Data-Driven Parameter Learning without Groundtruth for Improving Robotic State Estimation

by

Jeremy N. Wong

A thesis submitted in conformity with the requirements
for the degree of Master of Applied Science
Graduate Department of University of Toronto Institute for Aerospace Studies
University of Toronto

Abstract

Data-Driven Parameter Learning without Groundtruth for Improving Robotic State Estimation

Jeremy N. Wong
Master of Applied Science
Graduate Department of University of Toronto Institute for Aerospace Studies
University of Toronto
2020

Probabilistic state estimation for robotics is a field that has matured a lot in recent years. Yet the performance of modern estimators is still heavily dependent on robot model parameters that can be difficult to determine from first principles. While it may be sufficient to hand tune these parameters, this is often time consuming or simply impossible due to the sheer number of unknown parameters. In this thesis, we investigate methods for learning parameters based on data, to come up with the parameter values most suitable for the particular robot and sensor. We first develop a novel continuous-time motion prior trained with data to improve an existing continuous-time estimation framework. We then provide a detailed investigation into parameter learning within a Gaussian variational inference setting using Expectation Maximization. We validate our work on various trajectory estimation problems using a 36 km long vehicle dataset collected as part of this work.

Acknowledgements

It is only through the efforts and support of various groups of people that made the completion of this thesis possible. I am grateful for my family, my parents and older sister for their support from the very beginning. I would also like to thank the members of the Dynamic Systems Lab and the Autonomous Space Robotics Lab for creating a friendly atmosphere and always being open to discussions, whether related to research or not. Thank you David for working with me on various research endeavours and providing a source of guidance and knowledge. Finally, I would like to thank my supervisors Professor Angela Schoellig and Professor Tim Barfoot for their guidance, expertise, passion and commitment in helping me grow as researcher.

Contents

List of Tables

List of Figures

Notation

$\underrightarrow{\boldsymbol{\mathcal{F}}}_a$	A reference frame for a three-dimensional coordinate system
$SE(3)$	The special Euclidean group, a matrix Lie group used to represent poses
$\mathfrak{se}(3)$	The Lie algebra associated with a member of $SE(3)$
$\mathbf{T}_{a,b}$	A matrix in $SE(3)$ that transforms vectors from frame $\underrightarrow{\boldsymbol{\mathcal{F}}}_b$ to frame $\underrightarrow{\boldsymbol{\mathcal{F}}}_a$
$\mathbf{T}_k$	Short form for $\mathbf{T}_{k,i}$, a matrix in $SE(3)$ that transforms vectors from $\underrightarrow{\boldsymbol{\mathcal{F}}}_i$, the inertial frame, to $\underrightarrow{\boldsymbol{\mathcal{F}}}_k$, the frame representing the pose at time t_k
$\exp(\cdot^\wedge)$	A Lie algebra operator mapping from $\mathfrak{se}(3)$ to $SE(3)$
$\ln(\cdot)^\vee$	A Lie algebra operator mapping from $SE(3)$ to $\mathfrak{se}(3)$
$(\cdot)^\curlywedge$	An operator associated with the adjoint of an element from the Lie algebra for poses
$\boldsymbol{\mathcal{J}}(\cdot)$	The left Jacobian of $SE(3)$
$\boldsymbol{\mathcal{J}}_{k+1,k}$	Short form for $\boldsymbol{\mathcal{J}}(\ln(\mathbf{T}_{k+1,k})^\vee)$
$p(\mathbf{a})$	The probability density of $\mathbf{a}$.
$p(\mathbf{a} \mid \mathbf{b})$	The probability density of $\mathbf{a}$ given $\mathbf{b}$.
$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$	A normal distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$.
$\mathcal{GP}(\boldsymbol{\mu}, \boldsymbol{\mathcal{K}}(t, t'))$	A Gaussian process with mean function $\boldsymbol{\mu}$ and covariance function $\boldsymbol{\mathcal{K}}(t, t')$
$\mathbf{x}_k$	The value of a quantity $\mathbf{x}$ at timestep k.
$\mathbf{x}_{k,k+1}$	The set of values of $\mathbf{x}$ at timesteps k and $k+1$, or $\{\mathbf{x}_k, \mathbf{x}_{k+1}\}$
$\mathbf{1}$	The identity matrix.
$\mathbf{0}$	The zero matrix.

Chapter 1

Introduction

1.1 Background & Motivation

In mobile robotics, it is crucial for a robot to maintain an estimate of its state as it moves throughout and interacts with its environment. The field of state estimation is concerned with tackling this problem and is a key ingredient in enabling any autonomous navigation. For instance, a self-driving car without information about its location in a map or relative to features such as lane markings, would render attempts at achieving any level of autonomy futile. A key area of research that greatly advanced the field was probabilistic robotics. Critical to probabilistic robotics is the idea of incorporating uncertainty into estimation and control, enabling robustness in many real-world situations [49].

However, in many cases, there are still parameters or hyperparameters that must be tuned as these can greatly affect the performance of state estimation algorithms. For some problems, it may be sufficient to hand tune parameters but in other cases, it is simply impossible due to the number of parameters involved or the time that would be required to do so. Adhering to the paradigm of a probabilistic framework, it is essential that model parameters are selected in such a way to ensure that not only is the mean of our models realistic but that our uncertainty quantification is realistic as well.

Borrowing from the field of machine learning, there are learning methods that are based on finding parameters that maximize the log likelihood of the data. Much of this

research has focused on data expressed in a vectorspace, or in $\mathbb{R}^n$. With robots moving and rotating through three dimensional space, our data is no longer within this set, instead lying on a Lie group, in particular the special Euclidean group $SE(3)$.

In this thesis, we look to methods for training parameters based on data to come up with the best parameter values suited for the particular robot. Our methods continue to function with typical robot states, including poses that are in $SE(3)$. In the first part of this thesis, we focus on using a data-driven approach to improve a continuous-time estimation framework that has been widely used in the past [8, 4, 3, 48]. This is achieved by introducing a novel continuous-time motion prior, whose parameters we then learn from data. We show that using our novel motion prior allows for greater flexibility to learn a good prior from data and thus is able to outperform previously used continuous-time motion priors in several experimental state estimation problems. However, our method of parameter learning requires (noisy) observations of the full state. This is often an unrealistic assumption and so in the second part, we explore parameter learning within a novel state estimation framework, Exactly Sparse Gaussian Variational Inference (ESGVI) [6] and show how to achieve a robust extension (outlier rejection).

1.2 Related Work

1.2.1 Continuous-Time Estimation

It is common for state estimation in robotics to be carried out in discrete time, which is an approximation of the robot's continuous-time trajectory. This approximation is sufficient in many cases, but there are situations when it is inadequate. Examples include using continuous scanning-while-moving sensors (e.g., rolling-shutter camera or scanning laser rangefinder), or high-rate asynchronous sensors. In both these cases, the naive discrete approach of including a state at every measurement time can be expensive. Many people have attempted to address this problem by using ad-hoc assumptions about the smoothness of the trajectory in order to use a smaller number of discrete states and infer motion in between. One such example is using cubic splines as in [11] and [60].

However, the specific interpolation scheme chosen encodes certain assumptions about the robot motion that may not be accurate. By explicitly formulating the problem in a continuous-time framework, the need to make these often arbitrary smoothness assumptions is eliminated. Smoothness can be built into the estimation exactly using a continuous-time motion prior, thus gaining the ability to incorporate measurements at any time along the trajectory without introducing additional states at those measurement times. Furthermore, continuous-time techniques benefit from the advantage that posterior estimates can be queried at any time along the trajectory, not just at measurement times.

Tong et al. [51, 50] showed that batch continuous-time estimation can be carried out by representing the trajectory as a Gaussian process (GP). Barfoot et al. [8] extended the GP approach to STEAM, using a class of linear time-invariant (LTI) stochastic differential equations (SDEs). This resulted in an inverse kernel matrix that is exactly sparse, making the solution more computationally efficient. Anderson and Barfoot [3] extended this work in $SE(3)$ to enable continuous-time estimation of bodies undergoing translations and rotations in three-dimensional space using a white-noise-on-acceleration (WNOA) prior. Tang et al. [48] further extended this by using a white-noise-on jerk (WNOJ) prior in $SE(3)$. STEAM has been used in applications such as motion planning [38], crop monitoring [16], and visual teach and repeat [53].

Both WNOA and WNOJ models are commonly used in target tracking [34, 24, 36, 26]. These models incorporate assumptions about the motion of the target that may not be realistic but are attractive due to their simplicity. When white-noise models are insufficient, Markov process models are used where the control input is modelled as a Markov process instead of a white-noise process. One example is the Singer acceleration model [45].

Another way to view modelling is through the use of latent-force models, which combine mechanistic and data-driven approaches [2]. In latent-force models, typically an overly simplistic mechanistic model of the system is used but augmented with latent forces represented as a GP. The idea is that when training latent-force models through data, the GP can model interactions not captured by the mechanistic model. Thus the

GP must be rich enough to fully learn these interactions. In the case of the WNOA and WNOJ prior, the GP used is simply a white-noise process, which struggles with representational power. Thus tuning the hyperparameters of WNOA and WNOJ models alone do not give the model enough flexibility to learn a good representation through data.

Hartikainen and Särkkä [20] show how Gaussian process regression models can be restated as linear-Gaussian state space models. In particular, they show that the Matérn family of covariance functions can be exactly reformulated as a SDE with white noise. Furthermore, these Gaussian process latent force models can be reformulated as a single linear SDE driven by white noise [21], which has the form of priors in which we are interested.

1.2.2 Parameter Learning

In the domain of parameter learning, the most common approach is to find parameters that maximize the likelihood of the data. One way to do this is to directly maximize the likelihood function with respect to the parameters [56, 29, 30]. This can be a difficult problem to solve, particularly when the model depends on missing or unobserved variables. In this case, an indirect approach can be taken by introducing a latent state to the problem, which can be estimated alongside of the parameters. This is known as Expectation Maximization (EM), an iterative algorithm that alternates between optimizing for a distribution over the latent state and the parameters.

Past work has shown how to estimate all the parameters of a linear dynamical system using EM, with Kalman smoothing in the E-step to update states and calculating analytic equations for parameter updates in the M-step [44]. There have also been methods that attempt parameter learning for nonlinear systems with EM. Ghahramani and Roweis [19] learn a full nonlinear model using Gaussian Radial Basis Functions (RBFs) to approximate the nonlinear expectations that would otherwise be intractable to compute. This method was applied to learn a simple sigmoid nonlinearity. Other methods approximate the required expectation using particle smoothing [43] or sigmapoint smoothing [29, 30, 17]. These methods, however, do not learn a full nonlinear model, but only learned parameters of a mostly predefined model (e.g., calibration parameters), and were

tested only in simulation.

Unlike all these other methods, we use ESGVI within the EM parameter learning framework, which is a more general method not limited to problems with a specific factorization of the joint likelihood between the data and the latent state (e.g., smoothing problems with a block-tridiagonal inverse covariance). We also demonstrate a practical application of parameter learning by estimating the parameters of our motion prior and measurement noise models in a batch estimation framework.

While we are interested in batch estimation, previous work has investigated learning the noise model parameters of filters. Abbeel et al. [1] learn the noise model parameters of the Kalman Filter offline. However, these parameters are assumed to be static and do not vary with time. One popular area of study that handles changing covariances is Adaptive Kalman Filtering, where the measurement covariance is updated in an online fashion based on the statistics of the measurement innovation [37, 23, 59]. The measurement covariance in these cases is updated based solely on the data seen during inference, whereas our method will incorporate a prior.

Ko and Fox [27] use Gaussian processes to represent measurement models (mapping from state to observation) and motion models (mapping from state and control input to change in state) and show how to incorporate them in a filter estimation framework. Their method focuses on using GPs (a non-parametric model) whereas the method we present in Chapter 3 allows for learning the parameters of arbitrary models. In this thesis, our use of GPs is different in that we use them to represent continuous-time trajectories (mapping from time to state). We then learn the hyperparameters of our GP motion prior. While their initial method required groundtruth of the states, Ko and Fox [28] extended their work by jointly estimating the latent state and GP hyperparameters using conjugate gradient descent and provide analytic gradients in terms of the kernel matrix. They then use the estimated latent state as training data for their GP models as in [27]. Our use of Expectation Maximization instead of a direct log likelihood optimization approach allows us to learn parameters of arbitrary models since direct optimization may not always be possible.

Recent methods take advantage of deep neural networks (DNNs) to learn the robot

noise models [12, 35, 41] but in many cases require groundtruth to train the DNN. We bypass this requirement by simultaneously estimating a distribution over the latent state.

Barfoot et al. [6] show how to learn a constant covariance using ESGVI through EM but do not demonstrate it in practice. Our main contributions compared to [6] is demonstrating parameter learning for a specific application and learning time-varying covariances by introducing an Inverse-Wishart (IW) prior over our covariances, which enables outlier rejection. As an alternate method for outlier rejection, Chebrolu et al. [15] use EM to learn a tuning parameter for M-estimation but treat their latent variables as point estimates. The IW distribution has been used as a prior over covariances before, but the parameters were assumed to be known [55]. We seek to learn at least some of the parameters of the prior.

1.3 Contributions

The main contributions of this thesis are:

- The derivation of a richer continuous-time motion prior on $SE(3)$ and a principled method for training its hyperparameters with (noisy) observations of the full state. The work on this topic resulted in the following publication:

  Jeremy N Wong, David J Yoon, Angela P Schoellig, and Timothy D Barfoot. A Data-Driven Motion Prior for Continuous-Time Trajectory Estimation on SE (3). *IEEE Robotics and Automation Letters*, 5(2):1429–1436, 2020.

- A detailed investigation of parameter learning in the absence of groundtruth as part of ESGVI, in addition to a robust extension of ESGVI for outlier rejection. The work on this topic resulted in the following publication:

  Jeremy N Wong, David J Yoon, Angela P Schoellig, and Timothy D Barfoot. Variational Inference with Parameter Learning Applied to Vehicle Trajectory Estimation. *IEEE Robotics and Automation Letters*, 5(4):5291–5298, 2020.

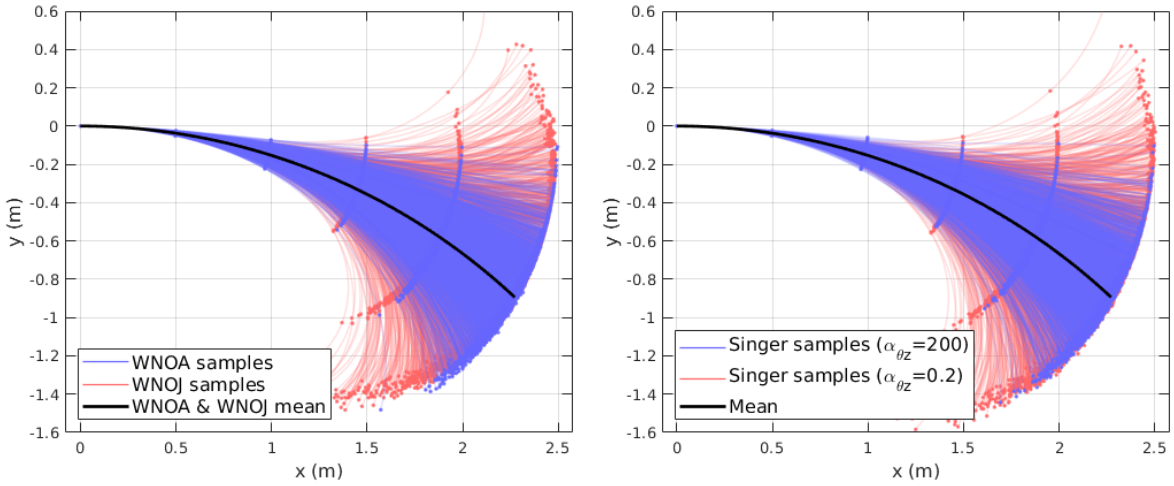
Chapter 2

A Data-Driven Prior on *SE(3)*

2.1 Introduction

Current formulations of continuous-time estimation employ either a white-noise-on-acceleration (WNOA) [3, 38, 58] or white-noise-on-jerk (WNOJ) [48, 39] motion prior, which assume the prior mean is constant velocity or constant acceleration, respectively. Tang et al. [48] showed that the WNOJ prior can be formulated so that its mean matches the mean of any WNOA prior. However, the two motion priors will always have different covariances as seen in Figure 2.1(a) (other than the trivial case when both have a covariance of $\mathbf{0}$). Integral to any probabilistic estimation framework is the notion of uncertainty, captured through covariances. As such, in keeping with the idea that models consist of both a mean and a measure of uncertainty, the WNOA and WNOJ prior are two non-overlapping sets of models. Thus the explicit choice of either one of these priors is limiting and we argue that it is much more principled to learn the parameters of a richer motion prior based on data.

With this in mind, we derive a motion prior that represents latent accelerations as a Gaussian Process (GP) with a Matérn covariance function and learn the hyperparameters of this GP from noisy groundtruth data. Our derivation starts by transforming the GP into a stochastic differential equation (SDE), which we show to be equivalent to the Singer acceleration model [45]. This motion prior can represent but is not limited to both WNOA and WNOJ trajectories as seen in Figure 2.1(b).



(a) WNOA & WNOJ samples **(b)** Singer samples

Figure 2.1: **(a)** While the white-noise-on-jerk prior can be formulated such that its mean matches that of any white-noise-on-acceleration prior, their covariances can never be the same. **(b)** The Singer prior is a richer prior that can represent but is not limited to both types of trajectories.

We then show that on a real world lidar test dataset with over 20 km of driving, our motion prior outperforms the WNOA and WNOJ priors, which have also been properly trained using groundtruth data. The contribution of this chapter is two-fold. The first is a principled method for hyperparameter training of continuous-time motion priors in $SE(3)$. This opens up the possibility of using far richer motion priors with more parameters. This leads to the second part of the chapter, the derivation of a new richer data-driven motion prior.

Using latent-force models in state estimation has been done before but in a discrete-time estimation framework and where the states were limited to be in $\mathbb{R}^n$ [22, 42, 52]. Regarding hyperparameter training for Gaussian process models, it is usually carried out by minimizing the negative log likelihood of data given some parameters, but this approach has only been carried out in $\mathbb{R}^n$ [54, 4].

To the best of our knowledge, the derivation we present in this chapter is the first attempt at modelling the trajectory using a latent-force model with a Matérn covariance in the context of continuous-time trajectory estimation on $SE(3)$ and using a data-driven approach to estimate its parameters.

2.2 Overview of STEAM

In this section, we first provide the problem setup and overview of the Simultaneous Trajectory Estimation and Maping (STEAM) framework. STEAM is an estimation framework that given a collection of factors involving an unknown set of discrete states, solves for the continuous posterior state estimate. In the original STEAM formulation using a WNOA prior, the state trajectory we wish to estimate is

$$\mathbf{x}(t) = \{\mathbf{T}(t), \boldsymbol{\varpi}(t)\}, \tag{2.1}$$

where $\mathbf{T}(t)$ is a pose expressed as a transformation matrix and $\boldsymbol{\varpi}(t)$ is the body-centric velocity. Thus the set of discrete states involved in our optimization problem is

$$\mathbf{x} = \{\mathbf{T}_1, \boldsymbol{\varpi}_1, \ldots, \mathbf{T}_K, \boldsymbol{\varpi}_K\}, \tag{2.2}$$

where we have K knots in our continuous-time trajectory with a single knot being defined as the state $\mathbf{x}_k$ at time t_k, or $\mathbf{x}(t_k)$.

Factors usually come from either a set of measurements or from a continuous-time motion prior, the key component that allows us to generate a continuous-time posterior trajectory. Each of the factors make up a term in the negative log likelihood cost function that is minimized by STEAM with respect to the states. We are taking the Maximum A Posterior estimation approach where we find the single best estimate of the state given the measurements and the motion prior. The general form of the cost function with the prior and measurement cost terms is as follows:

$$J(\mathbf{x}) = \underbrace{\sum_{k} J_k}_{\text{prior}} + \underbrace{\sum_{j} J_j}_{\text{measurement}}. \tag{2.3}$$

The optimal state estimate is then

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} J(\mathbf{x}). \tag{2.4}$$

The optimization is solved using Gauss-Newton with an $SE(3)$ perturbation scheme [5, 7] to update the states:

$$\begin{aligned} \mathbf{T}_{\text{op},k} &\leftarrow \exp(\delta\boldsymbol{\xi}_k^{\wedge})\mathbf{T}_{\text{op},k}, \\ \boldsymbol{\varpi}_{\text{op},k} &\leftarrow \boldsymbol{\varpi}_{\text{op},k} + \delta\boldsymbol{\varpi}_k, \end{aligned} \tag{2.5}$$

where $(\cdot)_{\text{op}}$ is the operating point. If all we cared about was estimating the state at all our discrete knot times, this approach would offer no advantage over the usual discrete-time approach to batch state estimation. However, combining the discrete set of posterior states with an interpolation scheme that is specific to the particular motion prior used results in a continuous-time trajectory estimate.

