$.

2.9 Summary

The tools presented in this chapter are used throughout this thesis. The invariant filtering theory applies directly to problems defined on matrix Lie groups. Matrix lie group theory, along with knowledge of geometry and kinematics, are used to build models. The optimization techniques detailed here are extensively used in solving SLAM problems and are essential to understanding the material presented in Chapter **??**. Lastly, this thesis builds on several well established state estimation techniques.

Chapter 3

Closing Remarks and Future Work

3.1 Conclusions

In this thesis, an in-depth analysis of state estimation in an invariant framework is presented. Through rigorous testing, the advantages and limitations of these different techniques are determined. Furthermore, an extension of invariant filtering theory to the problem of a batch solution to the SLAM problem is presented.

The IEKF is superior to the traditional MEKF in certain situations. It is better suited to problems where the state can be defined on matrix Lie groups, which is the case for many robotics problems. Throughout the simulations presented herein, the performance of the IEKF is on average better than that of the MEKF. However, only particular sample problems are used to illustrate this. It would therefore be irresponsible to state that the IEKF would always perform better than the MEKF. However, certain clear conclusions can be drawn.

First, state-independent Jacobians, such as those obtained in an IEKF, are advantageous in cases where the best estimate of the state is far from the true value. In most situations, this is seen when the initialization is poor. The IEKF's better performance is therefore mostly attributed to better performance in the transient period before the filter reaches steady state. Thus, the IEKF should be the state estimator of choice in applications where the initial state is unknown, and no other initialization scheme is available.

Second, leveraging the invariant framework in batch estimation only has limited advantages. In standard batch estimation, the Jacobians may initially be inaccurate if they depend on the state. However, as the solution converges, the Jacobians will be closer to the true Jacobians, as the error in the state estimate decreases. At this stage, there is minimal difference between a state-independent Jacobian and a Jacobian computed using an accurate state estimate.

3.2 Future Work

In Chapter **??**, the IEKF is compared to the MEKF. The MEKF was used a baseline as it is commonly used. However, comparing the IEKF to an iterative version of the MEKF may yield different results. The iterative MEKF improves upon the MEKF by recomputing the Jacobians at each time step until convergence. Furthermore, an iterative version of the IEKF could also be developed. This iterative IEKF would only be useful in scenarios where the process model is not group affine, or the measurement model is not invariant, leading to state dependent Jacobians like the MEKF.

Another avenue to explore would involve using a realistic sensor model in invariant batch SLAM. A future study using a stereo camera model or LIDAR model would also allow the invariant batch SLAM algorithms to be tested on experimental data.

Lastly, a study analyzing the consistency of the IEKF versus other other filtering techniques should be conducted. The IEKF should theoretically be more consistent, as its more accurate Jacobians mean the covariance better captures the underlying distribution. In a similar vein, the impact of unknown disturbances should be studied. The IEKF may be better suited to handle these, once again, due to theoretically exact Jacobians.

Bibliography

[1] A. Barrau and S. Bonnabel, "The Invariant Extended Kalman Filter as a Stable Observer", *IEEE Transactions on Automatic Control*, vol. 62, no. 4, pp. 1797–1812, 2017.

[2] A. Barrau and S. Bonnabel, "Invariant Kalman Filtering", *Annu. Rev. Control Robot. Auton. Syst*, vol. 1, no. 521, pp. 1–5, 2018.

[3] T. D. Barfoot, *State Estimation for Robotics*. Cambridge University Press, 2017.

[4] S. Särkkä, *Bayesian Filtering and Smoothing*. Cambridge University Press, 2013.

[5] G. Sibley, "A Sliding Window Filter for SLAM", University of Southern California, Tech. Rep., 2006, pp. 1–17.

[6] E. Eade, *Lie Groups for Computer Vision*, 2013.

[7] B. C. Hall, *Lie Groups, Lie Algebras, and Representations*, 2nd ed. Springer, 2014.

[8] T. D. Barfoot and P. T. Furgale, "Associating Uncertainty with Three-Dimensional Poses for Use in Estimation Problems", *IEEE Transactions on Robotics*, vol. 30, no. 3, pp. 679–693, 2014.

[9] A. Barrau, "Non-linear State Error Based Extended Kalman Filters with Applications to Navigation", PhD thesis, Mines Parsitech, 2015.

[10] A. Barrau and S. Bonnabel, "Linear Observed Systems on Groups", *Systems and Control Letters*, vol. 129, pp. 36–42, 2019.

[11] P. Hughes, *Spacecraft Attitude Dynamics*, ser. Dover Books on Aeronautical Engineering. Dover Publications, 2012.

[12] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. Springer, 2006.

[13] K. Levenberg, "A Method for the Solution of Certain Non-linear Problems in Least Squares", *Quarterly of Applied Mathematics*, vol. 2, no. 2, pp. 164–168, 1944.

[14] D. W. Marquardt, "A Method for the Solution of Certain Non-linear Problems in Least Squares", *Journal of the Society for Industrial and Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963.

[15] S. Roweis, *Levenberg-Marquardt Optimization*, `https://cs.nyu.edu/~roweis/notes/lm.pdf`, 1996.

In the introduction chapter Chapter 1, we referenced the appendices in Appendix A and in Appendix A.1. Then we have Chapter A.

Appendices

Appendix 1

A.1 Section 1

A.2 Section 2

Appendix 2

Appendix 2

B.1 Section 1

B.2 Section 2

Chapter 1

Hii