UNCERTAINTY QUANTIFICATION IN THE CONTEXT OF 6D POSE ESTIMATION

A Dissertation
Presented to
The Academic Faculty

By

Maya Boumerdassi

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in Computer Science
College of Computing

Georgia Institute of Technology

May 2022

UNCERTAINTY QUANTIFICATION IN THE CONTEXT OF 6D POSE ESTIMATION

Thesis committee:

Dr. Cédric Pradalier
College of Computing
Georgia Institute of Technology

Dr. Gerandy Brito
College of Computing
Georgia Institute of Technology

Dr. Diyi Yang
College of Computing
Georgia Institute of Technology

Date approved: April 22, 2022

To my family,

ACKNOWLEDGMENTS

First and foremost, I have to thank my research supervisors, Dr. Cédric Pradalier and Benjamin Joffe. Without their assistance and dedicated involvement in every step throughout the process, this thesis would have never been accomplished. I would like to thank you very much for your support and understanding over this past year.

I would like to thank Benjamin Joffe for allowing me to work at his side on this deep learning project at the Georgia Tech Research Institute, providing me the opportunity to work in a stimulating environment which helped me to develop my understanding of the field and for offering me continuous support and valuable insights.

I would also like to thank Dr. Cédric Pradalier for accepting to be my thesis advisor, for informing the direction of my research and for always providing valuable inputs that informed my research.

I would like to thank my friend and collaborator Richard Huang for his valuable help, contribution, guidance and support for my thesis.

I would also like to show gratitude to my committee, including Dr. Diyi Yang and Dr. Gerandy Brito.

I would like to give special thanks to my family and friends for their continuous support and understanding when undertaking my research and writing my thesis.

CONTENTS

LIST OF FIGURES

LIST OF ACRONYMS

CAD Computer Aided Design

NNs Neural Networks

OOD Out-Of-Distribution

UQ Uncertainty Quantification

SUMMARY

We use the work on Deep Evidential Regression [1] that was initially developed in the context of simple regression in $\mathbb{R}$, and extend it to work on higher dimensional groups and manifolds with a Lie algebra structure. We develop a general framework for the Deep Evidential Loss and assess how well the models perform on several manifolds, identify and discuss several limitations encountered through experiments. Finally, one of the major goals of this thesis is to assess whether we could be use the Deep Evidential method in the context of 6D Object Pose Estimation, where it can be crucial to have both information on the prediction and the uncertainty on the prediction (really important for safety-critical robotic manipulation).

CHAPTER 1
INTRODUCTION AND BACKGROUND

Deep Neural Networks (NNs) are now a widely used in many research fields, going from speech and text recognition tasks [2] to complex tasks in safety-critical settings such as medical image analysis [3]. In particular, a lot of research has been done in the field of robotics, where Deep Neural Networks can benefit from the access to -sometimes live- sensor data. Indeed, there has been a lot of work done in the field of autonomous driving as discussed in [4]. Such applications require the access to models detecting and predicting accurate pose of objects with respect to a car. Moreover, as we are in a safety-critical context, it is crucial that the model not only predicts accurate poses, but it should also give a reliable uncertainty measure over its predictions, resist to adversarial attacks to prevent malicious hijacking and detect In- and Out-Of-Distribution (OOD) samples [5] (as the model is less likely to perform well on OOD data). If the model can successfully predict a reliable uncertainty measure, we could then build a system where, for instance, a human takes over the machine when the uncertainty goes over a given threshold [6].

1.1 Background

In this work, we are interested in Uncertainty Quantification (UQ) in the field of 6D Object Pose Estimation. Let us first introduce the field of 6D Object Pose Estimation in Deep Learning.

Deep Pose Estimation

Deep Neural Networks are now widely used to predict 6D Object Pose in $SE(3)$. Recent work has enabled the prediction of 6D Object Pose from RGB-D images [7, 8], a prominent task in a lot of robotic applications such as robotic grasping or autonomous driving.

Recent work have focused on predicting 6D Pose from RGB-D input data -while it was previously done from RGB data- to build more robust method and benefit from the addition of the depth dimension: such models are more robust to lighting, scaling, shifting, noise, obstruction in the input data since they have access to the depth information, major information when dealing with problems in 3D. To accurately predict 6D Object Poses, some models [8] use predefined Computer Aided Design (CAD) models of the object classes at hand.

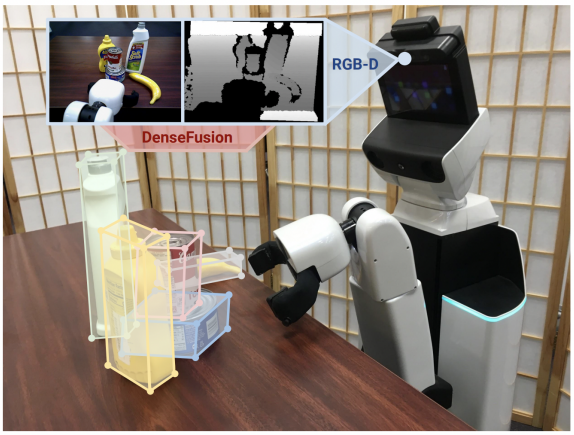


Figure 1.1: Dense Fusion Model: deep network model for 6D pose estimation from RGB-D data, performing predictions for real-time applications such as robot grasping and manipulation [8].

Other papers focused on improving the models to work on different instances belonging to the same class (for example, different instances, ie .models, of cameras in the class 'camera') without prior CAD models [7]. For a given object category, [7] use canonically oriented instances and normalize them to lie within the Normalized Object Coordinate Space (NOCS).

Most work focused on improving the accuracy of the models, leading to new state-of-the-art models for 6D Object Pose Estimation. But some robotic manipulation are safety-critical (e.g. manipulation with hazardous products, autonomous driving) such that we cannot simply rely on accuracy metric and require reliable uncertainty estimation to make decisions.

Uncertainty Quantification

Uncertainty Quantification has previously been studied in the context of 6D Pose Estimation [9]. Most of the methods used in the 3D Orthogonal Group $SO(3)$ and the 3D Special Euclidean Group $SE(3)$ use the Bingham Loss [10]. Other methods focused on estimating an uncertainty "by committee" [11]. We will further discuss these methods in chapter 2, but we believe that the Deep Evidential Regression method developed in [1] is promising as it combines aleatoric (uncertainty from the data) and epistemic (uncertainty from the model prediction) uncertainties.

1.2 Motivation

Uncertainty Quantification has also previously been studied in the context of 6D Pose Estimation [9]. Most of the methods used in the 3D Orthogonal Group $SO(3)$ and the 3D Special Euclidean Group $SE(3)$ use the Bingham Loss [10]. Other methods focused on estimating an uncertainty "by committee" [11]. We will further discuss these methods in chapter 2, but we believe that the Deep Evidential Regression method developed in [1] is promising as it combines aleatoric (uncertainty from the data) and epistemic (uncertainty from the model's prediction) uncertainties. This method [1] that it was only used in $\mathbb{R}$. Our work focuses on applying the method developed in [1] and extend it to higher dimensional groups and spaces with a Lie algebra structure (groups more commonly used in the context of robotic manipulation and state estimation). More precisely, we show that we can use our adapted version of the Deep Evidential Regression loss in $SE(3)$.

CHAPTER 2

RELATED WORK

In this section, we review previous research work on uncertainty quantification methods and their application to orientation and pose estimation.

Uncertainty Quantification is well researched topic that can be addressed using two approaches. The first one is to consider that UQ can be addressed in the form of a probabilistic distribution of possible outcomes on a prediction [10]. The second one is that it can be considered as a deterministic measure [11].

Overview of Uncertainty Quantification in Deep Learning

Let us first give an overview of uncertainty quantification in Deep Learning. Uncertainty Quantification is a well known topic in the field of classification and regression in $\mathbb{R}$ [12, 9, 13]: most of them introduce a new method based on density modelisation. Indeed, [14] shows that using Softmax as the last layer of the model is not a good choice (the model can be "fooled" if presented with noisy images, see Figure 2.1)

(a) 0.7 × Image **Softmax**: 0.8791 (b) Original Image **Softmax**: 0.9249 (c) 1.3 × Image **Softmax**: 0.9687

Figure 2.1: Softmax Score on a modified image from the ImageNet dataset using the VGG-16 classifier [14].

In [14], they propose their own new method for estimating predictive confidence based on density modeling. Their idea is to compare the distance of the current point to the points belonging to either classes to compute the confidence.

[15, 16] showed that "ReLU type" networks are not robust confidence estimators as they produce almost always high confidence predictions far away from the training data. They introduce a new robust optimization technique similar to adversarial training which enforces low confidence predictions far away from the training data. They show that this technique is surprisingly effective in reducing the confidence of predictions far away from the training data while maintaining high confidence predictions and test error on the original classification task compared to standard training. Most of these methods [14, 15, 16] cannot be used in our setting since we are not handling classification tasks.

Deterministic Uncertainty Estimation

Let us now consider the case where UQ is computed using a deterministic modelisation. Another important research area of Uncertainty Quantification (UQ) is ensemble methods [17, 18, 19], which is the combination of multiple models to get a more performant one. [11] focus on a deterministic approximation of the quality of a certain pose prediction.

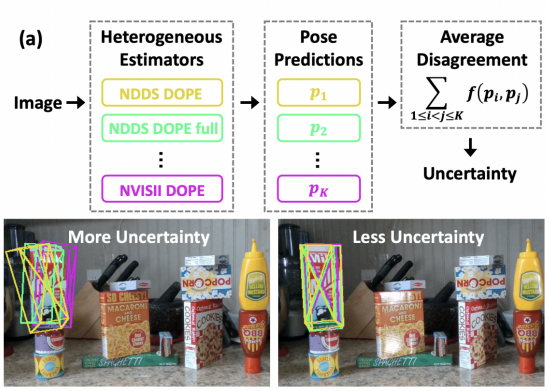


Figure 2.2: Illustration and examples of using an ensemble of heterogeneous models for uncertainty quantification. They calculate the average disagreement of K pose predictions from K different estimators as an estimation of uncertainty [11]

They introduce the Average Disagreement (ADD) as an uncertainty estimation. The idea is to train two or more heterogeneous models and computes their average disagreement against one another and consider this quantity as the uncertainty estimation - see Figure 2.2. Their study shows that this ADD metric is a relevant metric that successfully works in real-robot grasping application. This method benefits from the fact that it is a learning-free metric (ie. not internally learnt by the model). This method could be a good approach but it requires to train several Deep Neural Networks.

The first diadvantage is that, in the context of Pose Estimation NNs, the training is computationally heavy and can be hard to tune. Since most of the popular ensemble techniques require multiple samples during the inference, they are equivalent to independently trained models [20]. Besides, the number of parameters, along with the computational cost, linearly scales with the size of the ensemble, which is not acceptable in many critical time/memory-sensitive applications. Moreover, there is no clear recommendation as to how to select the different models to train. Besides, this method does not produce an uncertainty measure precisely identifying when the uncertainty comes from the data (for instance, when dealing with OOD samples) or when it comes from the model (epistemic uncertainty). That is why we decided to revert to another method.

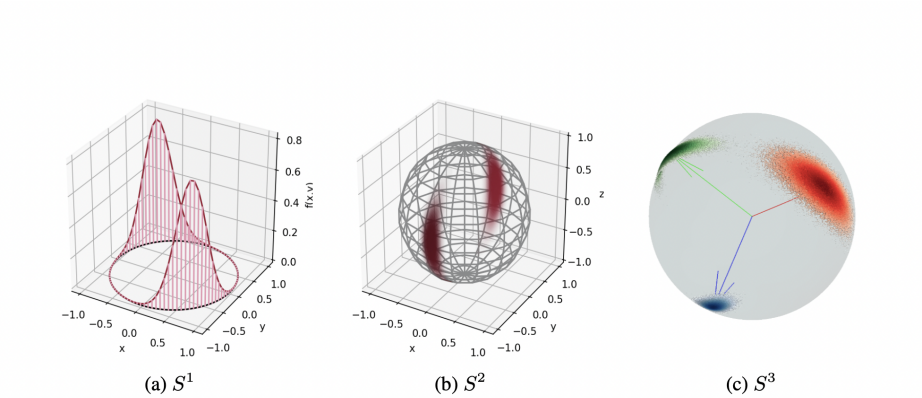
Probabilistic Uncertainty Estimation

The other type of approach to uncertainty quantification is to use probabilistic distribution of possible outcomes distribution [9, 21]. It is also referred to as Bayesian Deep Learning [22]. This allows a probabilistic interpretation of uncertainty by inferring some distributions over the model weights. Some techniques in Bayesian Deep Learning use methods such as dropout approximation [23] or Markov chain Monte Carlo [24].