The measurement terms are specific to the sensors on the robot. In this chapter, we will be focusing only on the motion prior side of the estimation problem. Each prior cost term has the form

$$J_k = \frac{1}{2}\mathbf{e}_k^T\mathbf{Q}_k^{-1}\mathbf{e}_k. \tag{2.6}$$

The formulation of the prior error terms, $\mathbf{e}_k$, and covariances, $\mathbf{Q}_k$, is dependent on the specific prior used. In the next section, we review the existing WNOA and WNOJ continuous-time motion priors before moving onto our new formulation.

2.3 Existing Continuous-Time Motion Priors

In this section, we first go over the details of the existing WNOA and WNOJ motion priors used in STEAM. In order to ensure that estimation can be done efficiently, we are interested in motion priors from a class of linear time-invariant (LTI) stochastic differential equations (SDEs) of the form

$$\begin{aligned} \dot{\boldsymbol{\gamma}}(t) &= \boldsymbol{A}\boldsymbol{\gamma}(t) + \boldsymbol{B}\mathbf{u}(t) + \boldsymbol{L}\mathbf{w}(t), \\ \mathbf{w}(t) &\sim \mathcal{GP}(\mathbf{0}, \mathbf{Q}_c\delta(t-t')), \end{aligned} \tag{2.7}$$

where $\boldsymbol{\gamma}(t)$ is the state at timestep t, $\mathbf{u}(t)$ is a known input, and $\mathbf{w}(t)$ is a zero-mean, white-noise GP with power spectral density matrix, $\mathbf{Q}_c$. If $\mathbf{u}(t) = \mathbf{0}$, the solution for the

mean function is

$$\check{\boldsymbol{\gamma}}(\tau) = \boldsymbol{\Phi}(\tau, t_k)\check{\boldsymbol{\gamma}}(t_k), \tag{2.8}$$

where $\check{\boldsymbol{\gamma}}$ is the prior mean, and $\boldsymbol{\Phi}(\tau, t_k)$ is the state transition function from timestep t_k to timestep τ. We use GP priors because of their rich mathematical connection to motion models and the propagation of uncertainty is well understood.

2.3.1 WNOA Prior for SE(3)

The WNOA prior originally used by STEAM is defined as follows:

$$\begin{aligned} \dot{\mathbf{T}}(t) &= \boldsymbol{\varpi}(t)^{\wedge}\mathbf{T}(t) \\ \dot{\boldsymbol{\varpi}} &= \mathbf{w}'(t), \quad \mathbf{w}'(t) \sim \mathcal{GP}(\mathbf{0}, \mathbf{Q}_c'\delta(t-t')), \end{aligned} \tag{2.9}$$

where $\mathbf{T}(t)$ is the pose expressed as a transformation matrix, $\boldsymbol{\varpi}(t)$ is the body-centric velocity and the operator, $\wedge$, transforms an element of $\mathbb{R}^6$ into a member of Lie algebra, $\mathfrak{se}(3)$. The state is

$$\mathbf{x}(t) = \{\mathbf{T}(t), \boldsymbol{\varpi}(t)\}. \tag{2.10}$$

The SDE in (2.9) is nonlinear and so cannot be cast into the form of (2.7), but [3] defines the local state variables

$$\boldsymbol{\xi}_k(t) = \ln(\mathbf{T}(t)\mathbf{T}_k^{-1})^{\vee}, \quad t_k \leq t \leq t_{k+1} \tag{2.11}$$

$$\dot{\boldsymbol{\xi}}_k(t) = \boldsymbol{\mathcal{J}}(\boldsymbol{\xi}_k(t))^{-1}\boldsymbol{\varpi}(t). \tag{2.12}$$

These local variables can be used to define a sequence of local priors where the prior at each knot is a LTI SDE in the form described in (2.7) with

$$\boldsymbol{\gamma}_k(t) := \begin{bmatrix} \boldsymbol{\xi}_k(t) \\ \dot{\boldsymbol{\xi}}_k(t) \end{bmatrix}, \quad \boldsymbol{A} = \begin{bmatrix} \mathbf{0} & \mathbf{1} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}, \quad \boldsymbol{B} = \mathbf{0}, \quad \boldsymbol{L} = \begin{bmatrix} \mathbf{0} \\ \mathbf{1} \end{bmatrix}. \tag{2.13}$$

This local prior is a good approximation of the global non-linear prior when $\boldsymbol{\xi}_k(t)$ is small or when velocity is near constant.

The error term from (2.6) for the WNOA prior is [3]

$$\mathbf{e}_k = \begin{bmatrix} \ln(\mathbf{T}_{k+1,k})^\vee - (t_{k+1} - t_k)\boldsymbol{\varpi}_k \\ \boldsymbol{\mathcal{J}}^{-1}_{k+1,k}\boldsymbol{\varpi}_{k+1} - \boldsymbol{\varpi}_k \end{bmatrix}, \tag{2.14}$$

while the covariance and inverse covariance is [8, 3]

$$\mathbf{Q}_k = \begin{bmatrix} \frac{1}{3}\Delta t_k^3 \mathbf{Q}_c & \frac{1}{2}\Delta t_k^2 \mathbf{Q}_c \\ \frac{1}{2}\Delta t_k^2 \mathbf{Q}_c & \Delta t_k \mathbf{Q}_c \end{bmatrix} \quad \text{and} \quad \mathbf{Q}_k^{-1} = \begin{bmatrix} 12\Delta t_k^{-3}\mathbf{Q}_c^{-1} & -6\Delta t_k^{-2}\mathbf{Q}_c^{-1} \\ -6\Delta t_k^{-2}\mathbf{Q}_c^{-1} & 4\Delta t_k^{-1}\mathbf{Q}_c^{-1} \end{bmatrix}. \tag{2.15}$$

2.3.2 WNOJ Prior for SE(3)

The WNOJ prior from [48] estimates the global state

$$\mathbf{x}(t) = \{\mathbf{T}(t), \boldsymbol{\varpi}(t), \dot{\boldsymbol{\varpi}}(t)\}, \tag{2.16}$$

where $\dot{\boldsymbol{\varpi}}(t)$ is the body-centric acceleration. Using the idea of local pose variables, [48] defines a sequence of local priors as a LTI SDE in the form of (2.7):

$$\boldsymbol{\gamma}_k(t) := \begin{bmatrix} \boldsymbol{\xi}_k(t) \\ \dot{\boldsymbol{\xi}}_k(t) \\ \ddot{\boldsymbol{\xi}}_k(t) \end{bmatrix} \quad \boldsymbol{A} = \begin{bmatrix} \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}, \quad \boldsymbol{B} = \mathbf{0}, \quad \boldsymbol{L} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \end{bmatrix}. \tag{2.17}$$

The relationship between $\boldsymbol{\xi}_k(t)$ and $\dot{\boldsymbol{\xi}}_k(t)$ and global state variables are shown in Equations (2.11) and (2.12). The relationship between $\ddot{\boldsymbol{\xi}}_k(t)$ and global state variables as shown in [48] is

$$\ddot{\boldsymbol{\xi}}_k(t) \approx -\frac{1}{2}(\boldsymbol{\mathcal{J}}(\boldsymbol{\xi}_k(t))^{-1}\boldsymbol{\varpi}(t))^\curlywedge \boldsymbol{\varpi}(t) + \boldsymbol{\mathcal{J}}(\boldsymbol{\xi}_k(t))^{-1}\dot{\boldsymbol{\varpi}}(t), \tag{2.18}$$

where the approximation holds as long as $\boldsymbol{\xi}_k(t)$ is small.

The error term from (2.6) for the WNOJ prior is [48]

$$\mathbf{e}_k = \begin{bmatrix} \ln(\mathbf{T}_{k+1,k})^\vee - (t_{k+1}-t_k)\boldsymbol{\varpi}_k - \frac{1}{2}(t_{k+1}-t_k)^2\dot{\boldsymbol{\varpi}}_k \\ \boldsymbol{\mathcal{J}}^{-1}_{k+1,k}\boldsymbol{\varpi}_{k+1} - \boldsymbol{\varpi}_k - (t_{k+1}-t_k)\dot{\boldsymbol{\varpi}}_k \\ -\frac{1}{2}(\boldsymbol{\mathcal{J}}^{-1}_{k+1,k}\boldsymbol{\varpi}_{k+1})^\curlywedge\boldsymbol{\varpi}_{k+1} + \boldsymbol{\mathcal{J}}^{-1}_{k+1,k}\dot{\boldsymbol{\varpi}}_{k+1} - \dot{\boldsymbol{\varpi}}_k \end{bmatrix}, \tag{2.19}$$

while the covariance and inverse covariance is

$$\mathbf{Q}_k = \begin{bmatrix} \frac{1}{20}\Delta t_k^5\mathbf{Q}_c & \frac{1}{8}\Delta t_k^4\mathbf{Q}_c & \frac{1}{6}\Delta t_k^3\mathbf{Q}_c \\ \frac{1}{8}\Delta t_k^4\mathbf{Q}_c & \frac{1}{3}\Delta t_k^3\mathbf{Q}_c & \frac{1}{2}\Delta t_k^2\mathbf{Q}_c \\ \frac{1}{6}\Delta t_k^3\mathbf{Q}_c & \frac{1}{2}\Delta t_k^2\mathbf{Q}_c & \Delta t_k\mathbf{Q}_c \end{bmatrix} \tag{2.20}$$

and

$$\mathbf{Q}_k^{-1} = \begin{bmatrix} 720\Delta t_k^{-5}\mathbf{Q}_c^{-1} & -360\Delta t_k^{-4}\mathbf{Q}_c^{-1} & 60\Delta t_k^{-3}\mathbf{Q}_c^{-1} \\ -360\Delta t_k^{-4}\mathbf{Q}_c^{-1} & 192\Delta t_k^{-3}\mathbf{Q}_c^{-1} & -36\Delta t_k^{-2}\mathbf{Q}_c^{-1} \\ 60\Delta t_k^{-3}\mathbf{Q}_c^{-1} & -36\Delta t_k^{-2}\mathbf{Q}_c^{-1} & 9\Delta t_k^{-1}\mathbf{Q}_c^{-1} \end{bmatrix}. \tag{2.21}$$

2.4 Latent-Force Model GP Prior

In the new motion prior that we derive, we represent the latent accelerations that the robot undergoes in each of its six degrees of freedom as a GP with a Matérn covariance function:

$$\ddot{\boldsymbol{\xi}}_k(t) \sim \mathcal{GP}(\mathbf{0}, \boldsymbol{\mathcal{K}}_v(t,t')). \tag{2.22}$$

For our prior, we choose $v = \frac{1}{2}$, which yields the exponential covariance function defined as

$$\boldsymbol{\mathcal{K}}_{1/2}(t,t') = \boldsymbol{\sigma}^2 \exp(-\boldsymbol{\ell}^{-1}|t-t'|), \tag{2.23}$$

where $\boldsymbol{\ell}$, the length-scale and $\boldsymbol{\sigma}^2$, the variance, are diagonal matrices in $\mathbb{R}^{6\times 6}$, with each diagonal term representing one of the six degrees of freedom.

Following the approach of [20], we can express the GP representing acceleration as

the SDE

$$\begin{aligned} \dddot{\boldsymbol{\xi}}_k(t) &= -\boldsymbol{\alpha}\ddot{\boldsymbol{\xi}}_k(t) + \mathbf{w}(t), \\ \mathbf{w}(t) &\sim \mathcal{GP}(\mathbf{0}, \mathbf{Q}_c\delta(t-t')), \end{aligned} \tag{2.24}$$

where $\boldsymbol{\alpha} = \boldsymbol{\ell}^{-1}$ and $\mathbf{Q}_c = 2\boldsymbol{\alpha}\boldsymbol{\sigma}^2$.

Now following the approach of [21], the model can be augmented to form a joint model in the form of (2.7) that includes the states $\boldsymbol{\xi}_k(t)$ and $\dot{\boldsymbol{\xi}}_k(t)$:

$$\boldsymbol{\gamma}_k(t) := \begin{bmatrix} \boldsymbol{\xi}_k(t) \\ \dot{\boldsymbol{\xi}}_k(t) \\ \ddot{\boldsymbol{\xi}}_k(t) \end{bmatrix} \quad \boldsymbol{A} = \begin{bmatrix} \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \\ \mathbf{0} & \mathbf{0} & -\boldsymbol{\alpha} \end{bmatrix}, \quad \boldsymbol{B} = \mathbf{0}, \quad \boldsymbol{L} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \end{bmatrix}. \tag{2.25}$$

The global state remains the same as in (2.16).

Our new motion prior now includes length-scale and variance as hyperparameters embedded in $\boldsymbol{\alpha}$ and $\mathbf{Q}_c$, respectively. This allows much greater flexibility in our motion prior compared to the previous WNOA and WNOJ formulations that did not have a tunable length-scale parameter. In fact, WNOA and WNOJ are special cases of this motion prior. WNOA is the case when length-scale approaches 0, meaning accelerations are uncorrelated. WNOJ is the case when length-scale approaches ∞, meaning that accelerations are correlated to accelerations at all other times.

This particular parameterization of the motion prior collapses to the exact form of the Singer acceleration model in [45]. As such, we will be referring to this prior as the Singer prior. In this work, we chose this particular Matérn covariance function but we could potentially use other covariance functions that have more representational power.

2.4.1 Prior Error Term

The state transition function is

$$\begin{aligned}\boldsymbol{\Phi}(t, t_k) &= \exp(\boldsymbol{A}\Delta t_k) \\ &= \begin{bmatrix} \mathbf{1} & \Delta t_k \mathbf{1} & (\boldsymbol{\alpha}\Delta t_k - \mathbf{1} + \exp(-\boldsymbol{\alpha}\Delta t_k))\boldsymbol{\alpha}^{-1} \\ \mathbf{0} & \mathbf{1} & (\mathbf{1} - \exp(-\boldsymbol{\alpha}\Delta t_k))\boldsymbol{\alpha}^{-1} \\ \mathbf{0} & \mathbf{0} & \exp(-\boldsymbol{\alpha}\Delta t_k) \end{bmatrix}.\end{aligned} \tag{2.26}$$

Using Equations (2.11), (2.12) and (2.18), the local state variables can be written in terms of global state variables as

$$\boldsymbol{\gamma}_k(t_k) = \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\varpi}_k \\ \dot{\boldsymbol{\varpi}}_k \end{bmatrix}, \qquad \boldsymbol{\gamma}_k(t_{k+1}) = \begin{bmatrix} \ln(\mathbf{T}_{k+1,k})^\vee \\ \boldsymbol{\mathcal{J}}^{-1}_{k+1,k}\boldsymbol{\varpi}_{k+1} \\ -\frac{1}{2}(\boldsymbol{\mathcal{J}}^{-1}_{k+1,k}\boldsymbol{\varpi}_{k+1})^\curlywedge \boldsymbol{\varpi}_{k+1} + \boldsymbol{\mathcal{J}}^{-1}_{k+1,i}\dot{\boldsymbol{\varpi}}_{k+1} \end{bmatrix}. \tag{2.27}$$

Now in terms of global state variables, the prior error term is

$$\mathbf{e}_k = \boldsymbol{\gamma}_k(t_{k+1}) - \boldsymbol{\Phi}(t_{k+1}, t_k)\boldsymbol{\gamma}_k(t_k) = \begin{bmatrix} \mathbf{e}_p \\ \mathbf{e}_v \\ \mathbf{e}_a \end{bmatrix}, \tag{2.28}$$

where

$$\begin{aligned}\mathbf{e}_p &= \ln(\mathbf{T}_{k+1,k})^\vee - (t_{k+1} - t_k)\boldsymbol{\varpi}_k - \mathbf{C}_1\dot{\boldsymbol{\varpi}}_k, \\ \mathbf{e}_v &= \boldsymbol{\mathcal{J}}^{-1}_{k+1,k}\boldsymbol{\varpi}_{k+1} - \boldsymbol{\varpi}_k - \mathbf{C}_2\dot{\boldsymbol{\varpi}}_k, \\ \mathbf{e}_a &= -\frac{1}{2}(\boldsymbol{\mathcal{J}}^{-1}_{k+1,k}\boldsymbol{\varpi}_{k+1})^\curlywedge \boldsymbol{\varpi}_{k+1} + \boldsymbol{\mathcal{J}}^{-1}_{k+1,k}\dot{\boldsymbol{\varpi}}_{k+1} - \mathbf{C}_3\dot{\boldsymbol{\varpi}}_k,\end{aligned} \tag{2.29}$$

and we have defined

$$\begin{aligned}\mathbf{C}_1 &= \boldsymbol{\alpha}^{-2}(\boldsymbol{\alpha}(t_{k+1} - t_k) - \mathbf{1} + \exp(-\boldsymbol{\alpha}(t_{k+1} - t_k))), \\ \mathbf{C}_2 &= \boldsymbol{\alpha}^{-1}(\mathbf{1} - \exp(-\boldsymbol{\alpha}(t_{k+1} - t_k))), \\ \mathbf{C}_3 &= \exp(-\boldsymbol{\alpha}(t_{k+1} - t_k)).\end{aligned} \tag{2.30}$$

The covariance matrix can be computed as

$$\begin{aligned}\mathbf{Q}_k(t) &= \int_0^{\Delta t_k} \exp(\boldsymbol{A}(\Delta t_k - s))\boldsymbol{L}\mathbf{Q}_c\boldsymbol{L}^T \exp(\boldsymbol{A}(\Delta t_k - s))^T \, ds \\ &= \begin{bmatrix} 2\boldsymbol{\alpha}\boldsymbol{\sigma}^2 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & 2\boldsymbol{\alpha}\boldsymbol{\sigma}^2 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & 2\boldsymbol{\alpha}\boldsymbol{\sigma}^2 \end{bmatrix} \begin{bmatrix} \mathbf{Q}_{11} & \mathbf{Q}_{12} & \mathbf{Q}_{13} \\ \mathbf{Q}_{12}^T & \mathbf{Q}_{22} & \mathbf{Q}_{23} \\ \mathbf{Q}_{13}^T & \mathbf{Q}_{23}^T & \mathbf{Q}_{33} \end{bmatrix},\end{aligned} \tag{2.31}$$

where

$$\begin{aligned}\mathbf{Q}_{11} &= \frac{1}{2}\boldsymbol{\alpha}^{-5}(1 - e^{-2\boldsymbol{\alpha}\Delta t_k} + 2\boldsymbol{\alpha}\Delta t_k + \frac{2}{3}\boldsymbol{\alpha}^3\Delta t_k^3 - 2\boldsymbol{\alpha}^2\Delta t_k^2 - 4\boldsymbol{\alpha}\Delta t_k e^{-\boldsymbol{\alpha}\Delta t_k}), \\ \mathbf{Q}_{12} &= \frac{1}{2}\boldsymbol{\alpha}^{-4}(e^{-2\boldsymbol{\alpha}\Delta t_k} + 1 - 2e^{-\boldsymbol{\alpha}\Delta t_k} + 2\boldsymbol{\alpha}\Delta t_k e^{-\boldsymbol{\alpha}\Delta t_k} - 2\boldsymbol{\alpha}\Delta t_k + \boldsymbol{\alpha}^2\Delta t_k^2), \\ \mathbf{Q}_{13} &= \frac{1}{2}\boldsymbol{\alpha}^{-3}(1 - e^{-2\boldsymbol{\alpha}\Delta t_k} - 2\boldsymbol{\alpha}\Delta t_k e^{-\boldsymbol{\alpha}\Delta t_k}), \\ \mathbf{Q}_{22} &= \frac{1}{2}\boldsymbol{\alpha}^{-3}(4e^{-\boldsymbol{\alpha}\Delta t_k} - 3 - e^{-2\boldsymbol{\alpha}\Delta t_k} + 2\boldsymbol{\alpha}\Delta t_k), \\ \mathbf{Q}_{23} &= \frac{1}{2}\boldsymbol{\alpha}^{-2}(e^{-2\boldsymbol{\alpha}\Delta t_k} + 1 - 2e^{-\boldsymbol{\alpha}\Delta t_k}), \\ \mathbf{Q}_{33} &= \frac{1}{2}\boldsymbol{\alpha}^{-1}(1 - e^{-2\boldsymbol{\alpha}\Delta t_k}).\end{aligned}$$

2.4.2 Querying the Trajectory

Because the prior we formulated is in continuous time, we now have the advantage of being able to interpolate for a state estimate at any given time. Suppose we have states at times t_k and t_{k+1} but want to query the state at time τ where $t_k < \tau < t_{k+1}$. This can be done using the results from [3]:

$$\begin{bmatrix} \boldsymbol{\xi}_k(\tau) \\ \dot{\boldsymbol{\xi}}_k(\tau) \\ \ddot{\boldsymbol{\xi}}_k(\tau) \end{bmatrix} = \boldsymbol{\gamma}_k(\tau) = \boldsymbol{\Lambda}(t)\boldsymbol{\gamma}_k(t_k) + \boldsymbol{\Omega}(t)\boldsymbol{\gamma}_k(t_{k+1}), \tag{2.32}$$

where $\mathbf{\Lambda}(\tau) \in \mathbb{R}^{18\times 18}$ and $\mathbf{\Omega}(\tau) \in \mathbb{R}^{18\times 18}$ are [8]

$$\begin{aligned}\mathbf{\Lambda}(\tau) &= \mathbf{\Phi}(\tau, t_k) - \mathbf{\Omega}(\tau)\mathbf{\Phi}(t_{k+1}, t_k),\\ \mathbf{\Omega}(\tau) &= \mathbf{Q}_k(\tau)\mathbf{\Phi}(t_{i+1}, \tau)^T\mathbf{Q}_k(t_{k+1})^{-1}.\end{aligned} \tag{2.33}$$

Using the relationship between the local and global state variables, we have that

$$\begin{aligned}\mathbf{T}_\tau &= \exp(\boldsymbol{\xi}_k(\tau))^\vee \mathbf{T}_k,\\ \boldsymbol{\varpi}_\tau &= \boldsymbol{\mathcal{J}}(\boldsymbol{\xi}_k(\tau))\dot{\boldsymbol{\xi}}_k(\tau).\end{aligned} \tag{2.34}$$

2.5 Hyperparameter Training

We developed a method for hyperparameter training from data in $SE(3)$. Even after applying this principled method of choosing hyperparameters for our WNOA and WNOJ priors, we found that these priors were limiting in the type of trajectories that could be accurately represented. As such, we proposed a new motion prior in Section 2.4 that has more representational power, but more parameters. The larger number of parameters greater highlights the importance of a principled hyperparameter training method, which we show how to do for $SE(3)$ in this section. We show the hyperparameter training method for the Singer prior but the WNOA and WNOJ priors were also trained with data in a similar way.