When the models are used to predict orientations in $SO(3)$ or 6D poses in $SE(3)$, one of the most commonly used prior-distribution is the Bingham Distribution, which is a prob-

ability distribution with antipodal symmetry defined on the n-sphere. The characteristics of this distribution allow to formulate losses that naturally captures the antipodal symmetry of the representation [10, 25, 26].

In [10], they introduced the Bingham loss, a loss function based on the Bingham distribution that enables neural networks to predict uncertainty over unit quaternions and thus predict uncertain orientations. This work enables the use of highly symmetric objects and ambiguous sensor data in the context of pose and orientation estimation with uncertainty predictions -see Figure 2.3.



(a) S^1 (b) S^2 (c) S^3

Figure 2.3: Densities of the Bingham distribution represented for different dimensionality. For the circular case (a), for the spherical case (b), and for the 4d case (quaternions in $SO(3)$ (c). This representation allows to simultaneously represent the orientation and the corresponding uncertainty [10].

The major drawback with these methods is once again that they are computationally heavy, since they require multiple forward passes with some priors on the model parameters.

The Deep Evidential Regression method developped in [1] benefits from the fact that there is no need for multiple forward passes or sampling during inference, which speeds up the process. The other major advantage of this method is that it predicts both an epistemic (uncertainty of the model in the prediction) and aleatoric (uncertainty coming from the data) uncertainties. We will now dive deeper into this method and the framework we use in our work.

CHAPTER 3

METHODS

In this thesis, we focus on applying the Deep Evidential Regression loss [1] to different groups and manifolds with a Lie algebra structure. The main objective is to assess whether the Deep Evidential method can be applied to more complex manifolds and accurately predict both the regression target as well as a good uncertainty measure.

3.1 Original Deep Evidential Approach

In the base paper, [1] estimates uncertainty over a regression prediction in $\mathbb{R}$. Their main idea is to suppose that the targets $y_i \in \mathbb{R}$ are drawn i.i.d from a Gaussian Distribution with unknown mean and variance (μ, σ^2) and train a model to estimate the parameters of this unknown distribution. To do so, they place a Gaussian prior on the unknown mean μ and an Inverse Gamma prior on the unknown variance σ^2. Let $m \in \mathbb{N}$ be the number of samples. We have the following:

$$\begin{aligned} \forall i \in [\![1, m]\!], y_i \in \mathbb{R} \\ (y_1, ..., y_m) \sim \mathcal{N}(\mu, \sigma^2), \\ \mu \sim \mathcal{N}(\gamma, \sigma^2 \nu^{-1}),\ \sigma^2 \sim \Gamma^{-1}(\alpha, \beta) \end{aligned} \tag{3.1}$$

where $\Gamma(\cdot)$ is the gamma function and $\alpha > 1, \beta > 0, \nu > 0$.

To estimate the parameters of this unknown distribution, they develop a Deep Evidential Regression model which outputs 4 parameters $\omega = [\gamma, \alpha, \beta, \nu]$ for each target y_i, where $\gamma \in \mathbb{R}, \alpha > 1, \beta > 0, \nu > 0$. ω is then used to compute both the regression target and the uncertainty over the prediction.

On Figure 3.1 is represented the architecture of the neural network used in [1]. They introduce a custom final dense layer referred to as "Linear Normal Gamma Layer". Through this custom layer, they apply different activation functions for each parameter: they apply a $softplus(\cdot)$ activation on (α, β, ν) with an additional $+1$ to α since $\alpha > 1$; they apply a linear activation function for γ.

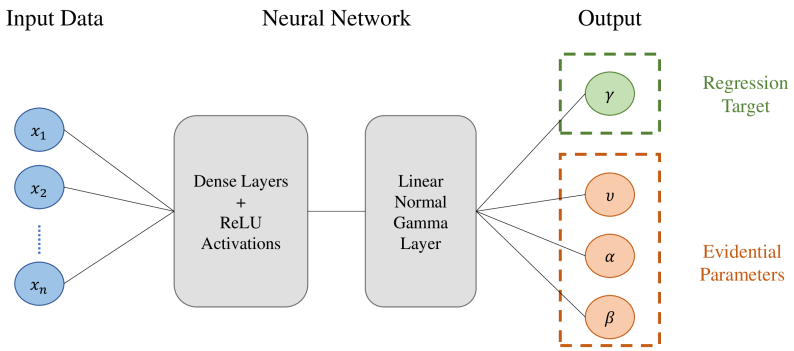


Figure 3.1: Deep Evidential Neural Network

To estimate the unknown distribution parameters μ and σ^2 and the uncertainty over the prediction, they use the following formulas:

$$\underbrace{\mathbb{E}(\mu) = \gamma}_{\text{prediction}}, \qquad \underbrace{\mathbb{E}(\sigma^2) = \frac{\beta}{\alpha - 1}}_{\text{aleatoric}}, \qquad \underbrace{\text{Var}(\mu) = \frac{\beta}{\nu(\alpha - 1)}}_{\text{epistemic}} \tag{3.2}$$

The three quantitiees respectively represent the prediction, the aleatoric uncertainty (uncertainty coming from the input data) and epistemic uncertainty (uncertainty of the model in the prediction).

To train the model, they introduce the Deep Evidential loss $\mathcal{L}_i$. Let us denote $y_i \in \mathbb{R}$ the i-th regression target and $\omega = [\gamma, \alpha, \beta, \nu]$ the model output where $\gamma \in$ R is the model's prediction and $\alpha > 1, \beta > 0, \nu > 0$ are the evidential parameters.

The loss $\mathcal{L}_i$ is defined by:

$$\mathcal{L}_i(y_i, \omega) = \mathcal{L}_i^{\text{NLL}}(y_i, \omega) + \lambda \mathcal{L}_i^{\text{R}}(y_i, \omega) \tag{3.3}$$

where $\lambda > 0$ the regularization coefficient.

$\mathcal{L}_i^{\text{NLL}}(y_i, \gamma)$ is the Negative Logarithm of model evidence and $\mathcal{L}_i^{\text{R}}(y_i, \gamma)$ is the evidence regularizer defined by::

$$\mathcal{L}_i^{\text{NLL}}(y_i, \omega) = \frac{1}{2}\log\frac{\pi}{\nu} - \alpha\log\Omega + \left(\alpha + \frac{1}{2}\right)\log\left(\left(y_i - \gamma\right)^2\nu + \Omega\right) + \log\left(\frac{\Gamma(\alpha)}{\Gamma\left(\alpha + \frac{1}{2}\right)}\right) \tag{3.4}$$

$$\mathcal{L}_i^{\text{R}}(y_i, \omega) = \left|y_i - \mathbb{E}\left[\mu_i\right]\right| . \Phi = \left|y_i - \gamma\right| (2\nu + \alpha) \tag{3.5}$$

where $\Omega = 2\beta(1 + \nu)$, and $\Gamma(\cdot)$ is the gamma function.

3.2 General Framework

We noticed that the work by Amini et. al. [1] on Deep Evidential Regression is adapted to regression tasks in $\mathbb{R}$ in the cartesian space. The thesis focuses on extending the Deep Evidential Loss to:

1. multi-dimension regression tasks in $\mathbb{R}^n$ in the Cartesian Space
2. regression tasks in groups and manifolds with a Lie Algebra structure.

Let us first define a general framework for Deep Evidential Regression. The dependence of $\mathcal{L}_i$ as defined by Amini et. al.[1] on y_i is through its distance to γ (see Equation 3.4 and Equation 3.5). Let us introduce the distance function $d(y_i, \gamma)$. Using this distance function, we can rewrite the formula of the loss $\mathcal{L}_i$ and get a general framework for computing

$\mathcal{L}_i$ as follows:

$$\mathcal{L}_i\left(d(y_i,\gamma),\omega\right) = \mathcal{L}_i^{\text{NLL}}\left(d(y_i,\gamma),\omega\right) + \lambda\mathcal{L}_i^{\text{R}}\left(d(y_i,\gamma),\omega\right) \tag{3.6}$$

with $d(y_i,\gamma) = |y_i - \gamma|$ the distance function in $\mathbb{R}$ and $\lambda > 0$ in the original case of a regression task in $\mathbb{R}$ in the cartesian space.

The geometry of the objects varies with the space they lie in, thus the distance function $d(y_i,\gamma)$ should also reflect the properties of the space. Since the distance $d(y_i,\gamma)$ is the only term in the Deep Evidential loss $\mathcal{L}_i$ that depends on y_i, we can hope to extend the Deep Evidential loss to other spaces by choosing a more appropriate distance function.

In this work, we decided to predict a 1D uncertainty parameterized by the evidential parameters α, ν and β (see Equation 3.2) and we also made the assumption that dimensions are independent. Even though this is a significant approximation, it leads to models that are easier to implement, tune and reproduce. Moreover in our experiments, we noticed that the loss was more stable than in a multivariate case described in the work by [27]. Even if we don't know the precise direction of the uncertainty when dealing with multi-dimension objects, we can still get a good approximation of the overall 1D uncertainty on the prediction, with its epistimic and aleatoric components. This thesis is a first step to extend Deep Evidential Regression to multiple spaces, and further work could be done using multivariate loss inspired from [27].

3.2.1 Multidimensional case

The most direct extension of the 1D regression is the multidimensional case. Let $n \in \mathbb{N}$ be the dimension of the cartesian space and $m \in \mathbb{N}$ be the number of samples.

Predicting an uncertainty measure by dimension

This section was done in collaboration with Richard Huang. In this case, $y_i \in \mathbb{R}^n$ and $\omega = [\gamma, \alpha, \beta, \nu] \in \mathbb{R}^{n \times 4}$. As a simplification with respect to the multivariate case [27], we make the assumption that the dimensions are independent.

This way, we can consider that for each dimension $j \in 1, ..., n$, the inputs $y_{i_j} \in \mathbb{R}$ follow Gaussian distributions with unknown mean and variance (μ_j, σ_j^2). In short, we have the following setting:

$$\begin{gathered} \forall i \in [\![1, m]\!], y_i \in \mathbb{R}^n \\ \forall j \in [\![1, n]\!], (y_{1_j}, ..., y_{m_j}) \sim \mathcal{N}(\mu_j, \sigma_j^2), \\ \mu_j \sim \mathcal{N}(\gamma_j, \sigma_j^2 \nu_j^{-1}),\ \sigma_j^2 \sim \Gamma^{-1}(\alpha_j, \beta_j) \end{gathered} \tag{3.7}$$

where $\Gamma(\cdot)$ is the gamma function and $\alpha_j > 1, \beta_j > 0, \nu_j > 0$.

Since the dimensions are independent, we can compute the losses $\mathcal{L}_{i_j}$ independently for each dimension $j \in 1, .., n$ using Equation 3.6 with the distance function $d_{\mathbb{R}}$ in $\mathbb{R}$ and sum the losses to compute the Deep Evidential loss $\mathcal{L}_i$:

$$\begin{gathered} \mathcal{L}_i\left(y_i, \omega\right) = \sum_{j=1}^{n} \mathcal{L}_{i_j}\left(d_{\mathbb{R}}(y_{i,j}, \gamma_j), \omega_j\right) \\ d_{\mathbb{R}}\left(y_{i_j}, \gamma_j\right) = \left|y_{i_j} - \gamma_j\right| \end{gathered} \tag{3.8}$$

where $y_{i,j}$ the j-th component of $y_i \in \mathbb{R}^n$ and $\omega_j = [\gamma_j, \alpha_j, \beta_j, \nu_j] \in \mathbb{R}^4$ the corresponding model output.

Predicting a single dimension of uncertainty

An alternative method is to predict a single dimension of uncertainty for the n dimensions. This is the method that we used in other manifolds. In this case, $y_i \in \mathbb{R}^n$ and $\omega = [\gamma, \alpha, \beta, \nu]$ with $\gamma \in \mathbb{R}^n$ and $(\alpha, \beta, \nu) \in \mathbb{R}^3$ the evidential parameters.

This way, we can consider that the inputs $y_i \in \mathbb{R}$ follow Gaussian distributions with unknown mean and variance (μ, σ^2). In short, we have the following setting:

$$\begin{aligned} \forall i \in [\![1, m]\!], y_i \in \mathbb{R}^n \\ (y_1, ..., y_m) \sim \mathcal{N}(\mu, \sigma^2), \\ \mu \sim \mathcal{N}(\gamma, \sigma^2 \nu^{-1}),\ \sigma^2 \sim \Gamma^{-1}(\alpha, \beta) \end{aligned} \tag{3.9}$$

where $\Gamma(\cdot)$ is the gamma function and $\alpha > 1, \beta > 0, \nu > 0$.

In this case, we compute a single the Deep Evidential loss $\mathcal{L}_i$ for each y_i using Equation 3.6 with the distance function d_{1D} in $\mathbb{R}$:

$$d_{1D}\left(y_i, \gamma\right) = \|y_i - \gamma\| \tag{3.10}$$

with the L_2 norm in $\mathbb{R}^n$, $y_i \in \mathbb{R}^n$ and $\omega = [\gamma, \alpha, \beta, \nu] \in \mathbb{R}^4$ the corresponding model output.

3.2.2 Predicting an angle

Let us consider that we want to predict an angle in $\mathbb{R}$. Depending on the representation used, this section can be applied to several groups that are closely related. Indeed, predicting an angle $y_i \in \mathbb{R}$ is equivalent to predicting points $y_i \in \mathbb{R}^2$ on a 1-sphere $S(1)$ such that $\|y_i\| = 1$. A 2D rotation matrix in $SO(2)$ is also parametrized by one rotation angle $\theta \in \mathbb{R}$. An angle $\theta \in \mathbb{R}$ is equivalent to $(\cos\theta, \sin\theta) \in S(1)$ and $[[\cos\theta, -\sin\theta], [\sin\theta, \cos\theta]] \in SO(2)$.

Let us assume $y_i \in \mathbb{R}$ is an angle in $\mathbb{R}$ and $\omega = [\gamma, \alpha, \beta, \nu] \in \mathbb{R}^4$. If y_i is an angle, and we don't consider too high variances with respect to π (since we are handling angles), we can use Gaussian distributions even though their use for angles remains an approximation. Let $m \in \mathbb{N}$ be the number of samples. In this case, we have the following setting:

$$\begin{gathered} \forall i \in [\![1, m]\!], y_i \in \mathbb{R} \text{ an angle} \\ \forall i \in [\![1, m]\!], (y_1, ..., y_m) \sim \mathcal{N}(\mu, \sigma), \\ \mu \sim \mathcal{N}(\gamma, \sigma^2 \nu^{-1}),\ \sigma^2 \sim \Gamma^{-1}(\alpha, \beta) \end{gathered} \tag{3.11}$$

where $\Gamma(\cdot)$ is the gamma function and $\alpha > 1, \beta > 0, \nu > 0$.

We can compute the corresponding Deep Evidential loss $\mathcal{L}_i$ with Equation 3.6 using two alternative distance functions d_{geo} for angles or $d_{S(1)}$ for points in $S(1)$, defined as:

$$\begin{aligned} d_{geo}(y_i, \gamma) &= |fmod\,(y_i - \gamma + \pi, 2\pi) - \pi| \\ d_{S(1)}(y_i, \gamma) &= \left| \frac{y_i \cdot \gamma}{\|y_i\|} - 1 \right| \end{aligned} \tag{3.12}$$

with the 2D dot product and the L_2 norm in $\mathbb{R}$.

3.2.3 2D Special Euclidean Group SE(2) case

Let us now consider $y_i \in SE(2)$ such that $y_i = [t_i, r_i] \in \mathbb{R}^3$ where $t_i \in \mathbb{R}^2$ and $r_i \in \mathbb{R}$ respectively represent 2 independent translations and 1 independent rotation angle.

Predicting multiple uncertainty measures

In this case, we explore the case where we predict one uncertainty measure for each translation and one uncertainty measure for the rotation part. The model outputs $\omega = [\omega_t, \omega_r]$ such that $\omega_t = [\gamma_t, \alpha_t, \beta_t, \nu_t] \in \mathbb{R}^{2\times4}$ and $\omega_r = [\gamma_r, \alpha_r, \beta_r, \nu_r] \in \mathbb{R}^{1\times4}$.

Let $m \in \mathbb{N}$ be the number of samples. We have the following setting:

$$\begin{gathered}
\forall i \in [\![1, m]\!], y_i = [t_i, r_i] \in SE(2) \\
\text{Translation part: } \forall j \in [\![1, 2]\!], (t_{1_j}, ..., tm_j) \sim \mathcal{N}(\mu_{t_j}, \sigma_{t_j}), \\
\text{with } \mu_{t_j} \sim \mathcal{N}(\gamma_{t_j}, \sigma_{t_j}^2 \nu_{t_j}^{-1}),\ \sigma_{t_j}^2 \sim \Gamma^{-1}(\alpha_{t_i}, \beta_{t_j}) \\
\text{Rotation part: } (r_1, ..., r_m) \sim \mathcal{N}(\mu_r, \sigma_r), \\
\text{with } \mu_r \sim \mathcal{N}(\gamma_r, \sigma_r^2 \nu_r^{-1}),\ \sigma_r^2 \sim \Gamma^{-1}(\alpha_r, \beta_r)
\end{gathered} \tag{3.13}$$

where $\Gamma(\cdot)$ is the gamma function and $\alpha_{t_j} > 1, \beta_{t_j} > 0, \nu_{t_j} > 0$ and $\alpha_r > 1, \beta_r > 0, \nu_r > 0$. Assuming the translations and rotation are independent, we compute the following loss $\mathcal{L}_i$:

$$\begin{aligned}
\mathcal{L}_i(y_i, \omega) &= \mathcal{L}_i(t_i, \omega_t) + \mathcal{L}_i(r_i, \omega_r) \\
&= \sum_{j=1}^{2} \mathcal{L}_{i_j}\left(d_{\mathbb{R}}(t_{i_j}, \gamma_{t_j}), \omega_{t_j}\right) + \mathcal{L}_i(d_{geo}(r_i, \gamma_r), \omega_r)
\end{aligned} \tag{3.14}$$

where the losses are computed using Equation 3.6.

Previous work [28] shows that we can set a weighting to regress translation and rotation (for both $SE(2)$ and $SE(3)$) for better results. We can either tune it as a hyperparameter or learn an optimal weighting inside the model. Either way, adding this weighting parameter will not be discussed here because it does not change the formalism of the loss $\mathcal{L}_i$.

Predicting a single dimension of uncertainty

An alternative method is to predict a single dimension of uncertainty for both the translation and rotation part. Let us recall that $y_i \in SE(2)$ such that $y_i = [t_i, r_i] \in \mathbb{R}^3$ where $t_i \in \mathbb{R}^2$ and $r_i \in \mathbb{R}$. In this case, the model predicts $\omega = [\gamma, \alpha, \beta, \nu]$ with $\gamma = [\gamma_t, \gamma_r] \in SE(2)$ and $(\alpha, \beta, \nu) \in \mathbb{R}^3$ the evidential parameters.

This way, we can consider that the inputs $y_i \in SE(2)$ follow Gaussian distributions with unknown mean and variance (μ, σ^2).

In short, we have the following setting:

$$\begin{aligned} \forall i \in [\![1, m]\!], y_i \in SE(2) \\ (y_1, ..., y_m) \sim \mathcal{N}(\mu, \sigma^2), \\ \mu \sim \mathcal{N}(\gamma, \sigma^2 \nu^{-1}),\ \sigma^2 \sim \Gamma^{-1}(\alpha, \beta) \end{aligned} \tag{3.15}$$

where $\Gamma(\cdot)$ is the gamma function and $\alpha > 1, \beta > 0, \nu > 0$.

In this case, we compute a single the Deep Evidential loss $\mathcal{L}_i$ for each y_i using Equation 3.6 with the distance function $d_{SE(2)}$ in $\mathbb{R}$:

$$d_{SE(2)}\left(y_i, \gamma\right) = \|t_i - \gamma_t\| + \lambda_d\, d_{S(1)}(r_i, \gamma_r) \tag{3.16}$$

with the L_2 norm in $\mathbb{R}^2$, $y_i \in SE(2)$, $\gamma = [\gamma_t, \gamma_r] \in SE(2)$ the corresponding model prediction and $\lambda_d > 0$.

3.2.4 Points on a sphere: the S(2) case

Let us consider $y_i \in S(2)$. We have $y_i \in \mathbb{R}^3$ such that $\|y_i\| = 1$.

Predicting a single dimension of uncertainty

For this manifold, we only consider predicting a 1D uncertainty over all dimensions, such that $\omega = [\gamma, \alpha, \beta, \nu]$ with $\gamma \in \mathbb{R}^3$ the mean vector in the unit-sphere predicted by the model and $\alpha, \beta, \nu \in \mathbb{R}$ the evidential parameters describing an error in terms of geodesic distance. As for the angle, the use of Gaussian distributions on the sphere is an approximation that will be acceptable as long as the uncertainties are small with respect to π. The complete adaptation of the evidential regression to generic distributions formally defined in manifolds is beyond the scope of this thesis.

Let $m \in \mathbb{N}$ be the number of samples. We have the following setting:

$$\begin{aligned} \forall i \in [\![1, m]\!], y_i \in S(2) \\ (y_1, ..., y_m) \sim \mathcal{N}(\mu, \sigma^2), \\ \mu \sim \mathcal{N}(\gamma, \sigma^2 \nu^{-1}),\ \sigma^2 \sim \Gamma^{-1}(\alpha, \beta) \end{aligned} \tag{3.17}$$

where $\Gamma(\cdot)$ is the gamma function and $\alpha > 1, \beta > 0, \nu > 0$.

We compute the loss $\mathcal{L}_i$ in $S(2)$ using Equation Equation 3.6 with the distance function $d_{S(2)}$ defined as follows with the 3D dot product and the L_2 norm in $\mathbb{R}^3$:

$$d_{S(2)}\left(y_i, \gamma\right) = \left| \frac{y_i \cdot \gamma}{\|y_i\|} - 1 \right|. \tag{3.18}$$

3.2.5 3D Orthogonal Group SO(3) case

Let us now focus on the 3D Orthogonal Group $SO(3)$ application. Rotations in $SO(3)$ can be represented using quaternions. Let us first review some information on quaternions. This part was done in collaboration with Richard Huang, who focused on the definition of the distance metric while I focused on the sampling of quaternions.

Overview of quaternions

Quaternions are a generalization of complex numbers with three imaginary parts (i, j and k). A quaternion is a complex number in dimension 4 that can be used to represent the orientation of a rigid body or the coordinates of a reference frame in a three-dimensional space. The general definition of a quaternion q is given by:

$$q = a + b.i + c.j + d.k = [a\ b\ c\ d] \tag{3.19}$$

where $a, b, c, d \in \mathbb{R}$.

Since we are interested in predicting rotations in $SO(3)$, we will focus on the manifold of unit quaternions q such that $\|q\| = 1$. Indeed, these quaternions are of particular interest to us for representing rotations as isometries. They are a continuous and smooth representation of rotations. They lie on the unit manifold, which is a simple constraint to enforce through back-propagation. This way, quaternions are an attractive choice for deep learning because they are easily formulated in a continuous and differentiable way.

There are several properties of quaternions that are important to remember as they are used in later sections in this thesis, but let us first focus on the Deep Evidential Framework definition for $SO(3)$.

Distance Metric in $SO(3)$

In the $SO(3)$ case, we will consider a quaternion as one object and assume that its dimensions are not independent (see Equation 3.25 for more details), which is the most important difference with the multidimensional case mentioned earlier in the thesis.

Let us consider y_i the 3D rotation target represented as a unit quaternion q_i such that $\|q_i\| = 1$ (by definition of quaternions in $SO(3)$ [29]). The model output is $\omega = [\gamma, \alpha, \beta, \nu]$ with $\gamma \in SO(3)$ the mean predicted quaternion and $(\alpha, \beta, \nu) \in \mathbb{R}^3$ the evidential parameters.