The standard approach for hyperparameter training is to minimize the negative log likelihood of the data given the parameters [54, 4]:

$$-\log p(\mathbf{y}|\mathbf{Q}_c, \boldsymbol{\alpha}) = \frac{1}{2}(\mathbf{y}-\check{\mathbf{x}})^T\mathbf{P}^{-1}(\mathbf{y}-\check{\mathbf{x}}) - \frac{1}{2}\log|\mathbf{P}| + \frac{n}{2}\log 2\pi, \tag{2.35}$$

$$\mathbf{P} = \check{\mathbf{P}}(\mathbf{Q}_c, \boldsymbol{\alpha}) + \sigma_n^2\mathbf{1}, \tag{2.36}$$

where $\mathbf{y}$ is a stacked vector of groundtruth measurements with additive noise $\mathcal{N}(\mathbf{0}, \sigma^2\mathbf{1})$, $\check{\mathbf{x}}$ is a stacked vector of the mean function evaluated at the groundtruth measurement times and $\check{\mathbf{P}}$ is the covariance matrix associated with $\check{\mathbf{x}}$ and generated using the hyperparameters, $\mathbf{Q}_c$ and $\boldsymbol{\alpha}$.

This cost function in this form does not lend itself nicely to training since our states are no longer vectors, but are in $SE(3)$. However, we can instead write the objective function of the prior in factor form as

$$-\log p(\mathbf{y}|\mathbf{Q}_c, \boldsymbol{\alpha}) = \frac{1}{2}\mathbf{e}^T\mathbf{Q}^{-1}\mathbf{e} + \frac{1}{2}\log|\mathbf{Q}| + \frac{n}{2}\log 2\pi, \tag{2.37}$$

where $\mathbf{e}$ is a stacked vector of error terms from (2.28) composed with the groundtruth measurements and $\mathbf{Q}$ is the block-diagonally stacked $\mathbf{Q}_k$ terms from (2.31).

By making this simple modification to the objective function, we are able to train hyperparameters in $SE(3)$ with noise-free groundtruth measurements. However, the process of incorporating noisy groundtruth measurements in $SE(3)$ is slightly more involved. We must incorporate the noise coming from the measurements to obtain a new value of $\mathbf{Q}$ to be used in the objective function in (2.37), otherwise the noise in the measurements will be attributed to the process noise, thus inflating the estimate of $\mathbf{Q}_c$.

Incorporating noise into the error equations and taking the approximation that process noise, measurement noise, and timesteps are small, we see that

$$\mathbf{e}_{p_k} \approx \boldsymbol{\epsilon}_{p_{k+1}} - \boldsymbol{\epsilon}_{p_k} - (t_{k+1} - t_k)\boldsymbol{\epsilon}_{v_i} - \mathbf{C}_1\boldsymbol{\epsilon}_{a_k} + \boldsymbol{\xi}_{p_k}, \tag{2.38}$$

$$\mathbf{e}_{v_k} \approx \boldsymbol{\epsilon}_{v_{k+1}} - \boldsymbol{\epsilon}_{v_k} - \mathbf{C}_2\boldsymbol{\epsilon}_{a_k} + \boldsymbol{\xi}_{v_k}, \tag{2.39}$$

$$\mathbf{e}_{a_k} \approx \boldsymbol{\epsilon}_{a_{k+1}} - \mathbf{C}_3\boldsymbol{\epsilon}_{a_k} + \boldsymbol{\xi}_{a_k}, \tag{2.40}$$

where

$$\begin{bmatrix}\boldsymbol{\epsilon}_{p_k}\\ \boldsymbol{\epsilon}_{v_k}\\ \boldsymbol{\epsilon}_{a_k}\end{bmatrix} = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix}\mathbf{R}^k_{pp} & \mathbf{R}^k_{pv} & \mathbf{R}^k_{pa}\\ \mathbf{R}^k_{vp} & \mathbf{R}^k_{vv} & \mathbf{R}^k_{va}\\ \mathbf{R}^k_{ap} & \mathbf{R}^k_{av} & \mathbf{R}^k_{aa}\end{bmatrix}\right) \tag{2.41}$$

is the noise on the measurements at timestep k and

$$\begin{bmatrix}\boldsymbol{\xi}_{p_k}\\ \boldsymbol{\xi}_{v_k}\\ \boldsymbol{\xi}_{a_k}\end{bmatrix} = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix}\mathbf{Q}^k_{pp} & \mathbf{Q}^k_{pv} & \mathbf{Q}^k_{pa}\\ \mathbf{Q}^k_{vp} & \mathbf{Q}^k_{vv} & \mathbf{Q}^k_{va}\\ \mathbf{Q}^k_{ap} & \mathbf{Q}^k_{av} & \mathbf{Q}^k_{aa}\end{bmatrix}\right) \tag{2.42}$$

is the process noise between timestep k and $k+1$.

We now have that

$$\begin{aligned}
\text{Cov}(\mathbf{e}_{p_k}, \mathbf{e}_{p_k}) &= \mathbf{R}_{pp}^{k+1} + \mathbf{R}_{pp}^{k} + \mathbf{C}_1 \mathbf{R}_{aa}^{k} \mathbf{C}_1^T + (t_{k+1} - t_k)\mathbf{R}_{pv}^{k} \\
&\quad + (t_{k+1} - t_k)\mathbf{R}_{vp}^{k} + (t_{k+1} - t_k)^2 \mathbf{R}_{vv}^{k} + \mathbf{Q}_{pp}^{k}, \\
\text{Cov}(\mathbf{e}_{p_k}, \mathbf{e}_{v_k}) &= (t_{k+1} - t_k)\mathbf{R}_{vv}^{k} + \mathbf{C}_1 \mathbf{R}_{aa}^{k} \mathbf{C}_2^T + \mathbf{R}_{pv}^{k+1} + \mathbf{R}_{pv}^{k} + \mathbf{Q}_{pv}^{k}, \\
\text{Cov}(\mathbf{e}_{p_k}, \mathbf{e}_{a_k}) &= \mathbf{C}_1 \mathbf{R}_{aa}^{k} \mathbf{C}_3^T + \mathbf{Q}_{pa}^{k}, \\
\text{Cov}(\mathbf{e}_{v_k}, \mathbf{e}_{v_k}) &= \mathbf{R}_{vv}^{k} + \mathbf{R}_{vv}^{k+1} + \mathbf{C}_2 \mathbf{R}_{aa}^{k} \mathbf{C}_2^T + \mathbf{Q}_{vv}^{k}, \\
\text{Cov}(\mathbf{e}_{v_k}, \mathbf{e}_{a_k}) &= \mathbf{C}_2 \mathbf{R}_{aa}^{k} \mathbf{C}_3^T + \mathbf{Q}_{va}^{k}, \\
\text{Cov}(\mathbf{e}_{a_k}, \mathbf{e}_{a_k}) &= \mathbf{C}_3 \mathbf{R}_{aa}^{k} \mathbf{C}_3^T + \mathbf{R}_{aa}^{k+1} + \mathbf{Q}_{aa}^{k}, \\
\text{Cov}(\mathbf{e}_{p_{k+1}}, \mathbf{e}_{p_k}) &= -\mathbf{R}_{pp}^{k+1} - (t_{k+2} - t_{k+1})\mathbf{R}_{vp}^{k+1}, \\
\text{Cov}(\mathbf{e}_{p_{k+1}}, \mathbf{e}_{v_k}) &= -(t_{k+2} - t_{k+1})\mathbf{R}_{vv}^{k+1} - \mathbf{R}_{pv}^{k+1}, \\
\text{Cov}(\mathbf{e}_{p_{k+1}}, \mathbf{e}_{a_k}) &= -\mathbf{C}_1 \mathbf{R}_{aa}^{k+1}, \\
\text{Cov}(\mathbf{e}_{v_{k+1}}, \mathbf{e}_{p_k}) &= -\mathbf{R}_{vp}^{k+1}, \\
\text{Cov}(\mathbf{e}_{v_{k+1}}, \mathbf{e}_{v_k}) &= -\mathbf{R}_{vv}^{k+1}, \\
\text{Cov}(\mathbf{e}_{v_{k+1}}, \mathbf{e}_{a_k}) &= -\mathbf{C}_2 \mathbf{R}_{aa}^{k+1}, \\
\text{Cov}(\mathbf{e}_{a_{k+1}}, \mathbf{e}_{a_k}) &= -\mathbf{C}_3 \mathbf{R}_{aa}^{k+1}.
\end{aligned}$$

Putting this all together, we arrive at the final expression for $\mathbf{Q}$ in our objective function from (2.37) when our groundtruth measurements are noisy:

$$\mathbf{Q} = \begin{bmatrix}
\boldsymbol{\Sigma}_{1,1} & \boldsymbol{\Sigma}_{2,1}^T & & \\
\boldsymbol{\Sigma}_{2,1} & \boldsymbol{\Sigma}_{2,2} & \boldsymbol{\Sigma}_{3,2}^T & \\
 & \boldsymbol{\Sigma}_{3,2} & \ddots & \ddots \\
 & & \ddots &
\end{bmatrix}, \tag{2.43}$$

where

$$\boldsymbol{\Sigma}_{i,j} = \begin{bmatrix} \text{Cov}(\mathbf{e}_{p_i}, \mathbf{e}_{p_j}) & \text{Cov}(\mathbf{e}_{p_i}, \mathbf{e}_{v_j}) & \text{Cov}(\mathbf{e}_{p_i}, \mathbf{e}_{a_j}) \\ \text{Cov}(\mathbf{e}_{p_i}, \mathbf{e}_{v_j})^T & \text{Cov}(\mathbf{e}_{v_i}, \mathbf{e}_{v_j}) & \text{Cov}(\mathbf{e}_{v_i}, \mathbf{e}_{a_j}) \\ \text{Cov}(\mathbf{e}_{p_i}, \mathbf{e}_{a_j})^T & \text{Cov}(\mathbf{e}_{v_i}, \mathbf{e}_{a_j})^T & \text{Cov}(\mathbf{e}_{a_i}, \mathbf{e}_{a_j}) \end{bmatrix}. \tag{2.44}$$

Because our final expression for $\mathbf{Q}$ is block-tridiagonal, we are still able to exploit sparsity in hyperparameter training when evaluating our cost function.

2.6 Toy Example Simulation

2.6.1 Hyperparameter Training

In this section, we validate our hyperparameter training method on a simple simulated toy example. We start by randomly sampling 100 trajectories from a WNOA prior with a known power spectral density matrix, $\mathbf{Q}_c$. Our state is position and velocity. In our first experiment, we use noise-free measurements of the groundtruth state as training data to try and learn a $\mathbf{Q}_c$ value. We sample another 100 trajectories as our test data and overlay the covariance bounds calculated from the learned $\mathbf{Q}_c$ to see if it is consistent with the groundtruth test data. In Figure 2.2, we show these plots for both position and velocity for a single dimension. The coloured lines are the sampled states while the thick dashed red lines are the 3σ covariance bounds, which are dependant on the learned $\mathbf{Q}_c$ value. Based on the calculated covariance bounds, we can see that our learned $\mathbf{Q}_c$ matches the data quite well.

In reality, groundtruth measurements are corrupted by some noise due to imperfect sensors. In our second experiment, we add zero mean Gaussian noise with a standard deviation of 0.0001 to our position and velocity measurements of the groundtruth states. We compare naively learning $\mathbf{Q}_c$ without accounting for the noise in the groundtruth measurements to our training method accounting for noisy measurements. In Figure 2.3(a), we can see that for the naive training method, $\mathbf{Q}_c$ is significantly overestimated as the noise from the measurements is being attributed to the process noise. We then learn $\mathbf{Q}_c$ using our new training method, where we account for noisy measurements and show the

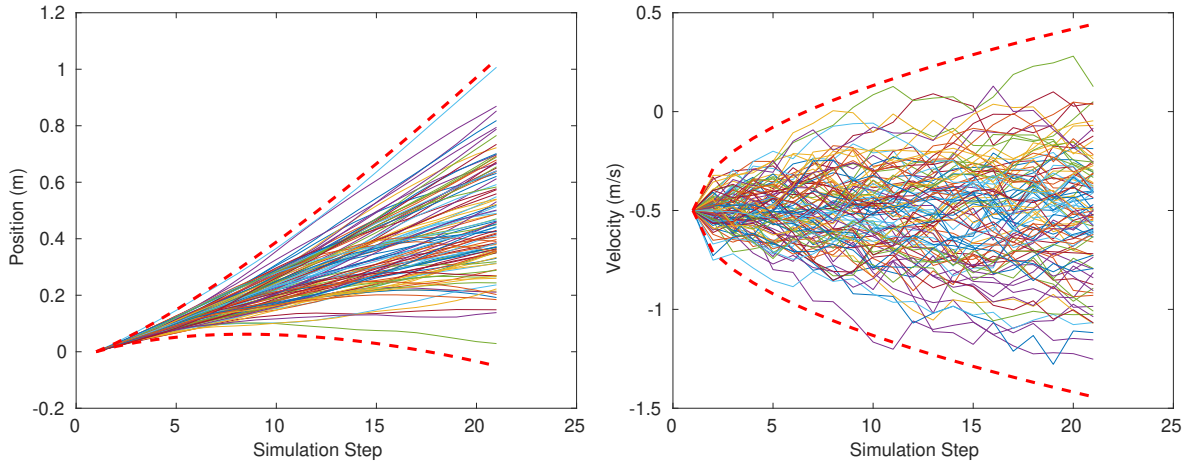


Figure 2.2: Position and velocity samples (coloured lines) overlaid with the 3σ covariance bounds (dashed red lines) calculated based on the learned value of $\mathbf{Q}_c$.

results in Figure 2.3(b). As can be seen, we are able to learn a $\mathbf{Q}_c$ value that is consistent with our groundtruth data, even with noisy measurements.

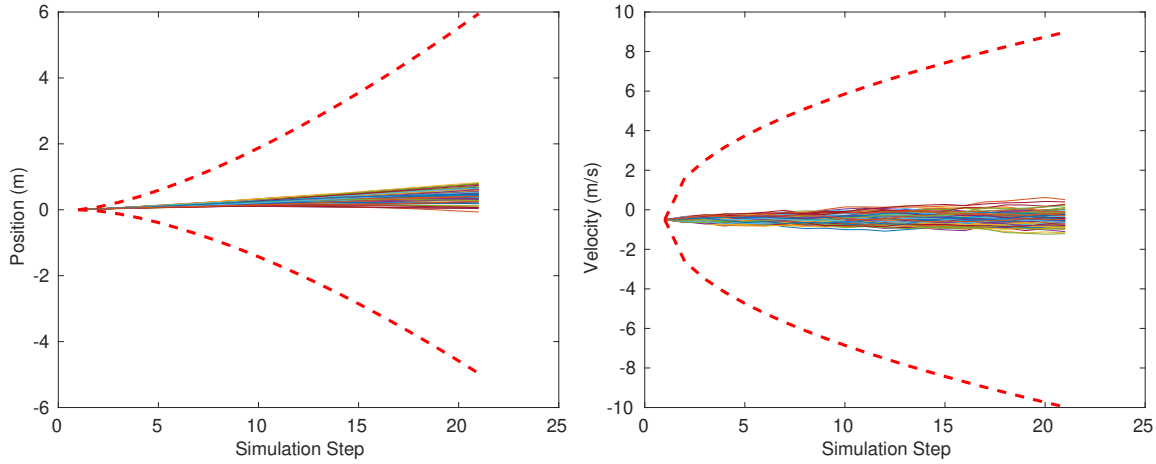
We conduct another experiment where we see how increasing measurement noise affects the quality of hyperparameter training for both the WNOA and WNOJ priors. We again compare training with and without taking the measurement noise into consideration. To make comparison easier, we define the following error metric:

$$\text{Normalized Error} = \frac{\|\mathbf{Q}_{c_{learn}} - \mathbf{Q}_{c_{true}}\|_F}{\|\mathbf{Q}_{c_{true}}\|_F}, \tag{2.45}$$

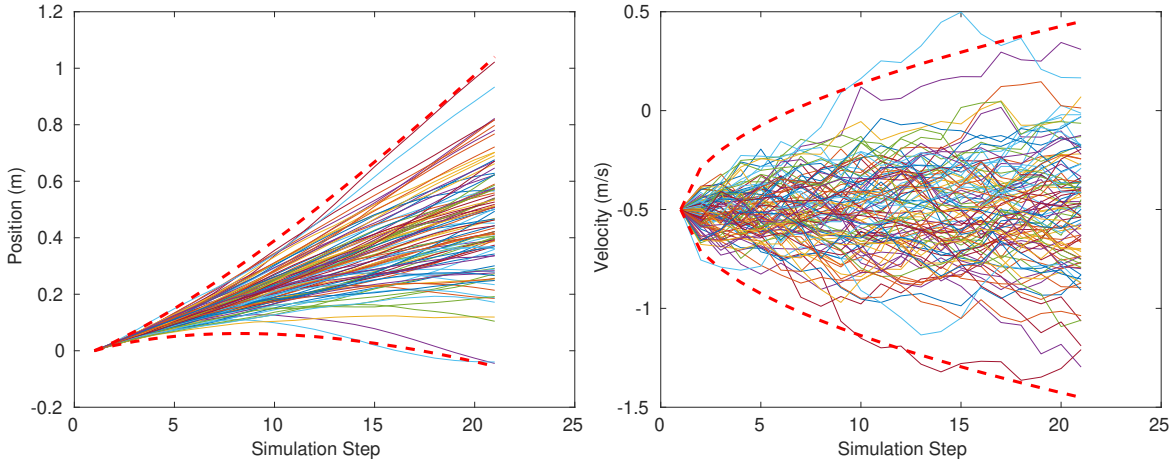
where $\mathbf{Q}_{c_{learn}}$ is the learned value of $\mathbf{Q}_c$, $\mathbf{Q}_{c_{true}}$ is the true value of $\mathbf{Q}_c$, and $\|\mathbf{X}\|_F$ is the Frobenius norm of $\mathbf{X}$. Figure 2.4 shows how the normalized error in our learned $\mathbf{Q}_c$ value increases with increasing noise on measurements for both methods with a WNOA prior. Figure 2.5 shows the exact same plot but with a WNOJ prior. We see that not accounting for noise in measurements has a big impact on our ability to accurately learn $\mathbf{Q}_c$.

2.6.2 Flexibility of Singer Prior

In this section, we will again be working with sampled trajectories from motion priors with a known $\mathbf{Q}_c$ value. We have shown that we are able to learn suitable parameters



(a) Don't account for noisy measurements in training



(b) Account for for noisy measurements in training by modifying objective function

Figure 2.3: Comparison of sampled states (coloured lines) overlaid with the 3σ covariance bounds (dashed red lines) calculated based on the learned value of $\mathbf{Q}_c$ for the cases **(a)** when we don't account for measurement noise and **(b)** when we account for measurement noise.