Let us first focus on defining a distance function for the quaternions in $SO(3)$. The distance between two quaternions is the norm of the rotation transforming one into the other. Let us introduce the distance function d_{quat} such that:

$$d_{quat}(y_i, \gamma) = \left| 1 - \left| \frac{y_i \cdot \gamma}{\|y_i\|_2} \right| \right| \tag{3.20}$$

with $\cdot$ being the group operator in $SO(3)$ and the L_2 norm in $\mathbb{R}^4$.

Let us denote S^3 the set of unit quaternions where $S^3 = \{q \in SO(3), | \ \|q\| = 1\}$. One edge case with unit quaternions $q \in S^3$, is that q and $-q$ represent the same orientation - even if the Euclidean distance between these two quaternions is 2. Thus, we would like the distance function to ensure that $d_{quat}(q, -q) = 0$ in that case.

Let us now prove that Equation 3.20 represents a distance function between two quaternions while ensuring $d_{quat}(q, -q) = 0$. A commonly used distance function for unit quaternions [30, 31] is the following distance d using $arccos$:

$$d(q_1, q_2) = arccos(2\,|\langle q_1, q_2 \rangle| - 1) \tag{3.21}$$

where $\langle\ ,\ \rangle$ is the inner product in S^3.

Let us first prove that d_{quat} is a distance function for unit quaternions. Let us consider $q_1, q_2 \in S^3$ with respective real parts $\Re(q_1) = \frac{y_i}{\|y_i\|_2} \in R^4$ and $\Re(q_2) = \gamma \in R^4$. Let $\theta \in [0, \pi]$ be the rotation angle between the orientations described by the unit quaternions $q_1 \in S^3$ and $q_2 \in S^3$, ie. such that $d(q_1, q_2) = \theta$. We have the following equations:

$$\theta = arccos(2\,|\langle q_1, q_2 \rangle| - 1) \Leftrightarrow \frac{1 - cos(\theta)}{2} = 1 - |\langle q_1, q_2 \rangle| = 1 - \left| \frac{y_i \cdot \gamma}{\|y_i\|_2} \right| \tag{3.22}$$

Equation 3.22 proves that d_{quat} is a distance function, as it is equivalent to the commonly used distance function d in S^3.

Regarding the edge case $q_1 = -q_2$, if $\Re(q_1) = \frac{y_i}{\|y_i\|_2} = -\gamma = \Re(q_2)$, then we have $d_{quat}(q_1, q_2) = 0$ from Equation 3.20.

However, we noticed that it is harder to use this distance function (Equation 3.20) in practice with neural networks. Indeed, when minimizing the loss, thus the distance function d_{quat}, it is difficult for the neural network to represent the whole 4D hyper-sphere while ensuring $d_{quat}\,(q, -q) = 0$. Thus, we chose to remove the absolute value on the dot product of $\frac{y_i}{\|y_i\|_2}$ and γ.

We also added restrictions on the generation of the quaternions in our dataset. The first one is that we restrict the quaternions to be only on the positive side of the hyper-sphere S^3, meaning that $\forall q \in S^3$ with $q = a + b.i + c.j + d.k$, we have $a \in \mathbb{R}_*^+$ and $(b, c, d) \in \mathbb{R}^3$ [32, 33]. When we sample q such that $a = 0$, we enforce that $b \in \mathbb{R}^+$. When q is sampled such that $a = b = 0$, we enforce $c \in \mathbb{R}^+$. Finally, when q is sampled such that $a = b = c = 0$, we enforce $d \in \mathbb{R}^+$.

Deep Evidential Loss and Model Architecture

Let us now resume the framing of the Deep Evidential Loss in $SO(3)$. As mentioned earlier, we consider that the dimensions of a quaternions are not independent, which is the most important difference with the multidimensional case. Let $m \in \mathbb{N}$ be the number of samples. We have the following setting:

$$\begin{aligned} \forall i \in [\![1, m]\!], y_i \in SO(3) \\ (y_1, ..., y_m) \sim \mathcal{N}(\mu, \sigma^2), \\ \mu \sim \mathcal{N}(\gamma, \sigma^2 \nu^{-1}),\ \sigma^2 \sim \Gamma^{-1}(\alpha, \beta) \end{aligned} \tag{3.23}$$

where $\Gamma(\cdot)$ is the gamma function and $\alpha > 1, \beta > 0, \nu > 0$.

We modified the structure of the last Linear Normal Gamma Layer (see Figure 3.2) so that it predicts only one set of evidential parameters α, ν, β in $\mathbb{R}$ for each target quaternion q_i.

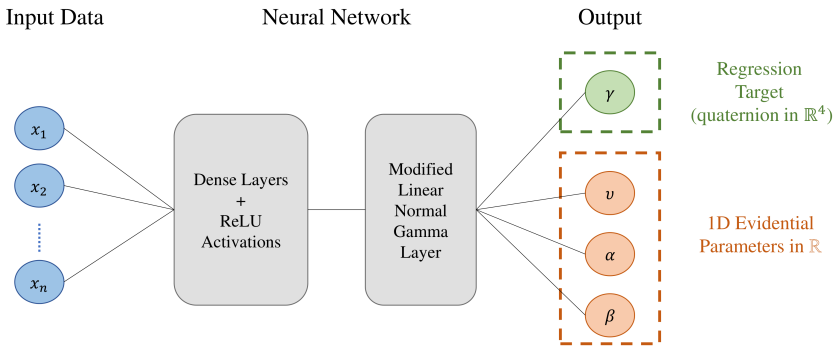


Figure 3.2: Modified Deep Evidential NN to predict 1D uncertainty in SO(3)

To compute the loss $\mathcal{L}_i$, we can use Equation 3.6 with a geodesic distance d_{quat} redefined as:

$$d_{quat}(y_i, \gamma) = \left| 1 - \frac{y_i \cdot \gamma}{\|y_i\|_2} \right| \tag{3.24}$$

with $\cdot$ being the group operator in $SO(3)$ and the L_2 norm in $\mathbb{R}^4$.

Let us now introduce other important properties of quaternions and discuss how we sampled quaternions in this work.

Quaternions and Euler angles

Another major property of quaternions is that they can be computed from Euler angles. Euler angles are angles describing the orientation of a solid or a reference frame with respect to a Cartesian trihedron of reference. Euler angles represent an orientation by a succession of 3 rotations.

It is common to use the aerospace notations for the Euler angles, referring to the three angles as the roll ϕ, pitch θ and yaw ψ angles (see Figure 3.3).

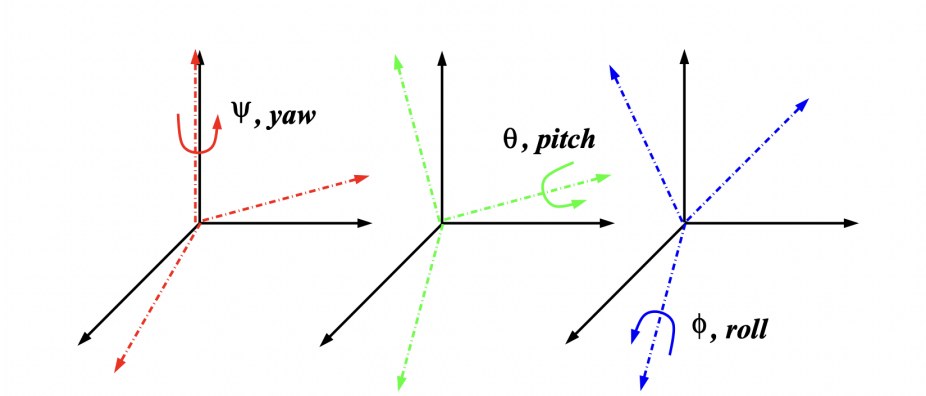


Figure 3.3: Euler angles using the roll, pitch, yaw representation [34]

More precisely, quaternions can be computed from Euler angles using the following:

$$q = \begin{pmatrix} cos(\phi/2)cos(\theta/2)cos(\psi/2) + sin(\phi/2)sin(\theta/2)sin(\psi/2) \\ sin(\phi/2)cos(\theta/2)cos(\psi/2) - cos(\phi/2)sin(\theta/2)sin(\psi/2) \\ cos(\phi/2)sin(\theta/2)cos(\psi/2) + sin(\phi/2)cos(\theta/2)sin(\psi/2) \\ cos(\phi/2)cos(\theta/2)sin(\psi/2) - sin(\phi/2)sin(\theta/2)cos(\psi/2) \end{pmatrix} \tag{3.25}$$

where ϕ, θ, ψ represent the roll, pitch and yaw angles. This method is used in section 4.6 to generate synthetic data.

Generating Noisy Quaternions

To add noise in the training data, we use the axis-angle representation in order to guarantee that the distance between the raw and noisy quaternions is equal to the applied noise. To do that, we iteratively sample an angle $\alpha \sim \mathbb{N}(\mu, \sigma^2)$ (where μ and σ are set by the user) and compute the corresponding quaternion q_{noise} using:

$$q_{noise} = \left[cos\left(\frac{\alpha}{2}\right),\ sin\left(\frac{\alpha}{2}\right) u \right] \tag{3.26}$$

where u is a vector randomly sampled in $[-1, 1]^3$ and $u \neq [0, 0, 0]$. The resulting noisy quaternion is obtained with $q_{new} = q_{raw} \cdot q_{noise}$ where $\cdot$ is the group operator in $SO(3)$.

This way, we can ensure that the distance between the noisy and raw quaternion is equal to the noise we applied, that is:

$$d_{quat}(q_{raw}, q_{new}) = \alpha. \tag{3.27}$$

This is the method that we use in section 4.6.

3.2.6 3D Special Euclidean Group SE(3) case

Let us consider $y_i \in SE(3)$.

Predicting multiple uncertainty measures

This subsection was done in collaboration with Richard Huang, as we worked together on the quaternion case. Let us denote $y_i = [t_i, q_i]$ where $t_i \in \mathbb{R}^3$ and $q_i \in SO(3)$ such that $\|q_i\| = 1$ respectively represent 3 independent translations and a 3D rotation. The model output is $\omega = [\omega_t, \omega_r]$ with $\omega_t = [\gamma_t, \alpha_t, \beta_t, \nu_t] \in \mathbb{R}^{3\times 4}$ and $\omega_r = [\gamma_r, \alpha_r, \beta_r, \nu_r]$ such that $\gamma_r \in \mathbb{R}^4$ is the predicted quaternion and $\alpha_r, \beta_r, \nu_r \in \mathbb{R}$ are the evidential parameters of the rotation part. Let $m \in \mathbb{N}$ be the number of samples. We have the following setting:

$$\begin{gathered}
\forall i \in [\![1, m]\!], y_i = [t_i, q_i] \in SE(3) \\
\text{Translation part: } \forall j \in [\![1, 3]\!], (t_{1_j}, ..., tm_j) \sim \mathcal{N}(\mu_{t_j}, \sigma_{t_j}), \\
\text{with } \mu_{t_j} \sim \mathcal{N}(\gamma_{t_j}, \sigma^2_{t_j}\nu^{-1}_{t_j}),\ \sigma^2_{t_j} \sim \Gamma^{-1}(\alpha_{t_i}, \beta_{t_j}) \\
\text{Rotation part: } (q_1, ..., q_m) \sim \mathcal{N}(\mu_q, \sigma_q), \\
\text{with } \mu_q \sim \mathcal{N}(\gamma_q, \sigma^2_q\nu^{-1}_q),\ \sigma^2_q \sim \Gamma^{-1}(\alpha_q, \beta_q)
\end{gathered} \tag{3.28}$$

where $\Gamma(\cdot)$ is the gamma function and $\alpha_{t_j} > 1, \beta_{t_j} > 0, \nu_{t_j} > 0$ and $\alpha_q > 1, \beta_q > 0, \nu_q > 0$. We assume for simplicity that both translations and rotations are independent, thus we can

compute loss $\mathcal{L}_i$ in $SE(3)$ using:

$$\begin{aligned}\mathcal{L}_i\left(y_i, \omega\right) &= \mathcal{L}_i\left(t_i, \omega_t\right) + \mathcal{L}_i\left(q_i, \omega_r\right) \\ &= \underbrace{\sum_{j=0}^{2} \mathcal{L}_i\left(d_{\mathbb{R}}(t_{i_j}, \gamma_j), \omega_j\right)}_{\text{translation}} + \underbrace{\mathcal{L}_i\left(d_{quat}(q_i, \gamma), \omega\right)}_{\text{rotation}}\end{aligned} \tag{3.29}$$

where the losses are computed using Equation Equation 3.6. As mentioned earlier in section subsection 3.2.3, we could add a weighting for translation and rotation to get better results as discussed in [28], but it does not change the formalism of our work.