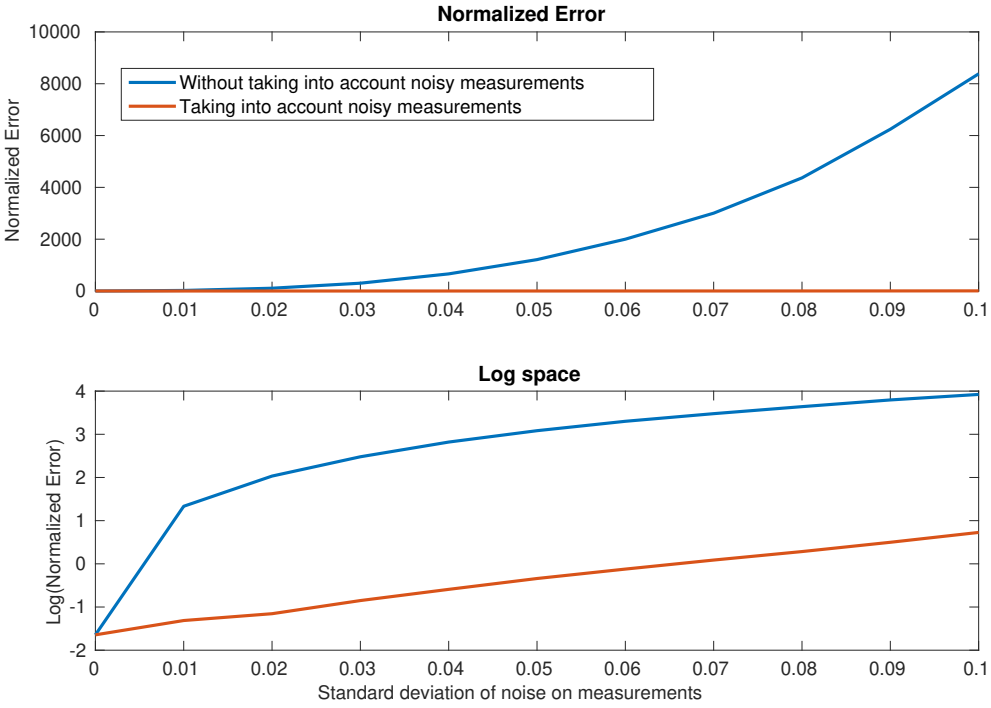


Figure 2.4: Comparison of errors in $\mathbf{Q}_c$ when training with noisy measurements for WNOA.

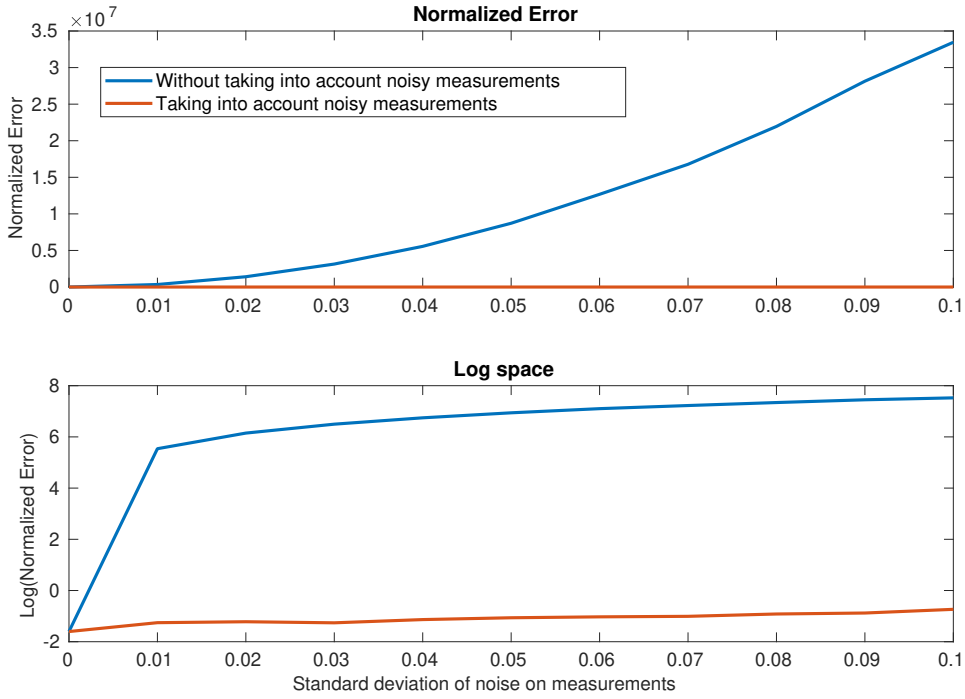


Figure 2.5: Comparison of errors in $\mathbf{Q}_c$ when training with noisy measurements for WNOJ.

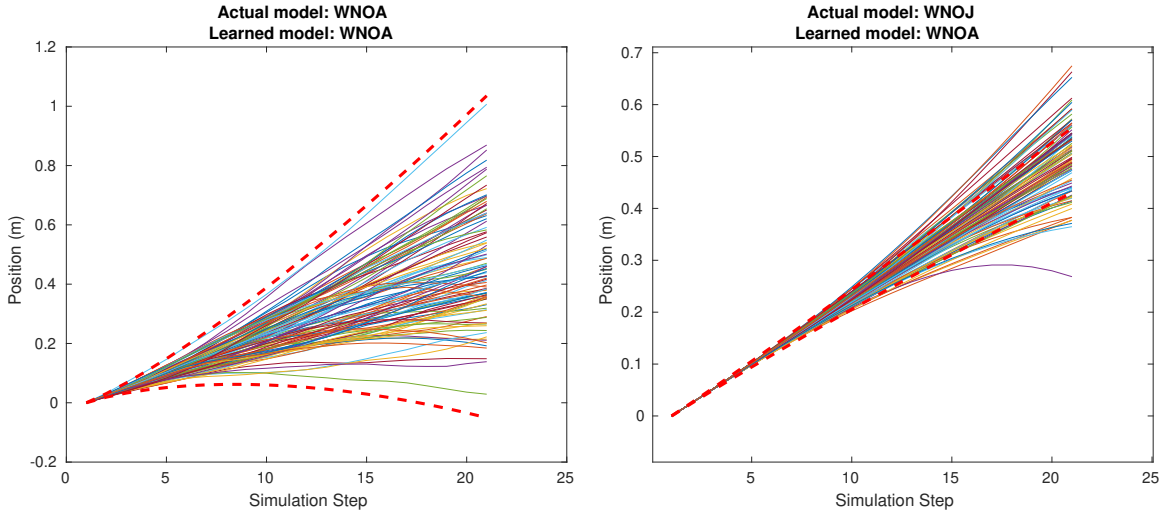
from data when we know what kind of prior the data was sampled from. In the real world, robot trajectories are not generated from these kinds of priors. As such, we want to ensure that our selected prior is appropriate for many different types of motions.

To get a taste of what this may look like before actually moving onto using real world robot data, we will show the results for training a WNOJ prior on data sampled from a WNOA prior and vice versa. The results are shown in Figure 2.6 where the left column is trajectories sampled from a WNOA prior and the right column is trajectories sampled from a WNOJ prior. Each row represents one of the models (WNOA, WNOJ or Singer) we wish to fit to the sampled data (WNOA or WNOJ). If the learned model fits the data well, the 3σ covariance bounds should be consistent with the sampled data. Due to the fact that the WNOA and WNOJ are two different models, we see that we are unable to achieve a good model fit even when training with data. We then show that the Singer prior is able to represent both types of trajectories.

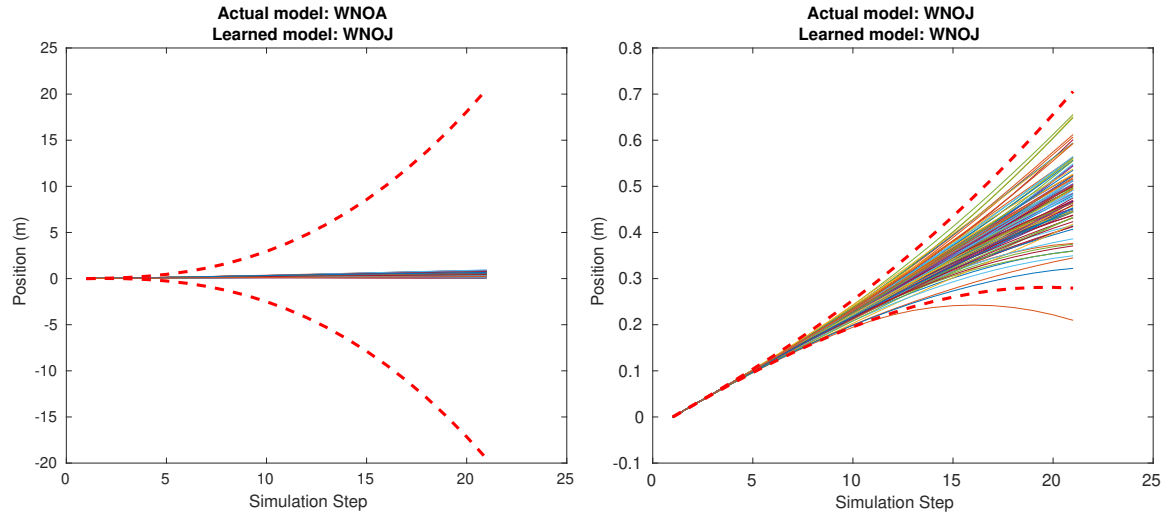
2.7 Experimental Validation

To evaluate our motion priors, we will be working with a dataset consisting of 36 km of driving with both Velodyne VLS-128 lidar data and groundtruth from an Applanix POS-LV system. The experimental setup also includes hardware time synchronization between the lidar and the POS-LV system. The test vehicle used for the dataset collection is shown in Figure 2.7.

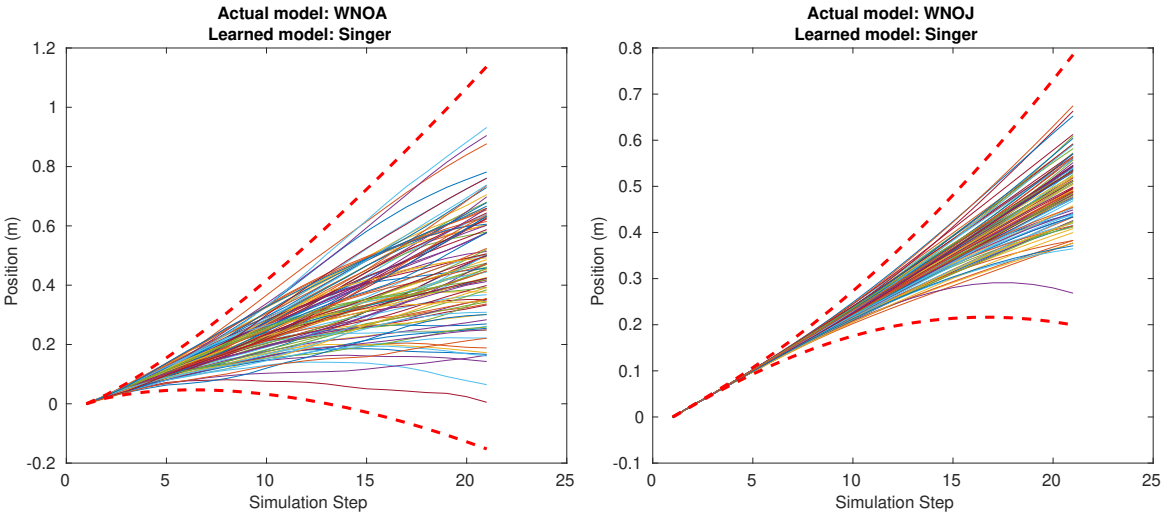
The dataset consists of the 16 km long Route A, shown in Figure 2.8(a), and the 20 km long Route B, shown in Figure 2.8(b). Each of the three motion priors were trained using the method from Section 2.5 with the POS groundtruth data from Route A. Table 2.1 shows the number of parameters for each model along with the number of iterations it took for hyperparameter training to converge. While the Singer model takes the most iterations to converge, hyperparameter training is a procedure with an upfront cost that only needs to be done once for each robotic platform before live operation.



(a) Learning a WNOA prior



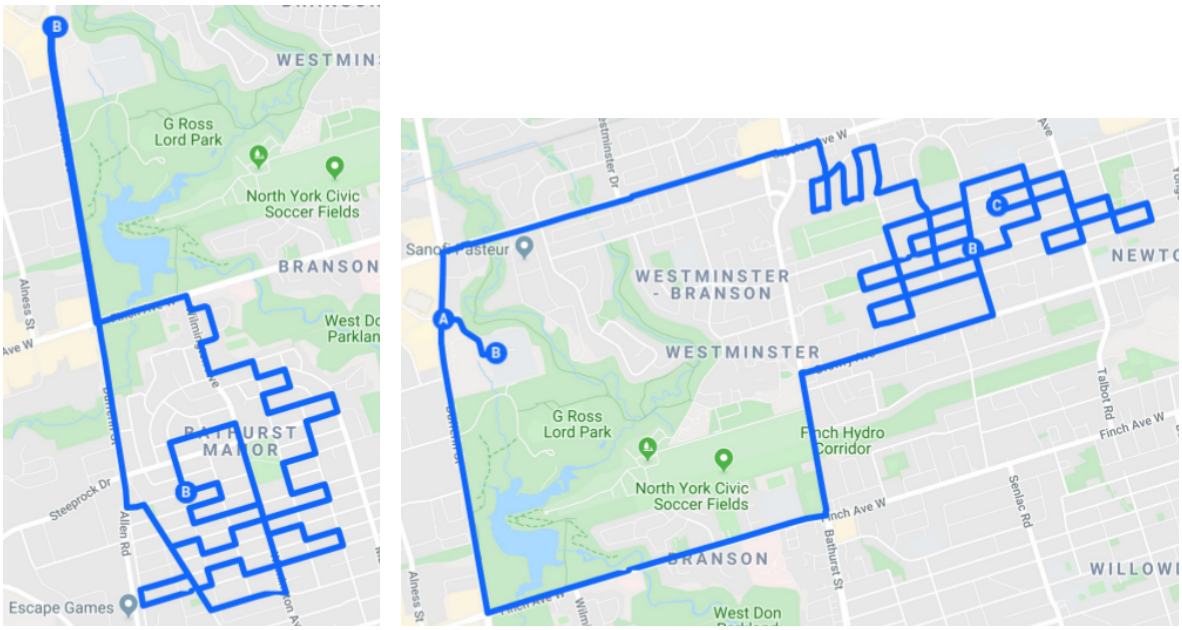
(b) Learning a WNOJ prior



(c) Learning a Singer prior

Figure 2.6: The Singer model is able to represent trajectories sampled from both a WNOA and WNOJ prior.

Figure 2.7: The Buick test vehicle used for data collection. The vehicle is equipped with a Velodyne VLS-128 lidar, and an Applanix POS-LV system.



(a) Route A **(b)** Route B

Figure 2.8: **(a)** 16 km long training route. **(b)** 20 km long test route.

Table 2.1: Number of parameters compared with number of iterations to convergence for each of the three motion priors.

Prior	WNOA	WNOJ	Singer
# of Parameters	6	6	12
# of Iterations to Convergence	26	48	178

2.7.1 Lidar Localization

We perform lidar localization as a batch trajectory optimization using each of the three trained motion priors on Route B, our 20 km long test trajectory. We use data from one of the runs along Route B to create a map of the area and then use a different run of Route B to perform lidar localization against the previously created map. We obtain 6-DOF pose estimates along with their covariances at 10 Hz from a lidar-only localization pipeline provided by Applanix. Because this pipeline is a commercial product that has been vigorously validated internally, we can assume that the covariance estimates are reasonable. We treat the pose estimates from lidar localization as pose measurements in our continuous-time estimator where we incorporate one of our motion priors. To ensure a fair comparison, all aspects of the estimator except for the motion prior are kept the same. An overview of this test procedure is shown in Figure 2.9.

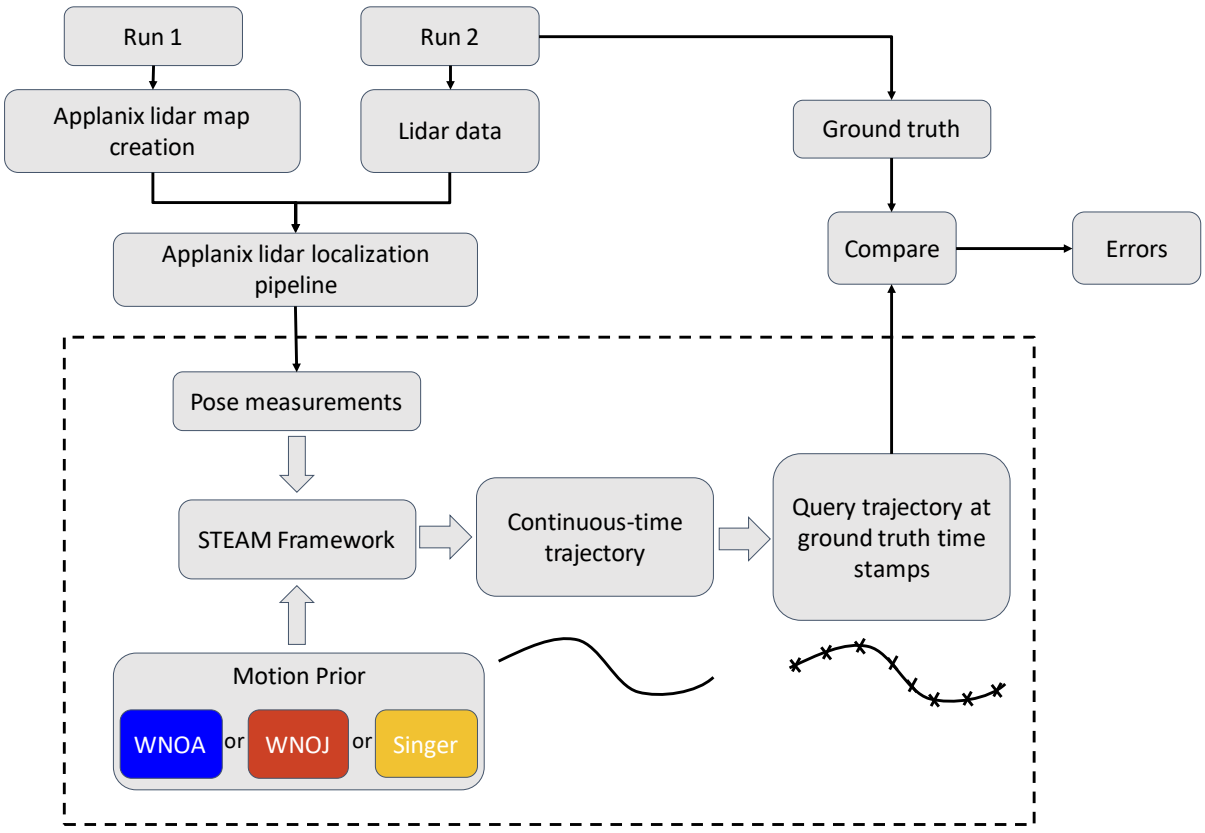


Figure 2.9: Overview of the lidar localization test procedure where one run of Route B is used to create a high-definition lidar map while a seperate run is used to localize against that map.

Table 2.2: Translational errors evaluated on lidar localization along Route B for the three motion priors and the reduction in error achieved by the Singer prior.

Seq. no.	WNOA (m)	WNOJ (m)	Singer (m)	Reduction in error from WNOA	Reduction in error from WNOJ
0	0.0690	0.0729	0.0677	**1.85%**	**7.01%**
1	0.0888	0.0892	0.0835	**5.95%**	**6.34%**
2	0.4071	0.3984	0.3999	**1.76%**	-0.38%
3	0.1947	0.1683	0.1667	**14.36%**	**0.93%**
4	0.2868	0.2686	0.2655	**7.43%**	**1.17%**
5	0.5703	0.5474	0.5471	**4.07%**	**0.05%**
6	0.3292	0.2850	0.2863	**13.03%**	-0.45%
7	0.2207	0.2224	0.2146	**2.74%**	**3.50%**
8	0.1115	0.1182	0.1126	-0.95	**4.77%**
9	0.0979	0.1050	0.0979	-0.02%	**6.73%**
overall	0.2376	0.2275	0.2242	**5.64%**	**1.47%**

Since part of evaluating a motion prior is the quality of interpolation, we also interpolate at the groundtruth timesteps since they occur more frequently than the lidar localization pose measurements. This also allows us to directly calculate localization errors.

We break down the test trajectory into 10 sequences and evaluate the performance of lidar localization on each of the sequences. The translational errors are shown in Table 2.2 where we see that the Singer prior results in an overall reduction of error by 5.64% compared to WNOA and 1.47% compared to WNOJ. The errors in longitudinal velocity are shown in Table 2.3, where we see that the Singer prior results in an overall reduction of error by 22.18% compared to WNOA and 2.32% compared to WNOJ.

Because the pose measurements we obtain from lidar localization are reliably accurate relative to groundtruth, we perform another experiment where we decrease the frequency at which we receive pose measurements from lidar localization but use the same motion prior hyperparameters trained from groundtruth measurements every 0.1s. In our new experiment, instead of receiving pose measurements every 0.1s from lidar localization, we increase measurement dropout to 1s and all the way up to 5s. The results from this experiment are shown in Figure 2.10. With a 5s measurement dropout, the Singer prior has a 29.57% reduction in translational error compared to WNOA and 67.89% compared

Table 2.3: Longitudinal velocity errors evaluated on lidar localization along Route B for the three motion priors and the reduction in error achieved by the Singer prior.

Seq. no.	WNOA (m/s)	WNOJ (m/s)	Singer (m/s)	Reduction in error from WNOA	Reduction in error from WNOJ
0	0.2561	0.1794	0.2010	**21.53%**	-12.04%
1	0.1585	0.1123	0.1216	**23.26%**	-8.25%
2	0.1656	0.1341	0.1332	**19.57%**	**0.65%**
3	0.1590	0.1109	0.1120	**29.55%**	-1.04%
4	0.1427	0.1342	0.1157	**18.88%**	**3.75%**
5	0.2279	0.1477	0.1452	**36.31%**	**1.69%**
6	0.2484	0.1642	0.1505	**39.42%**	**8.35%**
7	0.1655	0.1733	0.1744	-5.34%	-0.59%
8	0.1370	0.1744	0.1371	-0.05%	**21.39%**
9	0.1988	0.1511	0.1565	**21.28%**	-3.59%
overall	0.1860	0.1482	0.1447	**22.18%**	**2.32%**

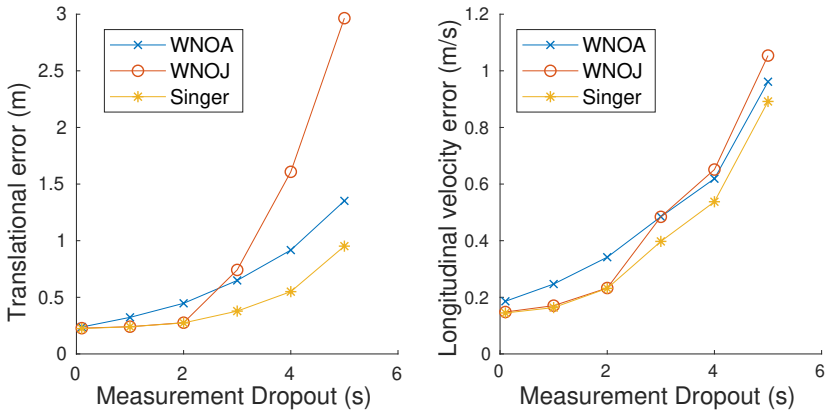


Figure 2.10: Estimation errors for each of the three priors as measurement dropout is increased.

to WNOJ. The Singer prior also decreased longitudinal velocity error by 7.25% for WNOA and 15.38% for WNOJ.

While the WNOJ prior outperforms the WNOA prior and is closely comparable to the Singer prior without measurement dropout, Figure 2.10 shows it is not robust to increasing measurement dropout. This is because when the model chosen to represent trajectories cannot sufficiently represent the true model, hyperparameter training is more sensitive to the frequency of the groundtruth measurements. The trained parameters work the best when the frequency of the groundtruth training data is close to the frequency of the measurement test data. It was noted in [48] that the WNOJ prior is

more sensitive to the hyperparameters chosen than the WNOA prior, which is consistent with our findings that the effect of measurement dropout on the WNOA prior is not as pronounced.

It is well known that if a model is too expressive, it may overfit to the training data and generalize poorly to new data. However, while the Singer model is more expressive, it is capable of a better fit to typical vehicle trajectories without overfitting to the training data. As a result, we see that the Singer model is able to maintain its performance advantage over the WNOA prior. Thus another advantage of the Singer prior over the WNOJ prior is that it is more robust to the frequency of measurements, which is desirable in a continuous-time estimation framework where the measurement frequency of the estimator does not need to be known beforehand.

As stated in Section 2.4, the WNOA prior assumes accelerations are uncorrelated with time while the WNOJ prior assumes that accelerations are correlated to accelerations at all other times. Both these assumptions are unrealistic because typical robot maneuvers will have accelerations correlated for a certain period of time (such as a car executing a turn). As such, the Singer prior allows the length-scale of acceleration to be adjusted based on what we learn from data.

Figure 2.11 shows a curved portion of sequence 5 with a 4s measurement dropout, where dotted red lines indicate the boundary at which the estimate of the car would cross the lane markings. We see that the Singer model is able to keep the estimate within the lane markings while both the WNOA and WNOJ estimates deviate outside of the markings for a short period of time. Taking a look at Figure 2.12, which shows the estimated velocities for another section of sequence 5 with a 4s measurement dropout, we can see how the velocity estimates are much better using the Singer prior as the peaks are better matched to the groundtruth. These peaks represent changes from acceleration to deceleration, which occur frequently in urban driving scenarios. The WNOA prior struggles the most to capture these motions while the Singer prior does the best.

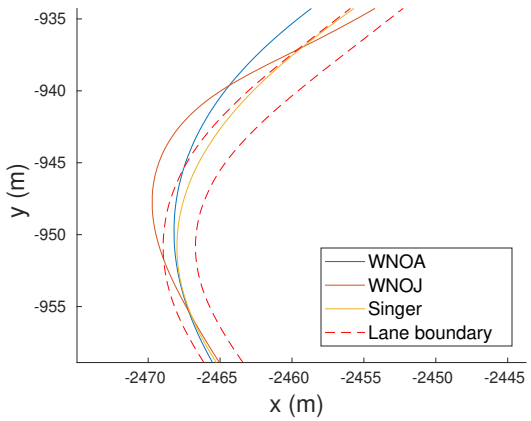


Figure 2.11: The estimator using the Singer prior keeps the estimates within the lane boundaries while the WNOA and WNOJ estimates deviate outside.

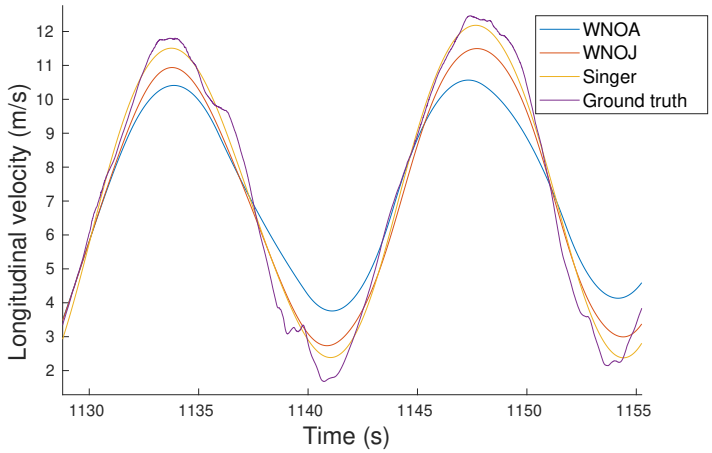


Figure 2.12: The estimator using the Singer prior is able to better capture the peaks in longitudinal velocity which represent changes in acceleration.

2.7.2 Lidar Odometry

We also evaluate our continuous-time motion priors with their trained hyperparameters on Route B, using the lidar-only odometry implementation from [47] and [48]. In this approach, a sliding-window is used rather than a batch optimization, where each window contains a reference point cloud consisting of 3 individual point clouds (where each is a single revolution of the lidar) and 2 target point clouds we are trying to align. We make use of the continuous-time interpolation scheme to handle point cloud motion distortion. We choose not to use the KITTI dataset because the point clouds have already been

Table 2.4: Percent translation errors evaluated on lidar odometry along Route B and the reduction in error achieved by the Singer prior.

Seq. no.	WNOA	WNOJ (with patch)	WNOJ (without patch)	Singer	Reduction in error from WNOA	Reduction in error from WNOJ (with patch)	Reduction in error from WNOJ (without patch)
0	3.26%	3.23%	3.18%	3.13%	**4.14%**	**3.10%**	**1.54%**
1	2.89%	2.86%	2.85%	2.87%	**0.71%**	-0.20%	-0.52%
2	2.41%	2.39%	2.43%	2.41%	**0.08%**	-0.74%	**0.90%**
3	2.26%	2.25%	-	2.28%	-0.95%	-1.34%	-
4	2.37%	2.40%	-	2.39%	-0.83%	**0.31%**	-
5	2.45%	2.40%	2.41%	2.38%	**2.75%**	**0.66%**	**0.91%**
6	2.56%	2.62%	2.57%	2.57%	-0.65%	**1.77%**	-0.01%
7	3.37%	3.33%	3.34%	3.34%	**0.72%**	-0.33%	-0.10%
8	3.57%	3.48%	3.48%	3.53%	**1.23%**	-1.39%	-1.30%
9	3.21%	3.17%	-	3.22%	-0.10%	-1.50%	-
overall (without 3,4,9)	2.929%	2.901%	2.894%	2.890%	**1.35%**	**0.39%**	**0.14%**
overall	2.834%	2.812%	-	2.811%	**0.82%**	**0.03%**	-

processed by the dataset authors to compensate for motion distortion.

Because of numerical instabilities, the WNOJ estimator in [48] required a patch that reverted to using a WNOA prior for a single window if any abnormalities were detected, such as a sudden increase in acceleration. We run lidar odometry with and without the patch and report results for both. Table 2.4 shows the average percent translation errors over path segments of lengths $100, 200, \ldots, 800$ meters for each sequence, the same evaluation metric used in the KITTI odometry benchmark [18]. The WNOJ estimator without the patch failed for sequences 3, 4, and 9. Excluding these sequences, overall, the Singer prior decreases error by 1.35% from the WNOA prior and 0.14% from the WNOJ prior without the patch.

Figure 2.13 shows the lidar odometry estimates for each of the motion priors for sequence 2, where the estimates for the WNOJ prior are without the patch. It can be seen that the estimator with the Singer prior reduces drift compared to the other two priors, most notably in the z direction.

We observe that the improvements in estimator accuracy with our lidar odometry

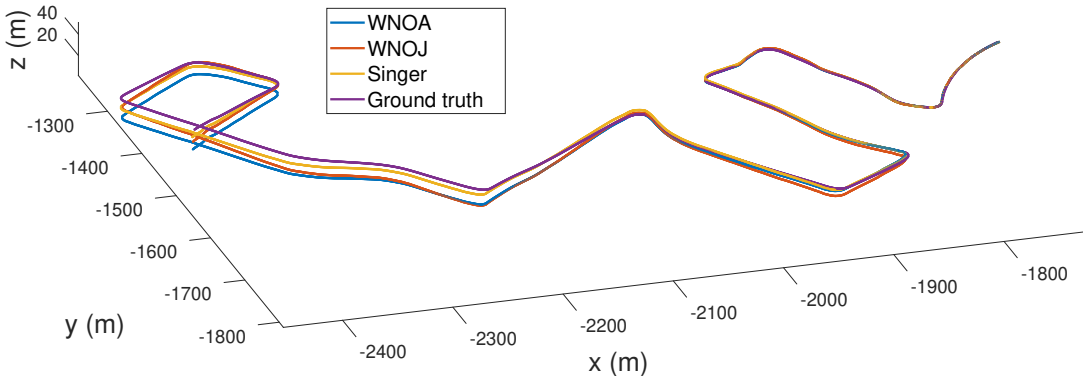


Figure 2.13: 3D plot of odometry estimates for sequence 2 show that the estimator using the Singer prior is able to reduce drift in the z direction.

experiment are not as significant compared to our lidar localization experiment. One major difference is that in lidar odometry, while the hyperparameters of the motion prior were trained in a principled way, the measurement covariances of the point cloud alignment, which are also hyperparameters of the estimator, were hand-picked. On the other hand, the pose measurements in lidar localization obtained from the Applanix pipeline have covariances attached, which were estimated in a principled manner.

It should be noted that the estimator using the Singer prior is able to run on all sequences without any patches, showing its increased robustness. On all sequences, the Singer prior decreases error overall by 0.82% from the WNOA prior and maintains a very similar performance as the WNOJ prior, which used the patch.

2.8 Summary

In this chapter, we showed that a continuous-time trajectory estimator in $SE(3)$ using a Singer prior trained with data outperforms the existing white-noise-on-acceleration and white-noise-on-jerk priors, also trained with data. It also exhibits more robustness compared to the WNOJ prior in the presence of extended measurement dropouts. This is because our hyperparameter training method allows us to use richer priors with more parameters to better fit the type of trajectories that our robot undergoes.

One limitation of the work in this chapter is we made the assumption that we have

noisy groundtruth measurements of our entire state for the hyperparameter training procedure. However, on some robotic platforms, this groundtruth may not be possible to collect. In the next chapter, we focus on a general method for parameter learning in a Gaussian variational inference setting where we no longer require observations of our full state.

Chapter 3

Parameter Learning without Groundtruth

3.1 Introduction

Probabilistic state estimation is a core component of mobile robot navigation. While the estimation machinery is reasonably mature, there are robot model parameters that are difficult to determine from first principles and vary with each new platform and sensor. In the previous chapter, we showed that we could learn robot model parameters when we had noisy but complete observations of the full state. However, this is information is often not available or difficult to collect. Our vision is to develop a learning framework that allows the deployment of a robot with arbitrary sensors onboard, and have it learn the model parameters required for estimation (and planning/control) solely from the sensor data. This can be viewed as a form of nonlinear system identification, although we will approach the problem using modern machine learning techniques.

In this chapter, we show that we can learn the parameters of a nonlinear system in concert with a nonlinear batch state estimation framework, namely Exactly Sparse Gaussian Variational Inference (ESGVI) [6]. ESGVI exploits the fact that the joint likelihood between the observed measurements (data) and the latent state can be factored, which provides a family of scalable state estimation tools starting from a variational inference objective. To extend this to parameter learning, we use Expectation Maxi-

mization (EM). In the E-step, we fix all model parameters and optimize a bound on the data log-likelihood, the so-called negative Evidence Lower Bound (ELBO); this is equivalent to ESGVI latent state inference. In the M-step, we hold the latent state estimate fixed and optimize the ELBO for the parameters. Our method is general and applicable to any nonlinear system identification problem, even when the factorization of the joint likelihood has cycles (e.g., Simultaneous Localization and Mapping (SLAM)). Barfoot et al. [6] hint at the ESGVI extension to parameter learning, but do not demonstrate it in practice.

Our demonstration of parameter learning focuses on robot noise models. The noise models of the motion prior and observed measurements are often assumed to be known or tuned by trial and error. The previous chapter demonstrated parameter learning for vehicle motion priors, but required accurate and complete (i.e., observation of the complete latent state) groundtruth. However, often times, collecting such groundtruth is not possible or extremely expensive. We demonstrate the ability to learn these noise models from only noisy measurements. If groundtruth is available, we treat it simply as another (noisy) measurement that can be included in the framework. We also demonstrate that an Inverse-Wishart (IW) prior over the time-varying measurement covariances, using a Maximum A Posteriori (MAP) treatment in the variational setting, achieves outlier rejection in both parameter learning and latent state inference. We then demonstrate our parameter learning method on a real-world lidar dataset and a pose graph optimization problem created from a front-end pose graph SLAM algorithm. We show that our parameter learning method is able to handle both noisy measurements and outliers during training and testing.

In summary, the main contribution of this chapter is a detailed investigation and experimental demonstration of parameter learning as part of ESGVI. Our application focuses on trajectory estimation, where we show nonlinear system identification using noisy measurements, without groundtruth. We also show that we can achieve a robust extension of ESGVI (with an outlier rejection scheme) by placing an IW prior on our measurement covariances. We also show comparable trajectory estimation performance between learning parameters with and without groundtruth.

3.2 ESGVI with Parameter Learning

3.2.1 Variational Setup

In this section, we give an overview of ESGVI [6]. We begin with the maximum-likelihood problem for the given data, $\mathbf{z}$, which is expressed as

$$\boldsymbol{\theta}^\star = \arg\max_{\boldsymbol{\theta}} p(\mathbf{z}|\boldsymbol{\theta}), \tag{3.1}$$

where $\boldsymbol{\theta}$ represents the parameters of our system that we wish to learn.

We define the loss that we wish to minimize as the negative log-likelihood of the data and introduce the latent state, $\mathbf{x}$. Applying the usual EM decomposition results in

$$\mathscr{L} = -\ln p(\mathbf{z}|\boldsymbol{\theta}) = \underbrace{\int q(\mathbf{x}) \ln\left(\frac{p(\mathbf{x}|\mathbf{z},\boldsymbol{\theta})}{q(\mathbf{x})}\right) d\mathbf{x}}_{\leq 0} \underbrace{- \int q(\mathbf{x}) \ln\left(\frac{p(\mathbf{x},\mathbf{z}|\boldsymbol{\theta})}{q(\mathbf{x})}\right) d\mathbf{x}}_{\text{upper bound}}, \tag{3.2}$$

where we define our approximate posterior as a multivariate Gaussian distribution, $q(\mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. We now proceed iteratively in two steps, the expectation step (E-step) and the maximization step (M-step)[1].

As commonly done in the EM framework, in both the E-step and the M-step, we optimize the upper bound term in (3.2), which is also known as the (negative) Evidence Lower Bound (ELBO). Using the expression for the entropy, $-\int q(\mathbf{x}) \ln q(\mathbf{x}) d\mathbf{x}$, for a Gaussian and dropping constants, the upper bound term is written as the loss functional of ESGVI,

$$V(q|\boldsymbol{\theta}) = \mathbb{E}_q[\phi(\mathbf{x}|\boldsymbol{\theta})] + \frac{1}{2} \ln\left(|\boldsymbol{\Sigma}^{-1}|\right), \tag{3.3}$$

where we define $\phi(\mathbf{x}|\boldsymbol{\theta}) = -\ln p(\mathbf{x},\mathbf{z}|\boldsymbol{\theta})$, $\mathbb{E}_q[\cdot]$ is the expectation conditioned on the distribution $q(\mathbf{x})$, and $|\cdot|$ is the matrix determinant. We drop $\mathbf{z}$ in the notation for convenience as our expectation is over $\mathbf{x}$.

Taking the derivatives of the loss functional with respect to $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}^{-1}$, Barfoot et

[1] We are working with the negative log-likelihood so we are technically applying Expectation Minimization, but the acronym stays the same.

al. [6] developed a Newton-style iterative optimizer to update our estimate of $q(\mathbf{x})$. We summarize the optimization scheme here as

$$\left(\boldsymbol{\Sigma}^{-1}\right)^{(i+1)} = \sum_{k=1}^{K} \mathbf{P}_k^T \mathbb{E}_{q_k^{(i)}} \left[\frac{\partial^2 \phi_k(\mathbf{x}_k|\boldsymbol{\theta})}{\partial \mathbf{x}_k^T \partial \mathbf{x}_k} \right] \mathbf{P}_k, \tag{3.4a}$$

$$\left(\boldsymbol{\Sigma}^{-1}\right)^{(i+1)} \delta\boldsymbol{\mu} = -\sum_{k=1}^{K} \mathbf{P}_k^T \mathbb{E}_{q_k^{(i)}} \left[\frac{\partial \phi_k(\mathbf{x}_k|\boldsymbol{\theta})}{\partial \mathbf{x}_k^T} \right], \tag{3.4b}$$

$$\boldsymbol{\mu}^{(i+1)} = \boldsymbol{\mu}^{(i)} + \delta\boldsymbol{\mu}, \tag{3.4c}$$

where superscript i is used to denote variables at the i^{th} iteration. We have exploited the factorization of the joint log-likelihood into K factors as

$$\phi(\mathbf{x}|\boldsymbol{\theta}) = \sum_{k=1}^{K} \phi_k(\mathbf{x}_k|\boldsymbol{\theta}). \tag{3.5}$$

For generality we have each factor, ϕ_k, affected by the entire parameter set, $\boldsymbol{\theta}$, but in practice it can be a subset. $\mathbf{P}_k$ is a projection matrix that extracts $\mathbf{x}_k$ from $\mathbf{x}$ (i.e. $\mathbf{x}_k = \mathbf{P}_k\,\mathbf{x}$). The marginal of q associated with $\mathbf{x}_k$ is

$$q_k(\mathbf{x}_k) = \mathcal{N}(\mathbf{P}_k\boldsymbol{\mu}, \mathbf{P}_k\boldsymbol{\Sigma}\mathbf{P}_k^T). \tag{3.6}$$

Critical to the efficiency of the ESGVI framework is the ability to compute the required marginals in (3.4a) and (3.4b), without ever constructing the complete (dense) covariance matrix, $\boldsymbol{\Sigma}$. A sparse solver based on the method of Takahashi et al. [46] is used to achieve this in [6].

The expectations in (3.4a) and (3.4b) can be approximated using Gaussian cubature samples (e.g., sigmapoints) of the marginal posterior. Importantly, approximating the expectations at only the mean of the posterior is equivalent to the MAP batch optimization with Newton's method. Barfoot et al. [6] also provide a derivative-free optimization scheme with only Gaussian cubature.

In the M-step, we hold $q(\mathbf{x})$ fixed and optimize the upper bound for the parameters,

$\boldsymbol{\theta}$. We can optimize for $\boldsymbol{\theta}$ by taking the derivative of the loss functional as follows:

$$\begin{aligned}\frac{\partial V(q|\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} &= \frac{\partial}{\partial \boldsymbol{\theta}} \mathbb{E}_q[\phi(\mathbf{x}|\boldsymbol{\theta})] \\ &= \frac{\partial}{\partial \boldsymbol{\theta}} \mathbb{E}_q \left[\sum_{k=1}^{K} \phi_k(\mathbf{x}_k|\boldsymbol{\theta}) \right] \qquad (3.7) \\ &= \sum_{k=1}^{K} \mathbb{E}_{q_k} \left[\frac{\partial}{\partial \boldsymbol{\theta}} \phi_k(\mathbf{x}_k|\boldsymbol{\theta}) \right]. \qquad (3.8)\end{aligned}$$

In the last step, the expectation reduces from being over the full Gaussian, q, to the marginal associated with the variables in each factor, q_k. We can then set the derivative to zero and isolate $\boldsymbol{\theta}$ for a critical point, formulating an M-step. If isolation is not possible, we can use the gradient in (3.7) for a partial M-step, which is known as Generalized Expectation Maximization (GEM) [9].