Predicting a single dimension of uncertainty

An alternative method is to predict a single dimension of uncertainty for both the translation and rotation parts. Let us denote $y_i = [t_i, q_i]$ where $t_i \in \mathbb{R}^3$ and $q_i \in SO(3)$ such that $\|q_i\| = 1$ respectively represent 3 translations and a 3D rotation. The model output is $\omega = [\gamma, \alpha, \beta, \nu]$ with $\gamma = [\gamma_t, \gamma_r] \in SE(3)$ and $(\alpha, \beta, \nu) \in \mathbb{R}^3$ are the evidential parameters. Let $m \in \mathbb{N}$ be the number of samples. We have the following setting:

$$\begin{gathered}\forall i \in [\![1, m]\!], y_i = [t_i, q_i] \in SE(3) \\ (y_1, ..., y_m) \sim \mathcal{N}(\mu, \sigma), \\ \text{with } \mu \sim \mathcal{N}(\gamma, \sigma^2 \nu^{-1}),\ \sigma^2 \sim \Gamma^{-1}(\alpha, \beta)\end{gathered} \tag{3.30}$$

where $\Gamma(\cdot)$ is the gamma function and $\alpha > 1, \beta > 0, \nu > 0$.
In this case, we compute a single the Deep Evidential loss $\mathcal{L}_i$ for each y_i using Equation 3.6 with the distance function $d_{SE(3)}$ in $\mathbb{R}$:

$$d_{SE(3)}\left(y_i, \gamma\right) = \|t_i - \gamma_t\| + \lambda_d\, d_{quat}(r_i, \gamma_r) \tag{3.31}$$

with the L_2 norm in $\mathbb{R}^3$, $y_i \in SE(3)$, $\gamma = [\gamma_t, \gamma_r] \in SE(3)$ the corresponding model prediction and $\lambda_d > 0$.

CHAPTER 4

EXPERIMENTS

In this section, we apply the modified Deep Evidential $\mathcal{L}_i$ losses (defined in Methods) on their respective lie algebra groups and manifolds and assess whether they successfully estimate both the regression target and consistent uncertainties on synthetic toy examples.

4.1 Replicating the experiments of the base paper

Let us first replicate and investigate the experiments described in the base paper [1].

4.1.1 Same setting and hyperparameters as base paper

Let us first reproduce the Deep Evidential model and toy example in $\mathbb{R}$ described in [1]. We use the code from [1] available on Github as base. The code was originally developed in Tensorflow, so the first step was to successfully convert it in PyTorch. In [1], they introduced the following toy example $y = t^3 + \sigma$ where $\sigma \sim \mathbb{N}(0, 3)$, $t_{train} \sim \mathbb{U}([-4, 4])$ and $t_{test} \sim \mathbb{U}([-7, 7])$, and show that the method is working on this example. We reproduced the experiments and obtained the results shown in Figure 4.1 and Figure 4.2. The model contains 3 Dense layers with ReLU activations and a final Linear Normal Gamma Layer. The results were obtained with $lr = 5e{-}4$, $\lambda = 0.01$ (evidential regularizer) and $epochs = 500$.

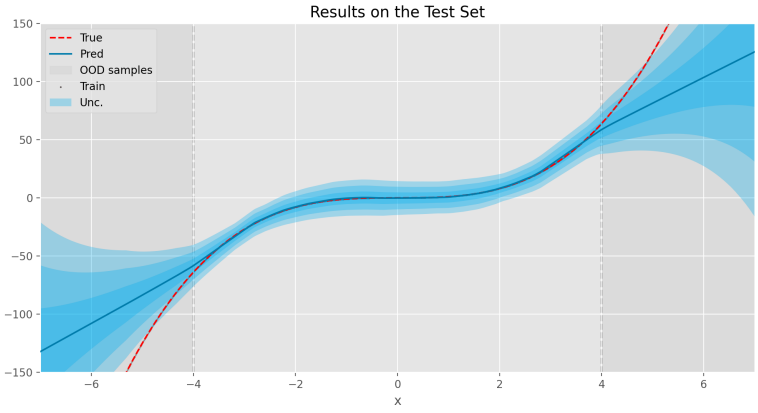


Figure 4.1: Reproducing the experiment on the toy example from the base paper [1]

The model seems to successfully regress the target and predict a consistent epistemic uncertainty. Indeed, we can see on Figure 4.1 that the epistemic uncertainty (uncertainty coming from the prediction), represented with the blue highlight, is increasing as we move closer to test data out-of train data distribution. Let us now look at the Deep Evidential Loss over iterations on Figure 4.2. The loss seems to successfully converge after 300 iterations. In fact, we realized that the method was not stable: depending on the training hyperparameters, the model may predict values that are not consistent (e.g. decreasing epistemic uncertainty for OOD samples). Moreover, when we reran several times the same training, the results differed from one iteration to another.

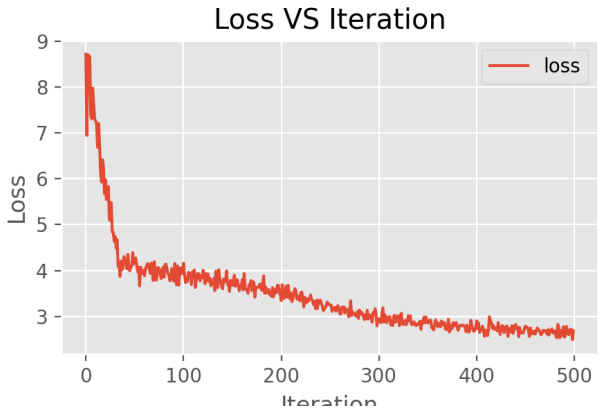


Figure 4.2: Loss obtain on the toy example from the base paper [1]

Let us now assess the aleatoric uncertainty (the uncertainty coming from the noise in the training input data). To do that, let us plot on Figure 4.3 the applied noise σ and both uncertainties. We can see that the aleatoric uncertainty seems to be higher and biased around when x is close to 0, while the noise is a constant.

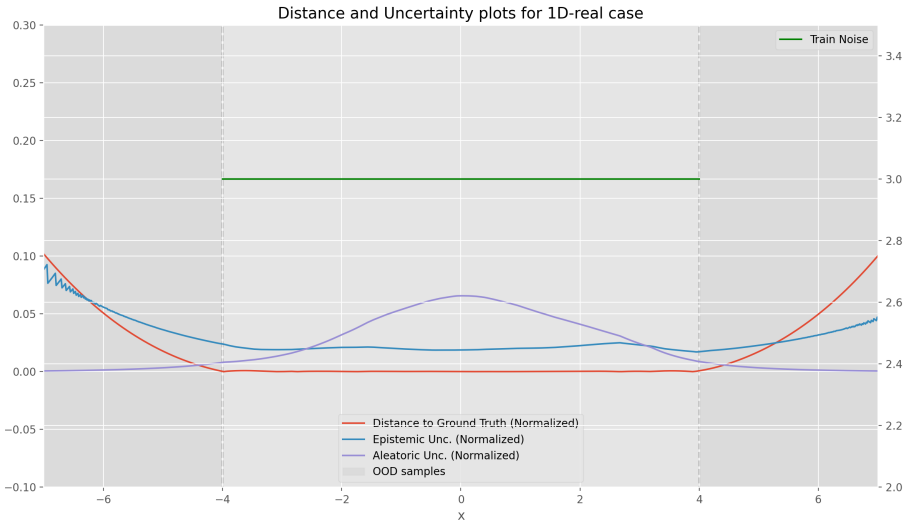


Figure 4.3: Plotting Aleatoric and Epistemic Uncertainties along with the train Noise (constant σ case)

4.1.2 Adding "holes" in the data

Let us now modify the training setting of the base experiment. The idea we try here is to add 'holes' in the data, ie. modifying the intervals from which we sample the training data so that there is at least two disconnected intervals. This time, we will uniformly sample t_{train} from $[-7, -2]\ U\ [2, 7]$ and t_{test} from $[-7, 7]$. The results on Figure 4.5 and Figure 4.4 are obtained with $lr = 5e{-}4$, $\lambda = 0.01$ (evidential regularizer) and $epochs = 1000$.

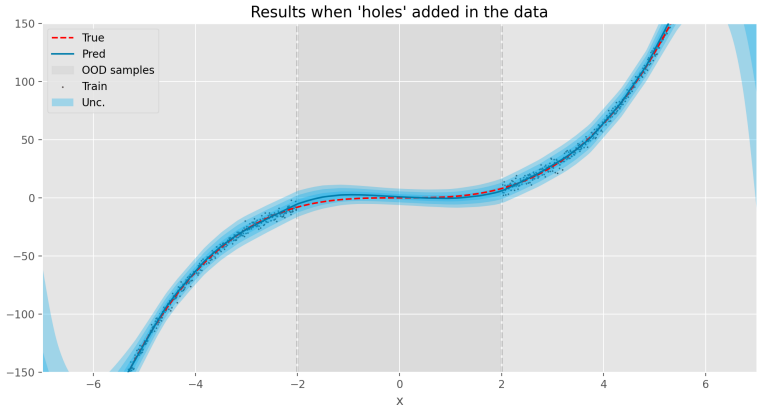


Figure 4.4: Experiment with holes in the train set centered aroung 0

We can see that while the model seems to give consistent regression predictions, both uncertainties do not behave as expected: the epistemic uncertainty is low on samples OOD, while it should be high for those samples. There seems to be a bias for the data centered around 0.

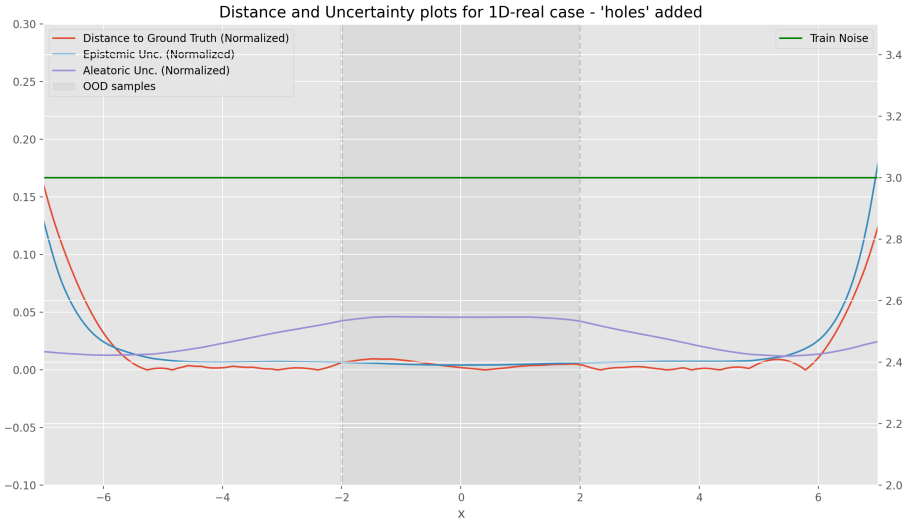


Figure 4.5: Plotting Aleatoric and Epistemic Uncertainties along with the train Noise with holes

4.1.3 Applying Gaussian Noise

Let us now see how the model reacts to varying noise: we will add more important noise at the center of the graph. To do that, we now consider $y(t) = t^3 + noise(t)$ where $noise(t) \sim \mathbb{N}(0, \sigma_n(t))$ and σ_n follows a bell-shaped curve.

To implement the bell-shaped curve noise, we used the following method. We consider that σ_n corresponds to the probability density function ϕ of a normal distribution centered on 0 with an offset of 3. We used this method as it was easy to implement using the method $norm.pdf(t, \mu, \sigma)$ from the package scipy.stats. In short, we have :

$$\sigma_n(t) = \phi_{\mu,\sigma}(t) + \text{offset} \tag{4.1}$$

where $\phi_{\mu,\sigma}$ is the probability density function of a normal distribution with $\mu = 0$ and $\sigma = 3$, and offset $= 3$. The corresponding results are on Figure 4.6. The epistemic uncertainty is increasing for samples OOD and the aleatoric seems consistently to follow the Gaussian distribution of the noise.