3.2.2 Alternate Loss Functional

In the E-step, we hold $\boldsymbol{\theta}$ fixed and optimize $q(\mathbf{x})$ for the best possible Gaussian fit. Barfoot et al. [6] present an alternate, Gauss-Newton-style loss functional for when the negative log-likelihood takes the form

$$\phi(\mathbf{x}|\mathbf{W}) = \frac{1}{2} \left(\mathbf{e}(\mathbf{x})^T \mathbf{W}^{-1} \mathbf{e}(\mathbf{x}) - \ln(|\mathbf{W}^{-1}|) \right), \qquad (3.9)$$

where $\boldsymbol{\theta}$ is now a covariance matrix, $\mathbf{W}$. With Jensen's inequality [25] and dropping the second term since $\mathbf{W}$ is a constant in the E-step, we can write

$$\mathbb{E}_q[\mathbf{e}(\mathbf{x})]^T \mathbf{W}^{-1} \mathbb{E}_q[\mathbf{e}(\mathbf{x})] \leq \mathbb{E}_q \left[\mathbf{e}(\mathbf{x})^T \mathbf{W}^{-1} \mathbf{e}(\mathbf{x}) \right]. \qquad (3.10)$$

Motivated by this relationship, we can define a new loss functional for the E-step as

$$V'(q) = \frac{1}{2} \mathbb{E}_q[\mathbf{e}(\mathbf{x})]^T \mathbf{W}^{-1} \mathbb{E}_q[\mathbf{e}(\mathbf{x})] + \frac{1}{2} \ln(|\boldsymbol{\Sigma}^{-1}|), \qquad (3.11)$$

which is a conservative approximation of $V(q)$, appropriate for mild nonlinearities and/or concentrated posteriors. The alternate loss functional is simpler to implement in practice as it does not require the second derivative of the factors[2]. Also note how evaluating the expectation only at the mean of the posterior is equivalent to MAP Gauss-Newton.

3.3 Parameter Learning for Robot Noise Models

3.3.1 Constant Covariance

Barfoot et al. [6] outline parameter learning (M-step) for constant covariance noise models, which we summarize here. Our loss functional is

$$V(q|\mathbf{W}) = \mathbb{E}_q[\phi^m(\mathbf{x}|\mathbf{W})] + \frac{1}{2}\ln\left(|\mathbf{\Sigma}^{-1}|\right), \tag{3.12}$$

where we use ϕ^m to denote a factor involving a measurement error term with a constant covariance. This expression is similar to (3.9), but we can exploit the factorization of $\phi^m(\mathbf{x}|\mathbf{W})$ to write:

$$\begin{aligned}\phi^m(\mathbf{x}|\mathbf{W}) &= \sum_{k=1}^{K} \phi_k^m(\mathbf{x}_k|\mathbf{W}) \\ &= \sum_{k=1}^{K} \frac{1}{2}\left(\mathbf{e}_k(\mathbf{x}_k)^T\mathbf{W}^{-1}\mathbf{e}_k(\mathbf{x}_k) - \ln(|\mathbf{W}^{-1}|)\right),\end{aligned} \tag{3.13}$$

where the K factors (in practice, it could be a subset) are affected by the unknown parameter $\mathbf{W}$, a constant covariance matrix. Evaluating the derivative, as shown in (3.7), with respect to $\mathbf{W}^{-1}$ and setting to zero for computing a minimum results in the optimal $\mathbf{W}$ to be

$$\mathbf{W} = \frac{1}{K}\sum_{k=1}^{K} \mathbb{E}_{q_k}\left[\mathbf{e}_k(\mathbf{x}_k)\mathbf{e}_k(\mathbf{x}_k)^T\right], \tag{3.14}$$

[2]Another alternative is the derivative-free optimization scheme with cubature sampling, at the cost of requiring more cubature samples (i.e., more computation). A derivative-free scheme for the alternate loss functional is also possible [6].

which can be approximated with Gaussian cubature if the error functions, $\mathbf{e}_k(\mathbf{x}_k)$, are nonlinear. This method guarantees that the learned covariance is always positive definite.

Alternatively, we can choose to linearize $\mathbf{e}_k(\mathbf{x}_k)$ at the posterior marginal, $q_k = \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, resulting in the following M-step:

$$\begin{aligned}\mathbf{W} &\approx \frac{1}{K}\sum_{k=1}^{K} \mathbb{E}_{q_k}\left[\left(\mathbf{e}_k(\boldsymbol{\mu}_k) + \mathbf{E}_k\,\delta\mathbf{x}_k\right)\left(\mathbf{e}_k(\boldsymbol{\mu}_k) + \mathbf{E}_k\,\delta\mathbf{x}_k\right)^T\right] \\ &= \frac{1}{K}\sum_{k=1}^{K}\left(\mathbf{e}_k(\boldsymbol{\mu}_k)\mathbf{e}_k^T(\boldsymbol{\mu}_k) + \mathbf{E}_k\boldsymbol{\Sigma}_k\mathbf{E}_k^T\right),\end{aligned} \tag{3.15}$$

where

$$\mathbf{E}_k = \left.\frac{\partial \mathbf{e}_k(\mathbf{x}_k)}{\partial \mathbf{x}_k}\right|_{\mathbf{x}_k=\boldsymbol{\mu}_k}. \tag{3.16}$$

3.3.2 White-Noise-On-Acceleration Prior on Latent State

We next demonstrate parameter learning for the situation where the covariance of a factor is indirectly estimated through another quantity. Consider the example of a white-noise-on-acceleration (WNOA) motion prior on the latent state, where the parameter we wish to estimate is the power-spectral density matrix, $\mathbf{Q}_c$ [8]. More specifically, let us study the application of the prior in $SE(3)$ [3], which we recall from Chapter 2 is defined as:

$$\begin{aligned}\dot{\mathbf{T}}(t) &= \boldsymbol{\varpi}(t)^\wedge \mathbf{T}(t), \\ \dot{\boldsymbol{\varpi}} &= \mathbf{w}(t), \quad \mathbf{w}(t) \sim \mathcal{GP}(\mathbf{0}, \mathbf{Q}_c\delta(t-t')),\end{aligned} \tag{3.17}$$

The state at time t_k is $\mathbf{x}_k = \{\mathbf{T}_k, \boldsymbol{\varpi}_k\}$, and similarly, $\mathbf{x}_{k,k+1}$ is the state at two consecutive times, t_k and t_{k+1}.

We express the factors of our loss functional from (3.3), but with only WNOA prior factors denoted as ϕ^p, for simplicity:

$$\begin{aligned}\phi^p(\mathbf{x}|\mathbf{Q}_c) &= \sum_{k=1}^{K-1}\phi_k^p(\mathbf{x}_{k,k+1}|\mathbf{Q}_c) \\ &= \sum_{k=1}^{K-1}\frac{1}{2}\left(\mathbf{e}_{p,k}^T\mathbf{Q}_k^{-1}\mathbf{e}_{p,k} + \ln|\mathbf{Q}_k|\right),\end{aligned} \tag{3.18}$$

where $\mathbf{e}_{p,k}$ is the WNOA error term defined in (2.14). We can decompose the covariance of the prior, $\mathbf{Q}_k$, into two factors,

$$\mathbf{Q}_k = \mathbf{Q}_{\Delta t} \otimes \mathbf{Q}_c, \quad \mathbf{Q}_k^{-1} = \mathbf{Q}_{\Delta t}^{-1} \otimes \mathbf{Q}_c^{-1},$$

$$\mathbf{Q}_{\Delta t} = \begin{bmatrix} \frac{1}{3}\Delta t^3 & \frac{1}{2}\Delta t^2 \\ \frac{1}{2}\Delta t^2 & \Delta t \end{bmatrix}, \quad \mathbf{Q}_{\Delta t}^{-1} = \begin{bmatrix} 12\Delta t^{-3} & -6\Delta t^{-2} \\ -6\Delta t^{-2} & 4\Delta t^{-1} \end{bmatrix},$$

where $\otimes$ is the Kronecker product. Solving for the derivative with respect to $Q_{c_{ij}}$, the (i,j) matrix element of $\mathbf{Q}_c$, we have

$$\frac{\partial V(q|\boldsymbol{\theta})}{\partial Q_{c_{ij}}} = \frac{1}{2}\mathrm{tr}\left(\sum_{k=1}^{K-1} \mathbb{E}_{q_{k,k+1}}[\mathbf{e}_{p,k}\mathbf{e}_{p,k}^T](\mathbf{Q}_{\Delta t}^{-1} \otimes \mathbf{1}_{ij})\right) - \frac{1}{2}(K-1)\dim(\mathbf{Q}_{\Delta t})Q_{c_{ij}}, \tag{3.19}$$

where $q_{k,k+1}$ is the marginal posterior at two consecutive times, t_k and t_{k+1}. Setting the derivative to zero, the optimal estimate of our parameter is then

$$Q_{c_{ij}} = \frac{\mathrm{tr}\left(\sum_{k=1}^{K-1} \mathbb{E}_{q_{k,k+1}}[\mathbf{e}_{p,k}\mathbf{e}_{p,k}^T](\mathbf{Q}_{\Delta t}^{-1} \otimes \mathbf{1}_{ij})\right)}{\dim(\mathbf{Q}_{\Delta t})(K-1)}. \tag{3.20}$$

As explained for (3.14), the expectation in (3.20) can be approximated with Gaussian cubature or linearization.

3.3.3 Inverse-Wishart Prior on Covariance

We further extend covariance estimation by incorporating a prior over measurement covariances. Instead of treating the measurement covariance as a static parameter, we treat it as a random variable and place an IW prior on it. We then learn some of the parameters of the prior. In order to do so, we redefine our joint likelihood as

$$p(\mathbf{x}, \mathbf{z}, \mathbf{R}) = p(\mathbf{x}, \mathbf{z}|\mathbf{R})p(\mathbf{R}), \tag{3.21}$$

where now we also include the covariances, $\mathbf{R} = \{\mathbf{R}_1, \mathbf{R}_2, \dots \mathbf{R}_K\}$, as random variables. We also redefine our posterior estimate to be

$$q'(\mathbf{x}) = q(\mathbf{x})s(\mathbf{R}), \tag{3.22}$$

a product between a Gaussian $q(\mathbf{x})$ and a posterior distribution for the covariances, $s(\mathbf{R})$.

The upper bound term in the EM decomposition of (3.2) can now be written as

$$-\int\int q(\mathbf{x})s(\mathbf{R}) \ln \left(\frac{p(\mathbf{x}, \mathbf{z}|\mathbf{R})p(\mathbf{R})}{q(\mathbf{x})s(\mathbf{R})} \right) d\mathbf{x}\, d\mathbf{R}. \tag{3.23}$$

We define the posterior over the covariances as

$$s(\mathbf{R}) = \delta(\mathbf{R} - \boldsymbol{\Upsilon}), \tag{3.24}$$

where $\delta(\cdot)$ is the Dirac delta function (interpreted as a probability density function) and $\boldsymbol{\Upsilon} = \{\boldsymbol{\Upsilon}_1, \boldsymbol{\Upsilon}_2 \dots \boldsymbol{\Upsilon}_K\}$ is the set of optimal covariances. The upper bound now simplifies to

$$-\int q(\mathbf{x}) \ln \left(p(\mathbf{x}, \mathbf{z}|\boldsymbol{\Upsilon})p(\boldsymbol{\Upsilon}) \right) d\mathbf{x} + \int q(\mathbf{x}) \ln q(\mathbf{x})\, d\mathbf{x} + \underbrace{\int s(\mathbf{R}) \ln s(\mathbf{R}) d\mathbf{R}}_{\text{indep. of } \boldsymbol{\Upsilon}},$$

where we have abused notation and written $p(\mathbf{R} = \boldsymbol{\Upsilon})$ as $p(\boldsymbol{\Upsilon})$, and similarly will later write $p(\mathbf{R}_k = \boldsymbol{\Upsilon}_k)$ as $p(\boldsymbol{\Upsilon}_k)$. We view our selection of the delta function as a convenient way of showing how we can approximate a Gaussian distribution for the trajectory and a MAP approximation of the covariances in a single variational framework. The last term is the differential entropy of a Dirac delta function, and because it is independent of our variational parameter, $\boldsymbol{\Upsilon}$, we choose to drop it from our loss functional even though it approaches negative infinity.

We assume $p(\boldsymbol{\Upsilon})$ factors as $p(\boldsymbol{\Upsilon}) = \prod_{k=1}^{K} p(\boldsymbol{\Upsilon}_k)$. We apply an IW prior over our covariances by defining

$$p(\boldsymbol{\Upsilon}_k) = \frac{|\boldsymbol{\Psi}|^{\nu/2}}{2^{\frac{\nu d}{2}} \Gamma_d(\frac{\nu}{2})} |\boldsymbol{\Upsilon}_k|^{-\frac{\nu+d+1}{2}} \exp\left(-\frac{1}{2} \mathrm{tr}(\boldsymbol{\Psi} \boldsymbol{\Upsilon}_k^{-1}) \right), \tag{3.25}$$

where d is the dimension of $\boldsymbol{\Upsilon}_k$, $\boldsymbol{\Psi} \in \mathbb{R}^{d\times d} > 0$ is the scale matrix, $\nu > d-1$ is the degrees-of-freedom (DOF), and $\Gamma_d(\cdot)$ is the multivariate Gamma function. The IW distribution has been used as a prior over covariance matrices before, which led to outlier rejection at inference [5, 40], but the parameters of the prior were assumed to be known. We choose to estimate the scale matrix parameter $\boldsymbol{\Psi}$ and leave the degrees-of-freedom ν as a metaparameter.

Now we define our factors as

$$\begin{aligned} -\ln(\boldsymbol{\Upsilon}) &= \sum_{k=1}^{K} -\ln p(\boldsymbol{\Upsilon}_k) \\ &= \sum_{k=1}^{K} \phi_k^w(\boldsymbol{\Upsilon}_k|\boldsymbol{\Psi}) = \phi^w(\boldsymbol{\Upsilon}|\boldsymbol{\Psi}), \end{aligned} \tag{3.26}$$

where ϕ^w is the factor for the Inverse-Wishart prior. Dropping constant terms, the loss functional can finally be written as

$$V(q'|\boldsymbol{\Upsilon}, \boldsymbol{\Psi}) = \sum_{k=1}^{K} \mathbb{E}_{q_k}\left[\phi_k^m(\mathbf{x}_k|\boldsymbol{\Upsilon}_k) + \phi_k^w(\boldsymbol{\Upsilon}_k|\boldsymbol{\Psi})\right] + \frac{1}{2}\ln\left(|\boldsymbol{\Sigma}^{-1}|\right), \tag{3.27}$$

where

$$\phi_k^m(\mathbf{x}_k|\boldsymbol{\Upsilon}_k) = \frac{1}{2}\left(\mathbf{e}_k(\mathbf{x}_k)^T\boldsymbol{\Upsilon}_k^{-1}\mathbf{e}_k(\mathbf{x}_k) - \ln(|\boldsymbol{\Upsilon}_k^{-1}|\right), \tag{3.28}$$

$$\phi_k^w(\boldsymbol{\Upsilon}_k|\boldsymbol{\Psi}) = -\frac{\alpha-1}{2}\ln|\boldsymbol{\Upsilon}_k^{-1}| - \frac{\nu}{2}\ln|\boldsymbol{\Psi}| + \frac{1}{2}\mathrm{tr}(\boldsymbol{\Psi}\boldsymbol{\Upsilon}_k^{-1}), \tag{3.29}$$

with $\alpha = \nu + d + 2$.

In the E-step, we hold $\boldsymbol{\Psi}$ fixed and optimize for $\boldsymbol{\Upsilon}_k$, which we accomplish by taking the derivative of the loss functional as follows:

$$\frac{\partial V}{\partial \boldsymbol{\Upsilon}_k^{-1}} = \frac{1}{2}\mathbb{E}_{q_k}\left[\mathbf{e}_k(\mathbf{x}_k)\mathbf{e}_k(\mathbf{x}_k)^T\right] - \frac{1}{2}\alpha\boldsymbol{\Upsilon}_k + \frac{1}{2}\boldsymbol{\Psi}. \tag{3.30}$$

Setting the derivative to zero,

$$\begin{aligned}\boldsymbol{\Upsilon}_k &= \frac{1}{\alpha}\boldsymbol{\Psi} + \frac{1}{\alpha}\mathbb{E}_{q_k}\left[\mathbf{e}_k(\mathbf{x}_k)\mathbf{e}_k(\mathbf{x}_k)^T\right] \\ &= \frac{\alpha-1}{\alpha}\underbrace{\left(\frac{\boldsymbol{\Psi}}{\alpha-1}\right)}_{\text{IW mode}} + \frac{1}{\alpha}\mathbb{E}_{q_k}\left[\mathbf{e}_k(\mathbf{x}_k)\mathbf{e}_k(\mathbf{x}_k)^T\right],\end{aligned} \tag{3.31}$$

where we see the optimal $\boldsymbol{\Upsilon}_k$ is a weighted average between the mode of the IW distribution and the optimal static covariance estimate from (3.14) at a single marginal factor. Since our E-step in ESGVI is already iterative, we can seamlessly extend it by applying (3.31) as iteratively reweighted least squares (IRLS).

In the M-step, we hold $\boldsymbol{\Upsilon}$ fixed and optimize for $\boldsymbol{\Psi}$, which we accomplish by taking the derivative of the loss functional as follows:

$$\frac{\partial V}{\partial \boldsymbol{\Psi}} = \sum_{k=1}^{K}\left(-\frac{\nu}{2}\boldsymbol{\Psi}^{-1} + \frac{1}{2}\mathbf{R}_k^{-1}\right). \tag{3.32}$$

Setting the derivative to zero,

$$\boldsymbol{\Psi}^{-1} = \frac{1}{K\nu}\sum_{k=1}^{K}\boldsymbol{\Upsilon}_k^{-1}. \tag{3.33}$$

Applying (3.31) in the E-step and (3.33) in the M-step, we found that our optimzation scheme was still ill-posed, and our covariance estimates tended toward the positive-definite boundary (i.e., the zero matrix). We propose constraining the determinant of $\boldsymbol{\Psi}$ to be a constant β, which can be thought of as constraining the volume of the uncertainty ellipsoid of the corresponding measurements to be fixed. We accomplish this by scaling the latest $\boldsymbol{\Psi}$ update as follows:

$$\boldsymbol{\Psi}_{\text{constrained}} \leftarrow \left(\beta\,|\boldsymbol{\Psi}|^{-1}\right)^{\frac{1}{d}}\,\boldsymbol{\Psi}. \tag{3.34}$$

We then rely on the noise models of other factors (e.g., the motion prior) to adapt to our selection of β during training.

3.4 Experimental Validation

To evaluate our parameter learning method, we will be working with the vehicle dataset collected and used in Chapter 2. Recall that the dataset consists of 36 km of driving, with Velodyne VLS-128 lidar data and an Applanix POS-LV positioning system. There are two sources of 6-DOF vehicle pose measurements. The first is from the POS-LV system, which we treat as groundtruth. The second is from a lidar localization system from Applanix, which localizes the lidar data to a prebuilt high-definition map.

We use Route A, our 16 km long training set, to learn the parameters of our noise models. For inference, we perform a batch trajectory optimization on Route B, our 20 km long test set, using the learned noise model parameters of our motion prior and measurements. These results are discussed in Sections 3.4.1-3.4.3.

In Section 3.4.4, we also evaluate our method on the publicly available Bicocca 25b dataset from [33], which provides a set of odometry and loop closure constraints (represented as a pose graph) created from a bag of words place recognition system run on data collected during the Rawseeds project [14]. We use our method to jointly learn the covariances of the loop closure constraints in addition to the optimized trajectory.

3.4.1 Experiment A: Training With and Without Groundtruth

In Experiment A, our first experiment, we only use the lidar localization measurements to learn our model parameters (training without groundtruth). As a benchmark, we also learn another set of model parameters where we additionally include groundtruth poses in our training (training with incomplete groundtruth). This is different from the previous chapter where the training method required groundtruth of the entire state (training with complete groundtruth), which for our problem setup is pose and body-centric velocity. Additionally, in that chapter, the measurement covariances were assumed to be known and not learned.