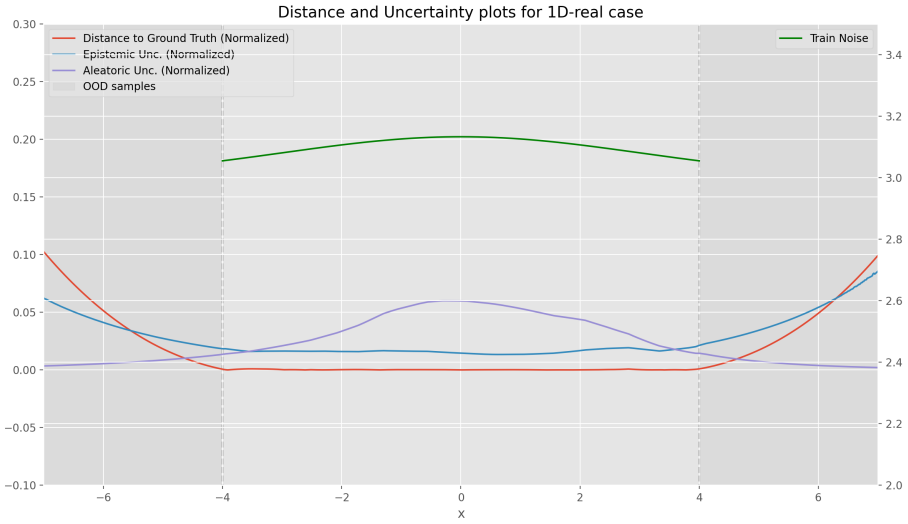


Figure 4.6: Plotting Aleatoric and Epistemic Uncertainties along with the train Noise ($\sigma \sim \mathbb{N}(0, 3) + 3$ case)

Let us now consider that σ_n follows the probability density function $\phi_{\mu,\sigma}$ centered on

$\mu = 1$ and $\sigma = 3$ with an offset of 3 (see Equation 4.1). When we offset the noise distribution, the aleatoric uncertainty predicted by the model does not follow the known noise distribution (see Figure Figure 4.7).

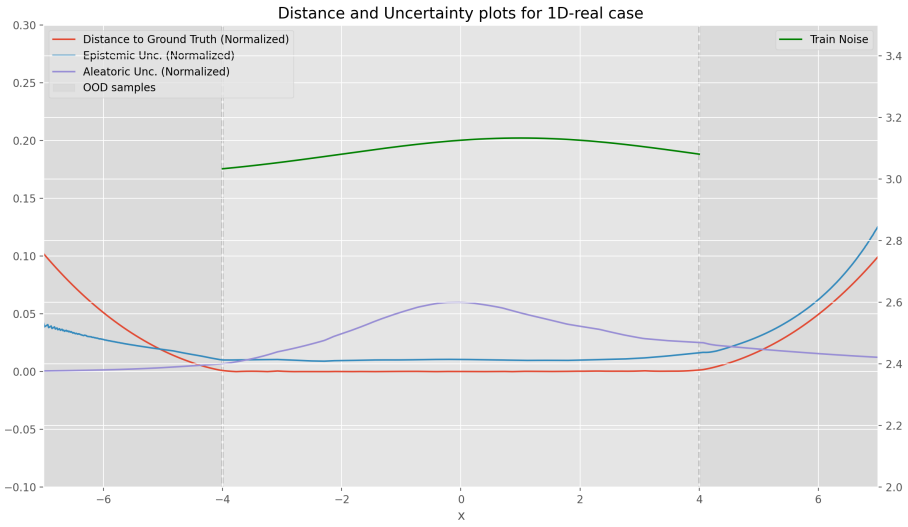


Figure 4.7: Plotting Aleatoric and Epistemic Uncertainties along with the train Noise ($\sigma \sim \mathbb{N}(1,3) + 3$ case)

4.1.4 Conclusion

We were able to reproduce the experiment in $\mathbb{R}$ from the base paper [1]. Indeed, we get similar results as those presented in their paper when we use the Deep Evidential method on the exact same toy example, ie. with $y = t^3 + \sigma$ where $\sigma \sim \mathbb{N}(0,3)$, $t_{train} \sim \mathbb{U}([-4,4])$ and $t_{test} \sim \mathbb{U}([-7,7])$. Further experiments on their method showed that the method is not robust to more complex noise and when data are not uniformly distributed. Indeed, even if the model seems to correctly regress the objective function, the uncertainty estimation is not consistent with the data sampling and noise applied. Besides, it seems that the method has a bias for target functions centered around 0.

There may be some underlying assumptions on how to set the toy examples that were not explicitly described by [1]. Nevertheless, this method that aims to predict both an aleatoric and an epistemic uncertainties seems promising and that is why we will assess how well it works on other manifolds.

4.2 Multidimensional case

Let us now apply the Deep Evidential loss $\mathcal{L}_i$ defined for each manifold on toy examples. For each manifold, we apply noise on the input data and evaluate the model on a test set containing in- and out-of-distribution samples. We also apply varying noise on the training data to assess how robust the method is to this parameter.

Regarding the definition of our toy problems, we mainly use odd target functions centered around 0 and apply Gaussian Noise, as it was done in previous work [1, 19, 35].

4.2.1 Predicting multiple uncertainty measures

Let us denote $k \in \mathbb{N}$ the dimension of a cartesian space and the targets as $y \in \mathbb{R}^k$. As mentioned in chapter 3, we make the assumption that the dimensions of y are independent in this manifold. For each independent dimension i, we consider data such that $y_i = f_i(t) + n_i$ where $f_i(\cdot)$ is a function in $\mathbb{R}$ centered around 0 and n_i represents Gaussian Noise. We later describe in this section precisely the functions f_i used and how we sampled the noise n_i.

Experimental Setup

Let us first define the architecture of the Neural Networks used for the multidimensional case. The model is sequentially constituted of:

1. one Linear Layer (input dim=1, out dim=128) + ReLU activation
2. two Linear Layers (input dim=128, out dim=128) + ReLU activation
3. one Linear Normal Gamma Layer (input dim=128, out dim=k) (see section 3.1 for details on this custom layer).

We use the Adam optimizer and the loss $\mathcal{L}_i$ defined by Equation 3.8 for the multidimensional case.

We work with $k = 4$ dimensions. The input data t is sampled such that $t_{train} \sim \mathbb{U}([-4, 4])$ and $t_{test} \sim \mathbb{U}([-7, 7])$, as it was done in the base paper [1].
The toy examples we used in this section are the following:

$$\begin{aligned} y_1(t) &= t^3 + n_1(t) \text{ with } n_1(t) \sim \mathbb{N}(0, \sigma^2_{n_1}(t)), \\ y_2(t) &= t^2 + n_2(t) \text{ with } n_2(t) \sim \mathbb{N}(0, \sigma^2_{n_2}(t)), \\ y_3(t) &= -t^3 + n_3(t) \text{ with } n_3(t) \sim \mathbb{N}(0, \sigma^2_{n_3}(t)), \\ y_4(t) &= t + n_4(t) \text{ with } n_4(t) \sim \mathbb{N}(0, \sigma^2_{n_4}(t)). \end{aligned} \tag{4.2}$$

For each dimension i, the σ_{n_i} used to sample the noise n_i is a bell-shaped curve. In our code, σ_{n_i} corresponds to the probability density function ϕ_{μ_i,σ_i} of the normal distribution with parameters μ_i and σ_i. More precisely, we have the following setting:

$$\begin{aligned} \sigma_{n_1}(t) &= \phi_{\mu_1,\sigma_1}(t) + 3 \text{ with } \mu_1 = 0 \text{ and } \sigma_1 = 3, \\ \sigma_{n_2}(t) &= \phi_{\mu_2,\sigma_2}(t) + 3 \text{ with } \mu_2 = 1 \text{ and } \sigma_2 = 3, \\ \sigma_{n_3}(t) &= \phi_{\mu_3,\sigma_3}(t) + 3 \text{ with } \mu_3 = -1 \text{ and } \sigma_3 = 3, \\ \sigma_{n_4}(t) &= \phi_{\mu_4,\sigma_4}(t) + 1 \text{ with } \mu_4 = 0 \text{ and } \sigma_4 = 3. \end{aligned} \tag{4.3}$$

See on Figure 4.9 the variations of the noise parameters σ_{n_i} defined in Equation 4.9.

We use $\lambda = 0.01$ for regularization, $lr = 5e^{-4}$ and $iterations = 1000$. We can see on Figure 4.8 the overall results of the training. On Figure 4.9, we can see the several σ_{n_i} noise applied on the training set, the distance between the prediction and ground truth, and the two uncertainties (aleatoric and epistemic).

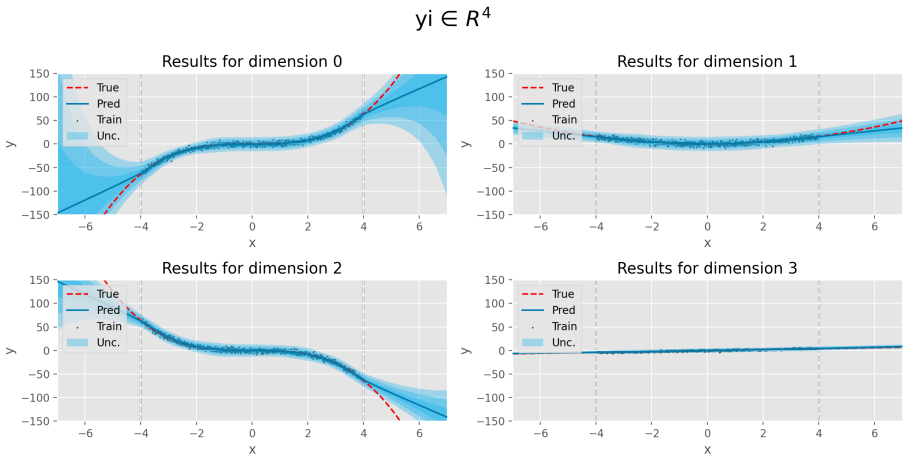


Figure 4.8: Overall Results for Multidimensional Case with Gaussian Noise

We can see on those graphs that the model regress consistent target functions and that it has more difficulty for OOD data. We can also notice that the epistemic uncertainty grows as we are further away from in-distribution data. The aleatoric uncertainty seems less consistent when the noise is not centered around 0 (see Figure 4.9).

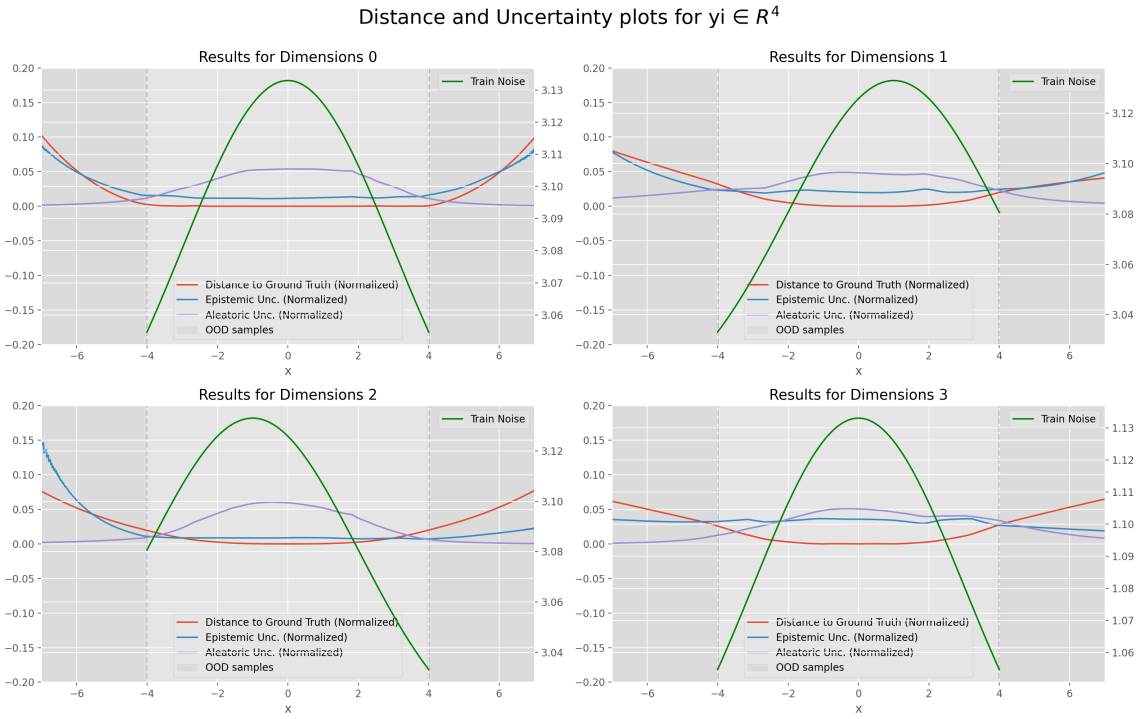


Figure 4.9: Plotting Distance to Ground Truth, Applied Gaussian Noise and Epistemic and Aleatoric Uncertainties for the Multidimensional Case

Conclusion

Under the assumption that the data is uniformly distributed and that there are no ”holes” in the data (meaning that the OOD data is located at the edges of the in-distribution interval), the Deep Evidential method seems to be extendable to the multidimensional case. We can still notice that the aleatoric uncertainty does not seem to detect correctly the applied noise when it is not centered around 0. We can also notice that the method seems to be working better when the toy case is an odd function.

4.2.2 Predicting a single dimension of uncertainty

This time, we consider that the dimensions are not independent. Thus, we predict only one set of evidential uncertainty parameters α, β, ν for each sample $y_i \in \mathbb{R}^4$ and we also apply the same noise on each dimension of y_i. The input data t is sampled such that $t_{train} \sim \mathbb{U}([-4, 4])$ and $t_{test} \sim \mathbb{U}([-7, 7])$.

Experimental Setup

We use for this sections the Neural Network architecture presented in Figure 3.2, but the last Linear Normal Gamma Layer has $outdim = 4$ as we predict a 1D-uncertainty parametrized by $\alpha, \beta, \nu \in \mathbb{R}$ for each sample $y_i \in \mathbb{R}^4$. We use the loss $\mathcal{L}_i$ defined in Equation 3.6 using the distance function d_{1D} defined in Equation 3.10. We have the following toy examples:

$$\begin{aligned} y_1(t) &= t^3 + n(t), \\ y_2(t) &= t^2 + n(t), \\ y_3(t) &= -t^3 + n(t), \\ y_4(t) &= t + n(t), \\ \text{with } n(t) &\sim \mathbb{N}(0, \sigma_n^2(t)). \end{aligned} \tag{4.4}$$

We used the same noise σ_n for each dimension i as we predict here a single dimension of

uncertainty. σ_n used to sample the noise n is a bell-shaped curve. In our code, σ_n corresponds to the probability density function $\phi_{\mu,\sigma}$ of the normal distribution with parameters $\mu = 0$ and $\sigma = 3$ and offset $= 0$. In short, we have the following setting for the noise:

$$\sigma_n(t) = \phi_{\mu,\sigma}(t) \text{ with } \mu = 0 \text{ and } \sigma = 3. \tag{4.5}$$

We use $\lambda = 0.005$ for regularization, $lr = 5e^{-4}$ and $iterations = 1000$. We can see on Figure 4.10 the overall results of the training. On Figure 4.11, we can see the overall σ_{n_i} noise applied on the training set, the distance between the prediction and ground truth, and the two uncertainties (aleatoric and epistemic).

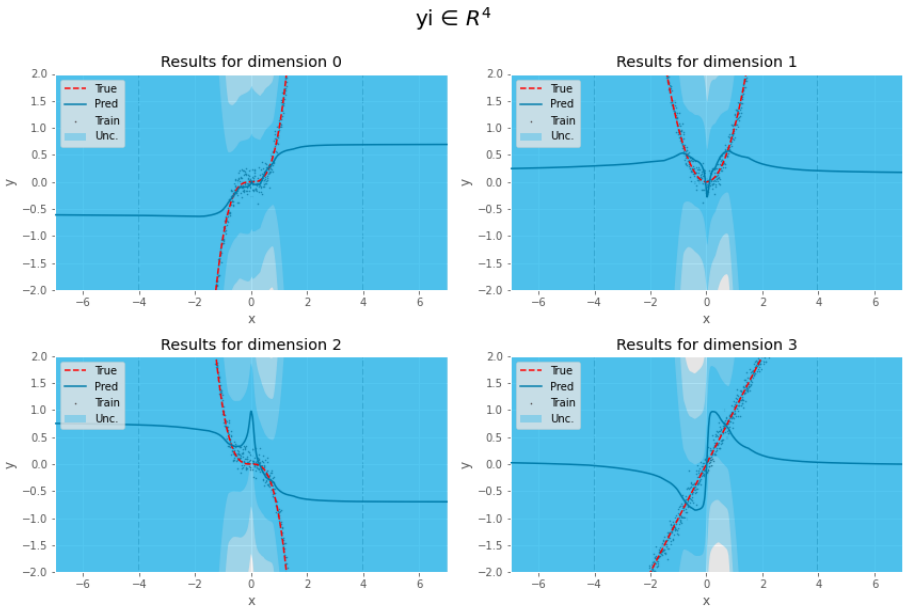


Figure 4.10: Overall Results for Multidimensional Case with Gaussian Noise - 1D uncertainty case

We can see on those graphs that the model does not regress consistent target functions. It seems to overfit on data centered around 0. We can also notice that the epistemic and aleatoric uncertainties are not consistent with neither the noise applied nor the data distribution.

Figure 4.11: Plotting Distance to Ground Truth, Applied Gaussian Noise and Epistemic and Aleatoric Uncertainties for the Multidimensional Case - 1D uncertainty case

Conclusion

It seems here that predicting one uncertainty for each dimensions leads to better results in the multidimensional case. It is important to note that even if the results are better, the model did not succeed to predict accurate epistemic and aleatoric uncertainties in the multidimensional case, which was our original goal.

4.3 Predicting an angle

4.3.1 Predicting a single dimension of uncertainty

This time we consider a 1-dimension target y such that it represents an angle in $\mathbb{R}$. To do that, we can use trigonometric functions. The toy example we used here is $y = sin(t) + k * 2\pi + n(t)$ where $k \in \mathbb{U}(\llbracket -6, 6 \rrbracket)$ and $n(t)$ is Gaussian Noise such that $n(t) \sim \mathcal{N}(0, \sigma_n^2(t))$ with $\sigma_n(t) \in [0, 1]$. We set $\sigma_n(t)$ such that it is small compared to π. The input data t is sampled such that $t_{train} \sim \mathbb{U}([-0.75\pi, 0.75\pi])$ and $t_{test} \sim \mathbb{U}([-\pi, \pi])$ as we are handling angles.

We use a similar architecture for the Neural Networks as described in section 4.2 but the last Linear Normal Gamma Layer has $outdim = 1$ as we are predicting a 1-dimension. The model uses the loss $\mathcal{L}_i$ defined by Equation 3.6 using the distance function described in Equation 3.12.

Experimental Setup

Once again, we sample the data such that $y = sin(t) + k * 2\pi + n(t)$ where $k \in \mathbb{U}(\llbracket -6, 6 \rrbracket)$ and $n(t)$ is Gaussian Noise such that $n(t) \sim \mathcal{N}(0, \sigma_n^2(t))$ where σ_n is a bell-shaped function. In our experiments, σ_n corresponds to density probability function $\phi_{\mu,\sigma}$ of a normal distribution with parameters $\mu = 0$ and $\sigma = 1$. That is, we have the following experimental setup:

$$\begin{gathered} y(t) = sin(t) + k * 2\pi + n(t) \text{ with } n(t) \sim \mathbb{N}(0, \sigma_n^2(t)) \text{ and } k \in \mathbb{U}(\llbracket -6, 6 \rrbracket), \\ \sigma_n(t) = \phi_{\mu,\sigma}(t) \text{ with } \mu = 0 \text{ and } \sigma = 0.5. \end{gathered} \tag{4.6}$$

As a remainder, the idea is to apply varying noise to assess whether the aleatoric uncertainty detects and is adapted to this known variation.

We introduced two alternative distance functions d_{geo} and $d_{S(1)}$ in Equation 3.12 for this $S(1)$ case. Let us now compare them.

We first use $\lambda = 0.1$ for regularization, $lr = 5e^{-4}$ and $epochs = 1000$ and the distance function $d_{S(1)}$. The results are shown on Figure 4.12.

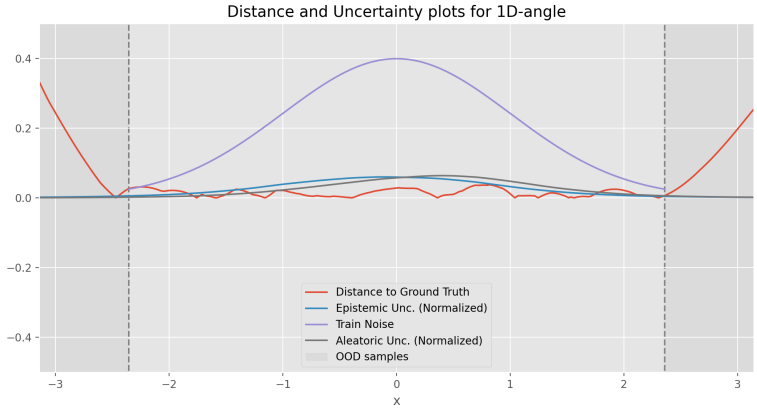


Figure 4.12: Plotting the Distance $d_{S(1)}$ to Ground Truth, Applied Gaussian Noise, Epistemic and Aleatoric Uncertainties for the 1D-angle case

Now, we use the same set of hyper-parameters $\lambda = 0.1$ for regularization, $lr = 5e^{-4}$ and $epochs = 1000$ and the distance function d_{geo}. The results are shown on Figure 4.13.

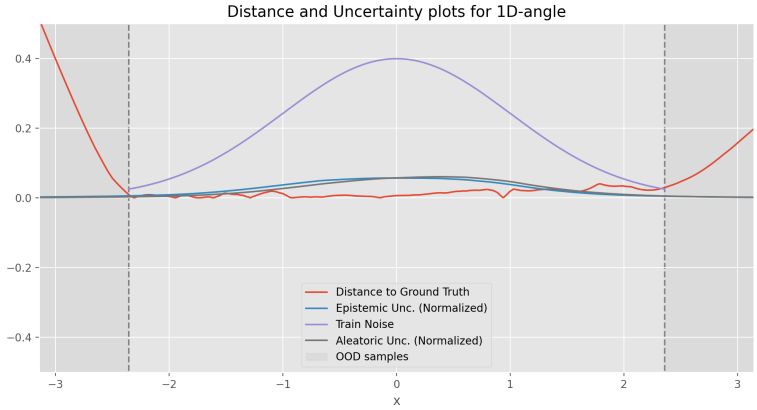


Figure 4.13: Plotting the Distance d_{geo} to Ground Truth, Applied Gaussian Noise, Epistemic and Aleatoric Uncertainties for the 1D-angle case

Conclusion

We can see on Figure 4.12 that the aleatoric uncertainty is consistent with the noise applied and that the distance between the ground truth and predicted output grows as we go further in the OOD set. This time, the epistemic uncertainty does not follow the OOD data. Besides, even when using distinct distance functions in $S(1)$, the results are equivalent and,

in both cases, the Deep Evidential method fails to predict reliable uncertainty measure in $S(1)$.

4.4 Points on a sphere: the S(2) case

We know consider a 3-dimensional target y_i in $S(2)$.

4.4.1 Predicting a single dimension of uncertainty

This time, we consider that the dimensions are not independent. Thus, we predict only one set of evidential uncertainty parameters α, β, ν for each sample y_i and we also apply the same noise on each dimension of y_i. The input data t is sampled such that $t_{train} \sim \mathbb{U}([-0.75\pi, 0.75\pi])$ and $t_{test} \sim \mathbb{U}([-\pi, \pi])$ as we use trigonometric functions to sample the data points.