The loss functional corresponding to this experiment is

$$V(q'|\boldsymbol{\Upsilon}, \boldsymbol{\Psi}, \mathbf{W}_{gt}, \mathbf{Q}_c) = \mathbb{E}_{q'}[\phi^p(\mathbf{x}|\mathbf{Q}_c) + \phi^m(\mathbf{x}|\mathbf{W}_{gt}) + \phi^m(\mathbf{x}|\boldsymbol{\Upsilon}) + \phi^w(\boldsymbol{\Upsilon}|\boldsymbol{\Psi})] + \frac{1}{2}\ln\left(|\boldsymbol{\Sigma}^{-1}|\right), \tag{3.35}$$

where $\phi^p(\mathbf{x}|\mathbf{Q}_c)$ are the WNOA prior factors, $\phi^m(\mathbf{x}|\mathbf{W}_{gt})$ are the groundtruth factors (when available), and $\phi^m(\mathbf{x}|\boldsymbol{\Upsilon})$ and $\phi^w(\boldsymbol{\Upsilon}|\boldsymbol{\Psi})$ are the lidar measurement factors with an IW prior over the covariances. See (3.18) for the definition of $\phi^p(\mathbf{x}|\mathbf{Q}_c)$ and (3.13) for the definition of $\phi^m(\mathbf{x}|\mathbf{W}_{gt})$ and $\phi^m(\mathbf{x}|\boldsymbol{\Upsilon})$. For the definition of $\phi^w(\boldsymbol{\Upsilon}|\boldsymbol{\Psi})$, see (3.26) and (3.29).

The WNOA error function (required for ϕ^p) is shown in (2.14), and the error function for pose measurements (required for ϕ^m) is defined as

$$\mathbf{e}_{m,k} = \ln(\mathbf{T}_k \mathbf{T}_{\text{meas},k}^{-1}). \tag{3.36}$$

The estimation problem in this experiment can be represented by the factor graph in Figure 3.1, where we can train with or without the groundtruth factors, which are shown inside the dashed box. For the sake of conciseness in our notation, we denote $\phi^p(\mathbf{x}_{k-1,k}|\mathbf{Q}_c)$ as $\phi^p_{\mathbf{x}_{k-1,k}|\mathbf{Q}_c}$, $\phi^m(\mathbf{x}_k|\mathbf{W}_{gt})$ as $\phi^m_{\mathbf{x}_k|\mathbf{W}_{gt}}$, $\phi^m(\mathbf{x}_k|\boldsymbol{\Upsilon}_k)$ as $\phi^m_{\mathbf{x}_k|\boldsymbol{\Upsilon}_k}$, and $\phi^w(\boldsymbol{\Upsilon}_k|\boldsymbol{\Psi})$ as $\phi^w_{\boldsymbol{\Upsilon}_k|\boldsymbol{\Psi}}$.

We choose to fix the parameters to $\nu = 6$ and $\beta = 1$ and learn the parameters $\boldsymbol{\Psi}$, $\mathbf{W}_{gt}$ (when groundtruth is available), and $\mathbf{Q}_c$. For both sets of learned parameters, we then perform trajectory estimation on our test set, where we only use the lidar localization measurements with our learned covariance model and our learned motion prior.

Figure 3.2 shows the error plots where we have trained without groundtruth for our estimated x, y, and z positions, along with their 3σ covariance envelopes. As can be seen, the errors consistently remain within the covariance envelopes. We do, however, note that our estimator appears to be underconfident. We believe that this is a result of our decision to constrain $|\boldsymbol{\Psi}| = \beta = 1$ in order for our training method to work in practice. This decision is analogous to fixing the volume of the covariance ellipsoid to be constant. In doing so, we relied on the learned covariance of the motion prior to adjust

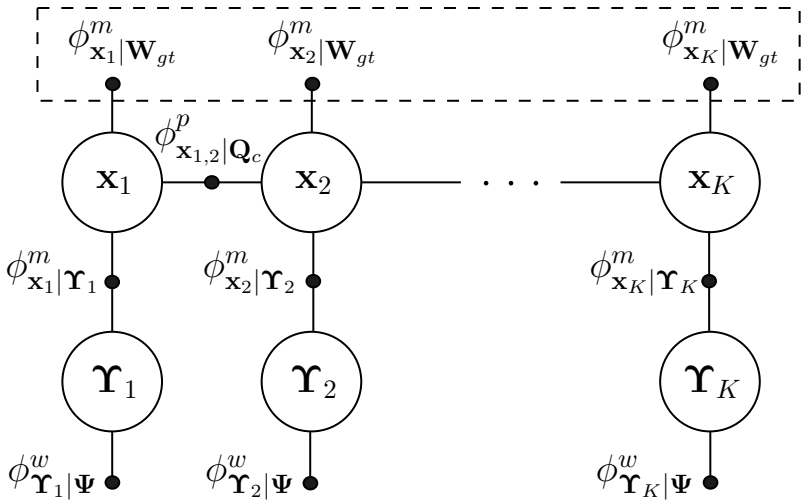


Figure 3.1: Factor graph for our vehicle estimation problem in Experiment A. White circles represent random variables to be estimated (vehicle state $\mathbf{x}$ and measurement covariances $\boldsymbol{\Upsilon}$). Small black dots represent factors in the joint likelihood of the data and the state. Binary motion prior factors, $\phi^p_{\mathbf{x}_{k-1,k}|\mathbf{Q}_c}$, depend on parameter $\mathbf{Q}_c$. Unary groundtruth pose factors (if available), $\phi^m_{\mathbf{x}_k|\mathbf{W}_{gt}}$, depend on parameter $\mathbf{W}_{gt}$. Factors $\phi^m_{\mathbf{x}_k|\boldsymbol{\Upsilon}_k}$ and $\phi^w_{\boldsymbol{\Upsilon}_k|\boldsymbol{\Psi}}$ are for applying an Inverse-Wishart prior over our measurement pose covariances, $\boldsymbol{\Upsilon}$, and depend on parameter $\boldsymbol{\Psi}$. We are able to learn parameters $\mathbf{Q}_c$ and $\boldsymbol{\Psi}$, even without groundtruth factors (factors inside dashed box).

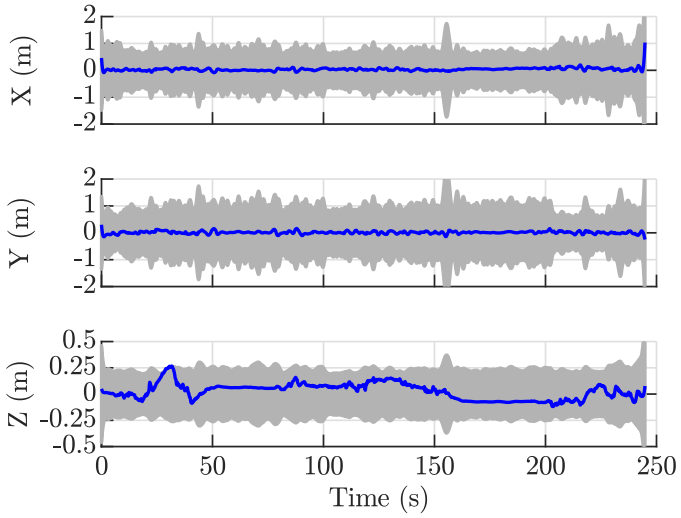


Figure 3.2: Experiment A - Error plots (blue lines) along with the 3σ covariance envelopes (gray background) when parameters are trained without groundtruth.

relative to the measurement covariances. The posterior mean is unaffected by this choice but not the posterior covariance.

Table 3.1 shows the resulting mean translational errors from both training methods on

Table 3.1: Experiment A - Comparison of translational errors on test set between training with complete groundtruth, with incomplete groundtruth, and without groundtruth (GT). We note that the first column, our previous work, did not learn the measurement covariances.

Seq no.	Trained with complete GT [56] (m)	Trained with incomplete GT (m)	Trained without GT (m)
0	0.0690	0.0720	0.0717
1	0.0888	0.1003	0.0925
2	0.4071	0.4148	0.4106
3	0.1947	0.1908	0.1847
4	0.2868	0.2866	0.2820
5	0.5703	0.5592	0.5549
6	0.3292	0.3014	0.2965
7	0.2207	0.2248	0.2230
8	0.1115	0.1151	0.1199
9	0.0979	0.1026	0.0997
overall	0.2376	0.2368	0.2335

all test sequences. We also include the results from our previous work where we trained using complete groundtruth for comparison.

While we achieve very similar errors across all training methods, the benefit is that we now do not require any groundtruth. Neither of the three training methods seem to outperform the others. We believe this is because our lidar localization measurements are quite accurate relative to groundtruth [56].

To further validate our method and show that we can indeed train with noisy measurements, we decided to artificially add additional noise to the measurements, where the noise statistics are unknown to the training process. We use the following $SE(3)$ perturbation scheme [5, 7] to inject noise into the position portion of our pose measurements:

$$\mathbf{T}_{\text{noisy}} = \exp(\boldsymbol{\xi}^{\wedge})\mathbf{T}_{\text{meas}}, \tag{3.37}$$

where

$$\boldsymbol{\xi} = \begin{bmatrix} \boldsymbol{\xi}_{1:3} \\ \mathbf{0} \end{bmatrix}, \quad \boldsymbol{\xi}_{1:3} \sim \mathcal{N}\left(\mathbf{0}, \sigma^2 \mathbf{I}\right). \tag{3.38}$$

We vary σ from 0.25 m to 1 m, injecting the same amount of noise into the test

measurements and training measurements.

Table 3.2 shows how our test errors change with increasing noise on measurements in both our training and test set. While measurement error increases significantly, up to over 1.6 m, we are still able to achieve translational errors of below 0.5 m on our estimated trajectory. This shows that we are still able to learn reasonable parameters of our system even without any groundtruth and quite noisy measurements.

Table 3.2: Experiment A - Analysis of how increasing noise on measurements affects the parameter learning method. Even with measurement errors of over 1.6 m, the errors on the estimated trajectory are under 0.5 m.

Measurement errors (m)	Estimated trajectory errors (m)
0.2407	0.2335
0.5010	0.2909
0.8653	0.3289
1.2481	0.3936
1.6383	0.4566

3.4.2 Experiment B: Training and Testing With Measurement Outliers

In Experiment B, we show that estimating time-varying covariances for each of our measurements with an IW prior results in outlier rejection. We artificially introduce outliers in our training and test set using the following method. With 5% probability, we apply the following perturbation to the actual pose measurement:

$$\mathbf{T}_{\text{outlier}} = \exp(\boldsymbol{\xi}^{\wedge})\mathbf{T}_{\text{meas}}, \tag{3.39}$$

with

$$\boldsymbol{\xi} \in \mathbb{R}^6 \sim \mathcal{U}(-200, 200). \tag{3.40}$$

Figure 3.3 shows an example of the measurement outliers on sequence 3 of our test set. We now seek to compare the performance between the cases where we have treated the measurement covariance, $\mathbf{W}$, as a static parameter to be learned, and where we have

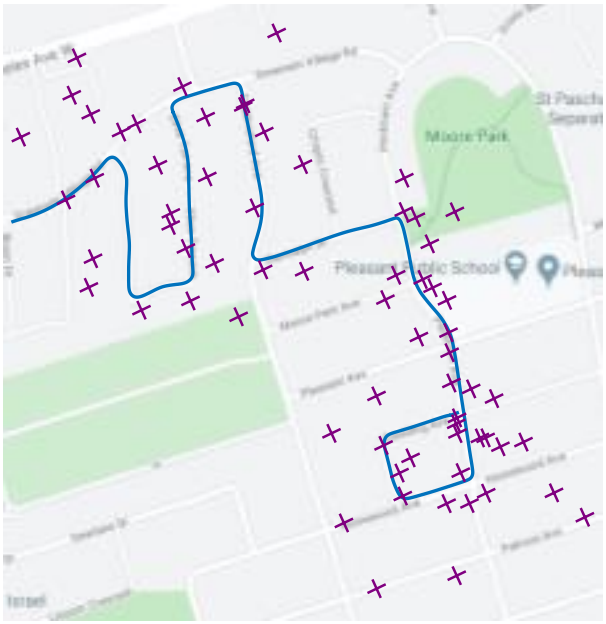


Figure 3.3: Experiments B & C - Measurement outliers (purple) overlaid with the groundtruth trajectory (blue) on sequence 3 of the test set. Background image from Google Maps.

treated the measurement covariance at each time as a random variable and learn the parameter, $\mathbf{\Psi}$, of the IW prior.

The loss functional corresponding to the static measurement covariance is

$$V(q'|\mathbf{W}, \mathbf{Q}_c) = \mathbb{E}_{q'}[\phi^p(\mathbf{x}|\mathbf{Q}_c) + \phi^m(\mathbf{x}|\mathbf{W})] + \frac{1}{2}\ln\left(|\mathbf{\Sigma}^{-1}|\right), \tag{3.41}$$

whereas for the IW prior on the measurement covariances, the loss functional is

$$V(q'|\mathbf{\Upsilon}, \mathbf{\Psi}, \mathbf{Q}_c) = \mathbb{E}_{q'}[\phi^p(\mathbf{x}|\mathbf{Q}_c) + \phi^m(\mathbf{x}|\mathbf{\Upsilon}) + \phi^w(\mathbf{\Upsilon}|\mathbf{\Psi})] + \frac{1}{2}\ln\left(|\mathbf{\Sigma}^{-1}|\right). \tag{3.42}$$

Table 3.3 shows the resulting translational errors on our test trajectory. We can see that without the IW prior, the estimation framework fails to reject outliers, resulting in an overall translation error of above 5 m. However, using the IW prior, we see that the error is only 0.2365 m. When we did not have any outliers at all in our data, the error was 0.2335 m (Table 3.1), meaning the average translational error on our test set only increased by 0.003 m.

Table 3.3: Experiment B - Translational errors using a static measurement covariance compared to using an IW prior when we have outliers in both our training and test set.

Seq no.	Static $\mathbf{W}$ (m)	IW Prior (m)
0	6.1976	0.0773
1	5.8371	0.0979
2	5.3652	0.4125
3	5.1217	0.1860
4	5.5186	0.2807
5	5.4780	0.5563
6	6.3936	0.3004
7	5.6898	0.2274
8	6.3717	0.1233
9	6.8032	0.1036
overall	5.8776	0.2365

From this experiment, we can see that using the IW prior allows for the handling of outliers in both training and testing due to our ability to estimate measurement covariances.

3.4.3 Experiment C: Training Without and Testing With Measurement Outliers

In Experiment B, we included outlier measurements in both the training and test set and saw that the IW prior allows us to achieve comparable errors to the case with no outliers. To see if this still holds even when we do not see any outliers in training, in Experiment C we train without any outliers but test with outliers. As the only difference between Experiment B and Experiment C is that we now train without any outliers, the loss functionals remain the same.

Table 3.4 shows that the resulting translational errors are again very high when we simply learn a static measurement covariance, but that we can still achieve reasonably low errors when learning the parameters of our IW prior. By incorporating the IW prior instead of learning a static measurement covariance, we decrease error from above 6 m to 0.2355 m. Compared to the error of 0.2335 m when there are no outliers in our test set (Table 3.1), we see an increase in error of only 0.002 m with the IW prior. This experiment shows that we can indeed still benefit from the outlier rejection scheme that

Table 3.4: Experiment C - Translational errors using a static measurement covariance compared to using an IW prior when we have outliers in our test set but not in our training set.

Seq no.	Static $\mathbf{W}$ (m)	IW Prior (m)
0	7.3504	0.0731
1	6.0754	0.0948
2	5.5771	0.4096
3	5.8157	0.1873
4	5.5503	0.2826
5	6.3057	0.5554
6	7.1858	0.2995
7	6.0332	0.2256
8	9.3079	0.1224
9	8.1237	0.1046
overall	6.7325	0.2355

comes with using an IW prior even when there are no outliers in our training set.

Comparing the results from Experiments B and C, we see that in both cases, incorporating the IW prior helps to reject outliers in the test set, regardless of whether there were any outliers in the training set. While errors for the static measurement covariance were similarly poor in both experiments, we note that the concentration of the errors are different as shown in Figure 3.4. What we see is that when we train without any outliers (Experiment C), the errors are concentrated where the outliers are in the test set. This result is unsurprising given that no outliers were seen in training, which in turn is reflected in our learned noise model parameters. When we train with outliers (Experiment B), the errors still peak around the outliers, but are more spread out over the entire trajectory. Regardless of this difference, we can see that using the IW prior is robust to both cases and still results in low translational errors.

3.4.4 Bicocca Dataset

We also evaluate on the Bicocca 25b real world dataset from [33] (github.com/ylatif/rrr.git) which provides a set of odometry constraints and loop closure constraints by running a front-end SLAM algorithm using the bag of words place recognition system from [13] on data collected during the Rawseeds project [14]. By varying the minimum confidence

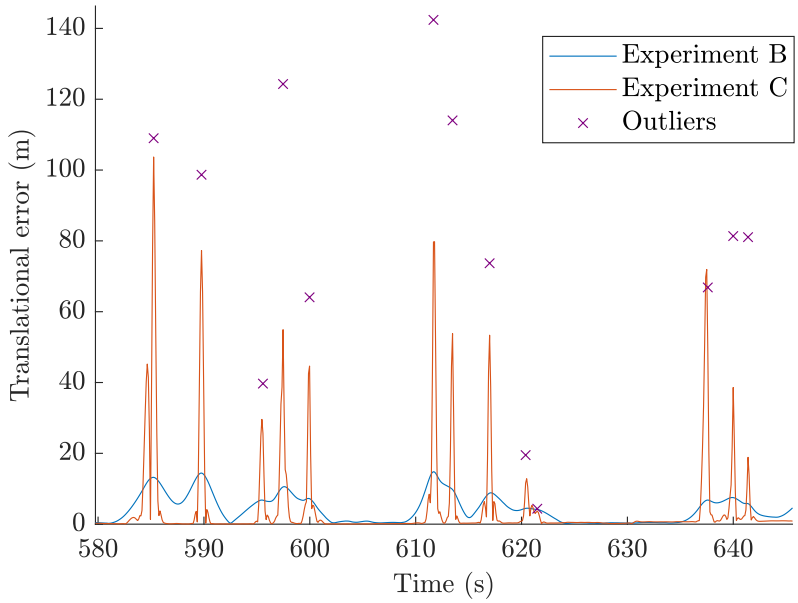


Figure 3.4: Experiments B & C - Translational errors for the static covariance method on a portion of the test set containing measurement outliers when training with and without outliers (Experiments B and C, respectively).

parameter (α^-) of the place recognition algorithm from 0 to 1 in increments of 0.025, [33] created 41 different datasets where the number of loop closures ranged from 23 to 446. Each dataset is represented as a pose graph that must be optimized. For our work, we will use two of these datasets with $\alpha^- = 0.45$ (no clear false loop closures) and $\alpha^- = 0.15$ (many clear false loop closures).

In the first dataset, we use $\alpha^- = 0.45$ which results in 137 loop closures as shown in Figure 3.5. The trajectory plotted from only odometry is shown in blue while the loop closures are shown in red.

We optimize the pose graph using 3 methods:

1. Our proposed framework using the IW prior on loop closure covariances and learning the parameter $\mathbf{\Psi}$ of our prior

2. Our proposed framework learning a static covariance $\mathbf{W}$ for all loop closure constraints

3. No covariance learning

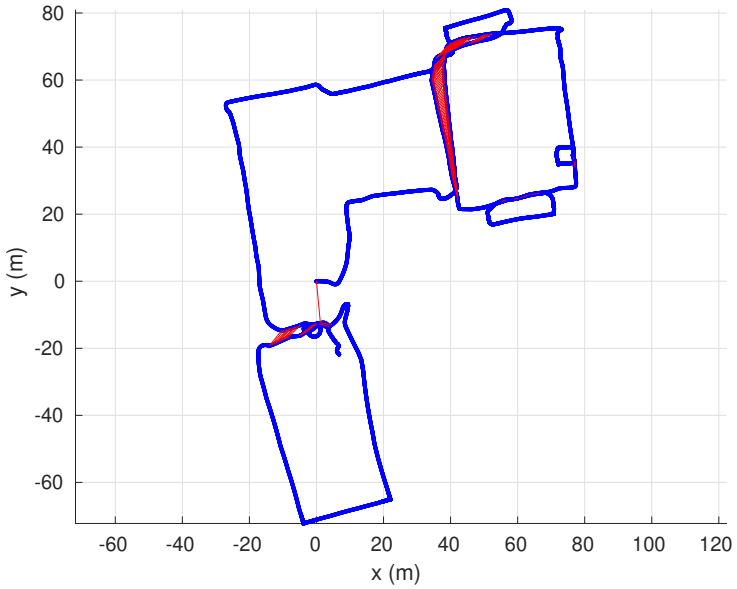


Figure 3.5: Odometric trajectory (blue) and loop closures (red) for a minimum confidence parameter of 0.45.

Table 3.5: Average Trajectory Error (ATE) as calculated by the Rawseeds Toolkit.

	IW prior (m)	Static $\mathbf{W}$ (m)	No covariance learning (m)
$\alpha^- = 0.45$	0.9094	0.9871	0.9968
$\alpha^- = 0.15$	2.3292	11.9059	29.4418

We show the results using these methods in Figure 3.6. Table 3.5 shows the Average Trajectory Error (ATE) as calculated by the Rawseeds Toolkit. We see that all three methods result in very similar estimates for $\alpha^- = 0.45$. This is because visually inspecting the loop closures from Figure 3.5, we can see that none of them are clear outliers.

We then test on a dataset with $\alpha^- = 0.15$, which results in 350 loop closures shown in Figure 3.7. It is known that this value of the minimum confidence parameter results in many false loop closures and is the same value used in [32] to test robustness of estimation algorithms to false loop closures. We can see from Table 3.5 that the false loop closures have a negative effect on the optimization without any covariance learning. Learning a static covariance helps improve the estimates a bit more, but with the IW prior, we are able to obtain a much better trajectory estimate as seen in Figure 3.8.

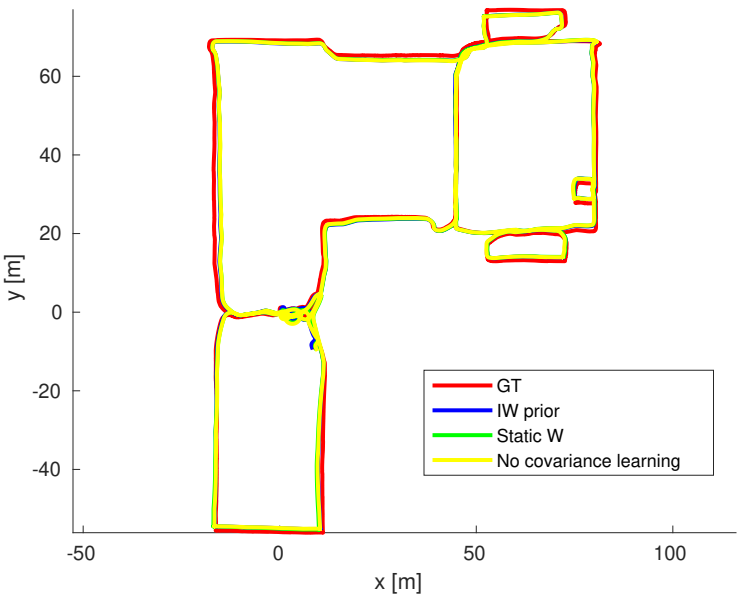


Figure 3.6: Resulting trajectory estimates for dataset with minimum confidence parameter of 0.45.

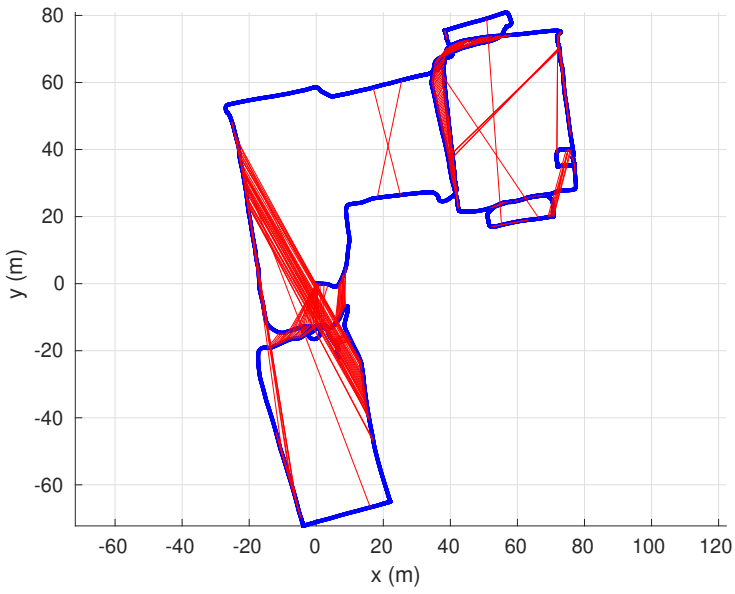


Figure 3.7: Odometric trajectory (blue) and loop closures (red) for a minimum confidence parameter of 0.15.

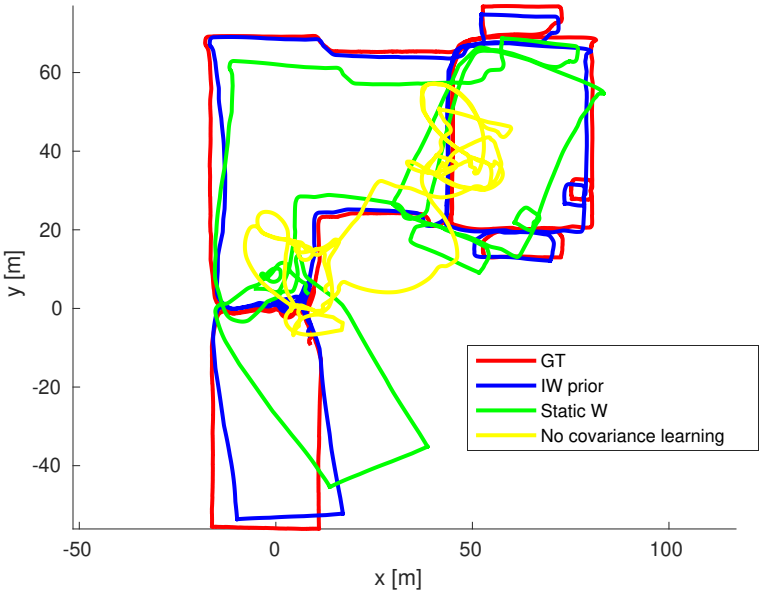


Figure 3.8: Resulting trajectory estimates for dataset with minimum confidence parameter of 0.15.

3.5 Summary

In this chapter, we presented parameter learning for ESGVI. Our parameter learning method does not need groundtruth, and is robust to noisy measurements and outliers. This is desirable because in many cases, we do not have a way of obtaining accurate groundtruth of robot trajectories. The implication of our work is that we now have a framework for estimating robot parameters based solely on whatever sensors are available.

We first experimentally demonstrated our method on a 36 km vehicle dataset where we show parameter learning for a trajectory estimation problem. We obtain comparable trajectory estimates even when learning parameters with increasingly noisy measurements and without observations of the full state. The ability to perform parameter learning without observations of the full state was a key limitation from Chapter 2 that was addressed in this chapter. We are also able to handle artificially introduced outlier measurements by placing placing an Inverse-Wishart prior on measurement covariances and estimating them alongside the latent trajectory. Lastly, we demonstrated our method on a pose graph optimization problem with real outliers from false loop closures.

Chapter 4

Conclusions and Future Work

4.1 Summary of Contributions

In the first part of our work, we showed how continuous-time trajectory estimation can be improved through the motion prior used. Instead of arbitrarily choosing a motion prior and its parameters, we developed a novel motion prior that can represent more types of trajectories. With these richer motion priors comes more parameters that must be tuned. We therefore also introduced a hyperparameter training method that uses data to select the hyperparameters most suitable for the particular robot. We show on a lidar localization and lidar odometry experiment, that by simply choosing a more appropriate prior and parameters, we can improve estimation.

Our parameter learning method required observations of the full state. This is often impractical for real world problems. Thus in the second part of our work, we show parameter learning within the Exactly Sparse Gaussian Variational Inference (ESGVI) framework. This enables us to use Expectation Maximization to learn model parameters with incomplete measurements of the state by alternating between estimation of the latent trajectory and finding the best parameters. We also show that outlier rejection can be achieved by estimating time-varying measurement covariances with an IW prior, whose parameters we also learn.

4.2 Future Work

A main portion of our work aimed at estimating proper parameters, including covariances. However, in the lidar odometry experiment we performed in Chapter 2, we did not have a principled way to choose the measurement covariances for our point cloud alignment. In the future, we can explore better methods to identify these measurement covariances to improve our estimator. Examples of work that estimate the covariance associated with point cloud alignment include [10] and [31]. This will be an important part of improving state estimation when point clouds are involved.

In modelling the trajectories, regardless of the continuous-time motion prior, we have represented each of the six pose variables as its own GP where there are no correlations between pose variables. However, this may not be the case as there can be correlations between them. As such, we could try to learn the parameters $\mathbf{Q}_c$ and $\boldsymbol{\alpha}$ without constraining them to be diagonal matrices. The new latent force model prior we derived represented latent accelerations as a Matérn covariance function with $v = 1/2$, which we showed was equivalent to the Singer acceleration model. Another extension would be to explore the use of other covariance functions and then perform hyperparameter training to estimate their parameters. Using methods from Chapter 3, this can be easily done without having to manually work out the covariance of error terms as was done in Chapter 2.

We could also consider incorporating control input data when training our motion prior to be able to better capture the dynamics of the robotic system.

In estimating time-varying covariances with the IW prior in Chapter 3, we still assumed two parameters to be known: ν, the DOF parameter for the IW distribution, and β, the determinant constraint on the scale matrix, $\boldsymbol{\Psi}$. For future work, we will investigate how to also learn ν and eliminate the need to constrain the determinant of $\boldsymbol{\Psi}$ to be a constant, β.

In this thesis, we chose to mostly learn the noise model parameters as a useful practical application of our framework. However, ESGVI can be used to learn entire robot models that are represented by rich modelling techniques, such as DNNs.

Bibliography

[1] Pieter Abbeel, Adam Coates, Michael Montemerlo, Andrew Y Ng, and Sebastian Thrun. Discriminative training of kalman filters. In *RSS*, volume 2, page 1, 2005.

[2] Mauricio Alvarez, David Luengo, and Neil D Lawrence. Latent Force Models. In *Artificial Intelligence and Statistics*, pages 9–16, 2009.

[3] Sean Anderson and Timothy D Barfoot. Full STEAM Ahead: Exactly Sparse Gaussian Process Regression for Batch Continuous-Time Trajectory Estimation on SE(3). In *IROS, 2015*, pages 157–164. IEEE, 2015.

[4] Sean Anderson, Timothy D Barfoot, Chi Hay Tong, and Simo Särkkä. Batch nonlinear continuous-time trajectory estimation as exactly sparse Gaussian process regression. *Auton. Robots*, 39(3):221–238, 2015.

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

[6] Timothy D Barfoot, James R Forbes, and David J Yoon. Exactly sparse Gaussian variational inference with application to derivative-free batch nonlinear state estimation. *The International Journal of Robotics Research*, 2020. doi:10.1177/0278364920937608.

[7] Timothy D Barfoot and Paul T Furgale. Associating Uncertainty With Three-Dimensional Poses for Use in Estimation Problems. *IEEE Transactions on Robotics*, 30(3):679–693, 2014.

[8] Timothy D Barfoot, Chi Hay Tong, and Simo Särkkä. Batch Continuous-Time Trajectory Estimation as Exactly Sparse Gaussian Process Regression. In *RSS*. Citeseer, 2014.

[9] C M Bishop. *Pattern Recog. and Machine Learning*. Springer, 2006.

[10] Silvère Bonnabel, Martin Barczyk, and François Goulette. On the Covariance of ICP-based Scan-matching Techniques. In *American Control Conference*, pages 5498–5503. IEEE, 2016.

[11] Michael Bosse and Robert Zlot. Continuous 3D scan-matching with a spinning 2D laser. In *ICRA, 2009*, pages 4312–4319. IEEE, 2009.

[12] Martin Brossard, Axel Barrau, and Silvère Bonnabel. Ai-imu dead-reckoning. *IEEE Transactions on Intelligent Vehicles*, 2020.

[13] César Cadena, Dorian Gálvez-López, Juan D. Tardós, and José Neira. Robust place recognition with stereo sequences. In *IEEE Transactions on Robotics*, 2012.

[14] Simone Ceriani, Giulio Fontana, Alessandro Giusti, Daniele Marzorati, Matteo Matteucci, Davide Migliore, Davide Rizzi, Domenico G. Sorrenti, and Pierluigi Taddei. Rawseeds ground truth collection systems for indoor self-localization and mapping. *Autonomous Robots*, 2009.

[15] Nived Chebrolu, Thomas Läbe, Olga Vysotska, Jens Behley, and Cyrill Stachniss. Adaptive robust kernels for non-linear least squares problems. *arXiv preprint arXiv:2004.14938*, 2020.

[16] Jing Dong, John Gary Burnham, Byron Boots, Glen Rains, and Frank Dellaert. 4D crop monitoring: Spatio-Temporal Reconstruction for Agriculture. In *ICRA, 2017*, pages 3878–3885. IEEE, 2017.

[17] Matej Gašperin and Dani Juričić. Application of unscented transformation in non-linear system identification. *IFAC Proceedings Volumes*, 44(1):4428–4433, 2011.

[18] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *CVPR, 2012*, pages 3354–3361. IEEE, 2012.

[19] Zoubin Ghahramani and Sam T Roweis. Learning nonlinear dynamical systems using an em algorithm. In *NIPS*, pages 431–437, 1999.

[20] Jouni Hartikainen and Simo Särkkä. Kalman Filtering and Smoothing Solutions to Temporal Gaussian Process Regression Models. In *IEEE International Workshop on Machine Learning for Signal Processing*, pages 379–384. IEEE, 2010.

[21] Jouni Hartikainen and Simo Särkkä. Sequential Inference for Latent Force Models. In *Conference on Uncertainty in Artificial Intelligence*, 2012.

[22] Jouni Hartikainen, Mari Seppanen, and Simo Särkkä. State-space Inference for Non-Linear Latent Force Models with Application to Satellite Orbit Prediction. In *ICML*, 2012.

[23] Congwei Hu, Wu Chen, Yongqi Chen, Dajie Liu, et al. Adaptive kalman filtering for vehicle navigation. *Journal of Global Positioning Systems*, 2(1):42–47, 2003.

[24] Jong Tai Jang, Sung Tae Moon, Sanghyuck Han, Hyeon Cheol Gong, Gi-Hyuk Choi, In Hee Hwang, and Joon Lyou. Trajectory Generation with Piecewise Constant Acceleration and Tracking Control of a Quadcopter. In *International Conference on Industrial Technology*, pages 530–535. IEEE, 2015.

[25] Johan Ludwig William Valdemar Jensen. Sur les fonctions convexes et les inégalités entre les valeurs moyennes. *Acta mathematica*, 30:175–193, 1906.

[26] Satish R Jondhale and Rajkumar S Deshpande. Tracking Target with Constant Acceleration Motion Using Kalman Filtering. In *International Conference On Advances in Communication and Computing Technology (ICACCT)*, pages 581–587. IEEE, 2018.

[27] Jonathan Ko and Dieter Fox. Gp-bayesfilters: Bayesian filtering using gaussian process prediction and observation models. *Autonomous Robots*, 27(1):75–90, 2009.

[28] Jonathan Ko and Dieter Fox. Learning gp-bayesfilters via gaussian process latent variable models. *Autonomous Robots*, 30(1):3–23, 2011.

[29] Juho Kokkala, Arno Solin, and Simo Särkkä. Expectation maximization based parameter estimation by sigma-point and particle smoothing. In *Intl. Conf. on Information Fusion*, pages 1–8. IEEE, 2014.

[30] Juho Kokkala, Arno Solin, and Simo Särkkä. Sigma-point filtering and smoothing based parameter estimation in nonlinear dynamic systems. *Journal of Advances in Information Fusion*, 11(1):15–30, 2016.

[31] David Landry, François Pomerleau, and Philippe Giguère. CELLO-3D: Estimating the Covariance of ICP in the Real World. In *ICRA, 2019*, pages 8190–8196. IEEE, 2019.

[32] Yasir Latif, Cesar Cadena, and Jose Neira. Detecting the correct graph structure in pose graph SLAM. *cs.adelaide.edu.au*, 2013.

[33] Yasir Latif, César Cadena, and José Neira. Robust loop closing over time for pose graph SLAM. *The International Journal of Robotics Research*, 32(14):1611–1626, dec 2013.

[34] X Rong Li and Vesselin P Jilkov. Survey of Maneuvering Target Tracking. Part 1: Dynamic Models. *IEEE Transactions on Aerospace and Electronic Systems*, 39(4):1333–1364, 2003.

[35] Katherine Liu, Kyel Ok, William Vega-Brown, and Nicholas Roy. Deep inference for covariance estimation: Learning gaussian noise models for state estimation. In *ICRA, 2018*, pages 1436–1443. IEEE, 2018.

[36] Takashi Matsuzaki, Hiroshi Kameda, Shingo Tsujimichi, and K Kosuge. Maneuvering Target Tracking Using Constant Velocity and Constant Angular Velocity

Model. In *International Conference on Systems, Man and Cybernetics*, volume 5, pages 3230–3234. IEEE, 2000.

[37] AH Mohamed and KP Schwarz. Adaptive kalman filtering for ins/gps. *Journal of geodesy*, 73(4):193–203, 1999.

[38] Mustafa Mukadam, Jing Dong, Frank Dellaert, and Byron Boots. Simultaneous Trajectory Estimation and Planning via Probabilistic Inference. In *RSS*, 2017.

[39] Mustafa Mukadam, Jing Dong, Xinyan Yan, Frank Dellaert, and Byron Boots. Continuous-time Gaussian process motion planning via probabilistic inference. *IJRR*, 37(11):1319–1340, 2018.

[40] Valentin Peretroukhin, William Vega-Brown, Nicholas Roy, and Jonathan Kelly. Probe-gk: Predictive robust estimation using generalized kernels. In *RA-L*, pages 817–824. IEEE, 2016.

[41] Rebecca L Russell and Christopher Reale. Multivariate uncertainty in deep learning. *arXiv preprint arXiv:1910.14215*, 2019.

[42] Simo Särkkä, Mauricio A Alvarez, and Neil D Lawrence. Gaussian Process Latent Force Models for Learning and Stochastic Control of Physical Systems. *IEEE Transactions on Automatic Control*, 2018.

[43] Thomas B Schön, Adrian Wills, and Brett Ninness. System ident. of nonlinear state-space models. *Automatica*, 47(1):39–49, 2011.

[44] R H Shumway and D S Stoffer. An approach to time series smoothing and forecasting using the EM algorithm. *Journal of Time Series Analysis*, 3(4):253–264, 1982.

[45] Robert A Singer. Estimating Optimal Tracking Filter Performance for Manned Maneuvering Targets. *IEEE Transactions on Aerospace and Electronic Systems*, (4):473–483, 1970.

[46] K Takahashi, J Fagan, and M-S Chen. A sparse bus impedance matrix and its application to short circuit study. In *Proc. of the PICA Conference*, 1973.

[47] Tim Tang, David Yoon, François Pomerleau, and Timothy D Barfoot. Learning a Bias Correction for Lidar-only Motion Estimation. In *CRV, 2018*, pages 166–173. IEEE, 2018.

[48] Tim Yuqing Tang, David Juny Yoon, and Timothy D Barfoot. A White-Noise-on-Jerk Motion Prior for Continuous-Time Trajectory Estimation on SE(3). *RA-L*, 4(2):594–601, 2019.

[49] Sebastian Thrun. *Probabilistic robotics*, volume 45. ACM New York, NY, USA, 2002.

[50] Chi Hay Tong, Paul Furgale, and Timothy D Barfoot. Gaussian Process Gauss-Newton for non-parametric simultaneous localization and mapping. *IJRR*, 32(5):507–525, 2013.

[51] Chi Hay Tong, Paul T Furgale, and Timothy D Barfoot. Gaussian Process Gauss-Newton: Non-Parametric State Estimation. In *CRV, 2012*, pages 206–213. IEEE, 2012.

[52] Jack M Wang, David J Fleet, and Aaron Hertzmann. Gaussian Process Dynamical Models for Human Motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):283–298, 2007.

[53] Michael Warren, Melissa Greeff, Bhavit Patel, Jack Collier, Angela P Schoellig, and Timothy D Barfoot. There's No Place Like Home: Visual Teach and Repeat for Emergency Return of Multirotor UAVs During GPS Failure. *RA-L*, 4(1):161–168, 2018.

[54] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian Processes for Machine Learning*, volume 2. Cambridge, MA: MIT Press, 2006.

[55] Andrew Gordon Wilson and Zoubin Ghahramani. Generalised wishart processes. In *Proc. of Conf. on Uncertainty in Artificial Intelligence*, UAI'11, page 736–744, Arlington, Virginia, USA, 2011. AUAI Press.

[56] Jeremy N Wong, David J Yoon, Angela P Schoellig, and Timothy D Barfoot. A Data-Driven Motion Prior for Continuous-Time Trajectory Estimation on SE (3). *IEEE Robotics and Automation Letters*, 5(2):1429–1436, 2020.

[57] Jeremy N Wong, David J Yoon, Angela P Schoellig, and Timothy D Barfoot. Variational Inference with Parameter Learning Applied to Vehicle Trajectory Estimation. *IEEE Robotics and Automation Letters*, 5(4):5291–5298, 2020.

[58] Xinyan Yan, Vadim Indelman, and Byron Boots. Incremental sparse GP regression for continuous-time trajectory estimation and mapping. *RAS*, 87:120–132, 2017.

[59] Yuanxi Yang and Weiguang Gao. An optimal adaptive kalman filter. *Journal of Geodesy*, 80(4):177–183, 2006.

[60] Ji Zhang and Sanjiv Singh. LOAM: Lidar Odometry and Mapping in Real-time. In *RSS*, volume 2, page 9, 2014.