Experimental Setup

We use for this sections the Neural Network architecture presented in Figure 3.2, but the last Linear Normal Gamma Layer has $outdim = 3$ as we predict a 1D-uncertainty parametrized by $\alpha, \beta, \nu \in \mathbb{R}$ for each sample $y_i \in S(2)$. We use the loss $\mathcal{L}_i$ defined in Equation 3.6 using the distance function in $S(2)$ defined in Equation 3.18. We have the following toy examples:

$$\begin{aligned} y_1(t) &= sin(t) + n(t), \\ y_2(t) &= cos(4t) + n(t), \\ y_3(t) &= sin(2t) + n(t), \\ \text{where } n(t) &\sim \mathbb{N}(0, \sigma_n^2(t)), \\ \sigma_n(t) &= \phi_{\mu,\sigma}(t) \text{ with } \mu = 0 \text{ and } \sigma = 1. \end{aligned} \tag{4.7}$$

We normalize y before training $y = \frac{y}{\|y\|}$ so that it ensures $\|y\| = 1$ as it is a property of points in $S(2)$. We use $\lambda = 0.5$ for regularization, $lr = 5e^{-4}$ and $epochs = 500$. The

results are shown on Figure 4.14.

Figure 4.14: Plotting Geodesic Distance to Ground Truth, Applied Gaussian Noise, Epistemic and Aleatoric Uncertainties for the S(2) case

Conclusion

The model outputs the consistent regression outputs as the distance to ground truth samples is close to 0 in Figure 4.14. The aleatoric uncertainty predicted by the model is also consistent with the known noise applied on the training data. Nonetheless, the epistemic uncertainty mimics the aleatorics uncertainty and does not detect OOD samples.

4.5 2D Special Euclidean Group SE(2) case

4.5.1 Predicting multiple uncertainty measures

For this case, we consider a 3-dimension target $y \in \mathbb{R}^3$ such that $y = [T, R]$ respectively representing the translation $T \in \mathbb{R}^2$ and rotation $R \in \mathbb{R}$ parts. We use $T_i = f_i(t) + n_i(t)$ where $f_i(\cdot)$ is an odd function in $\mathbb{R}$ centered around 0 and $n_i(t)$ is Gaussian Noise such that $n_i \sim \mathcal{N}(0, \sigma^2_{n_i}(t))$. The input data t is sampled such that $t_{train} \sim \mathbb{U}([-0.75\pi, 0.75\pi])$ and $t_{test} \sim \mathbb{U}([-\pi, \pi])$.

Experimental Setup

We use an architecture for the Neural Networks similar to the one described in section 4.2 but the last Linear Normal Gamma Layer has $outdim = 3$ as we are predicting a 2 independent translations and 1 rotation angle in $SE(2)$. We use the corresponding loss $\mathcal{L}_i$ in $SE(2)$ defined in Equation 3.14. The toy examples we use in this section are the following:

$$\begin{aligned} T_1 &= t^3 + n_1(t) \text{ with } {}_1(t) \sim \mathbb{N}(0, \sigma^2_{n_1}(t)), \\ T_2 &= -t^3 + n_2(t) \text{ with } {}_2(t) \sim \mathbb{N}(0, \sigma^2_{n_2}(t)), \\ R &= sin(t) + k * 2\pi + n_3(t) \text{ with } {}_3(t) \sim \mathbb{N}(0, \sigma^2_{n_3}(t)) \text{ and } k \in \mathbb{U}(\llbracket -6, 6 \rrbracket). \end{aligned} \tag{4.8}$$

As discussed in section 4.2, we sample σ_{n_i} such that it is a bell-shaped function. We chose to define σ_{n_i} as the density probability function ϕ_{μ_i,σ_i} of a normal distribution with parameters μ_i and σ_i. We use the following noise parameters:

$$\begin{aligned} \sigma_{n_1}(t) &= \phi_{\mu_1,\sigma_1}(t) + 3 \text{ with } \mu_1 = 0 \text{ and } \sigma_1 = 3, \\ \sigma_{n_2}(t) &= \phi_{\mu_2,\sigma_2}(t) \text{ with } \mu_2 = 1 \text{ and } \sigma_2 = 3, \\ \sigma_{n_3}(t) &= 0.5\phi_{\mu_3,\sigma_3}(t) \text{ with } \mu_3 = -1 \text{ and } \sigma_3 = 3. \end{aligned} \tag{4.9}$$

We respectively set $\lambda_t = 0.1$ and $\lambda_r = 0.6$ for regularization on translation and rotation, $lr = 5e^{-4}$ and $epochs = 2000$. We can see on Figure 4.15 the visualization of the distance between ground truth and predictions, the applied noise and the corresponding uncertainties for both in- and out-of-distribution samples.

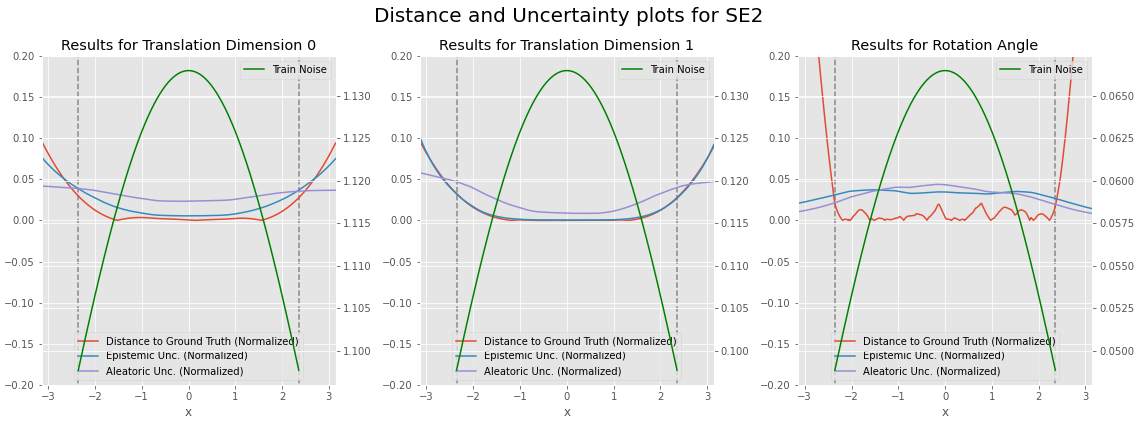


Figure 4.15: Plotting Distance to Ground Truth, Applied Gaussian Noise, Epistemic and Aleatoric Uncertainties for the SE(2) case

The epistemic uncertainty grows as the model predicts data out of the training distribution for both translations and rotation. The aleatoric uncertainty seems more stable for the rotation part: indeed, it detects and follows the distribution of the noise applied in the training data. On the other hand, the epistemic uncertainty is more consistent on the translation part than the rotation part.

Conclusion

In short, the method outputs somewhat consistent regression predictions in $SE(2)$ case: indeed, the distance to ground truth is close to 0 for in-distribution samples and increases for OOD samples. The epistemic uncertainty is more stable for translations while the aleatoric uncertainty is better for the rotation part. Once again, the Deep Evidential framework does not guarantee reliable uncertainty prediction in $SE(2)$.

4.5.2 Predicting a single dimension of uncertainty

This time, we consider that the dimensions are not independent. Thus, we predict only one set of evidential uncertainty parameters α, β, ν for each sample $y_i \in SE(2)$ and we also apply the same noise on each dimension of y_i. The input data t is sampled such that $t_{train} \sim \mathbb{U}([-0.75\pi, 0.75\pi])$ and $t_{test} \sim \mathbb{U}([-\pi, \pi])$ as we use trigonometric functions to sample the data points for the rotation part.

Experimental Setup

We use for this sections the Neural Network architecture presented in Figure 3.2, but the last Linear Normal Gamma Layer has $outdim = 3$ as we predict a 1D-uncertainty parametrized by $\alpha, \beta, \nu \in \mathbb{R}$ for each sample $y_i \in SE(2)$. We use the loss $\mathcal{L}_i$ defined in Equation 3.6 using the distance function in $SE(2)$ defined in Equation 3.16. We have the following toy examples:

$$\begin{gathered} y_1(t) = t^3 + n(t), \\ y_2(t) = -t^3 + n(t), \\ y_3(t) = sin(2t) + k * 2\pi + n(t) \text{ with } k \in \mathbb{U}(\llbracket -6, 6 \rrbracket), \\ \text{where } n(t) \sim \mathbb{N}(0, \sigma_n^2(t)), \\ \sigma_n(t) = 0.5 * \phi_{\mu,\sigma}(t) \text{ with } \mu = 0 \text{ and } \sigma = 3. \end{gathered} \tag{4.10}$$

We use $\lambda_d = 0.1$ to weigh rotation in Equation 3.16, $\lambda = 0.1$ for regularization, $lr = 5e^{-4}$ and $epochs = 2000$. The results are shown on Figure 4.16.

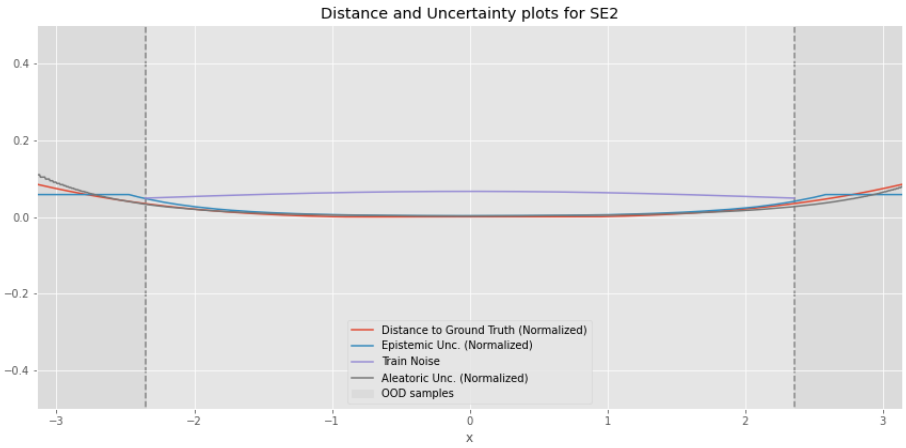


Figure 4.16: Plotting Geodesic Distance to Ground Truth, Applied Gaussian Noise, Epistemic and Aleatoric Uncertainties for the S(2) case - 1D uncertainty case

Conclusion

Based on our results, it seems that predicting a 1D uncertainty in $SE(2)$ does not enable an reliable uncertainty predictions. Indeed, as we can see in Figure 4.16, the aleatoric uncertainty is not consistent with the noise we applied, while the epistemic uncertainty gives somewhat relevant results. The two methods (1D versus multiple uncertainties) for Deep Evidential Regression do not guarantee a reliable uncertainty estimation in $SE(2)$.

4.6 3D Orthogonal Group SO(3) case

4.6.1 Predicting a single dimension of uncertainty

We consider in this section a 4-dimension target $y \in \mathbb{R}^4$ representing quaternion $q = [q_x, q_y, q_z, q_w]$ in $SO(3)$. Since we are dealing with rotations, we consider targets q such that $\|q\| = 1$. The input data t is sampled such that $t_{train} \sim \mathbb{U}([-0.75\pi, 0.75\pi])$ and $t_{test} \sim \mathbb{U}([-\pi, \pi])$.

Experimental Setup

In this section, we used the architecture of the Neural Network described in Figure 3.2: the main idea is that we predict one global 1-D uncertainty for each quaternion (instead of one uncertainty measure for each dimension of the quaternion). The main assumption behind that is that the dimensions of a quaternions are not independent as explained in subsection 3.2.5. We implemented the loss $\mathcal{L}_i$ defined in Equation 3.6 using the distance function in $SO(3)$ described in Equation 3.24.

This quaternion sampling method is the that one guarantees $d_{quat}(q_{raw}, q_{new}) = n$. The method is described in subsection 3.2.5. The main idea is that, to add noise in the training data, we use the axis-angle representation. In short, we iteratively sample an angle $\alpha \sim \mathbb{N}(\mu, \sigma_\alpha^2)$ and compute the corresponding quaternion q_{noise} using Equation 3.27. The resulting noisy quaternion is obtained with $q_{new} = q_{raw} \cdot q_{noise}$ where $\cdot$ is the group operator in $SO(3)$.

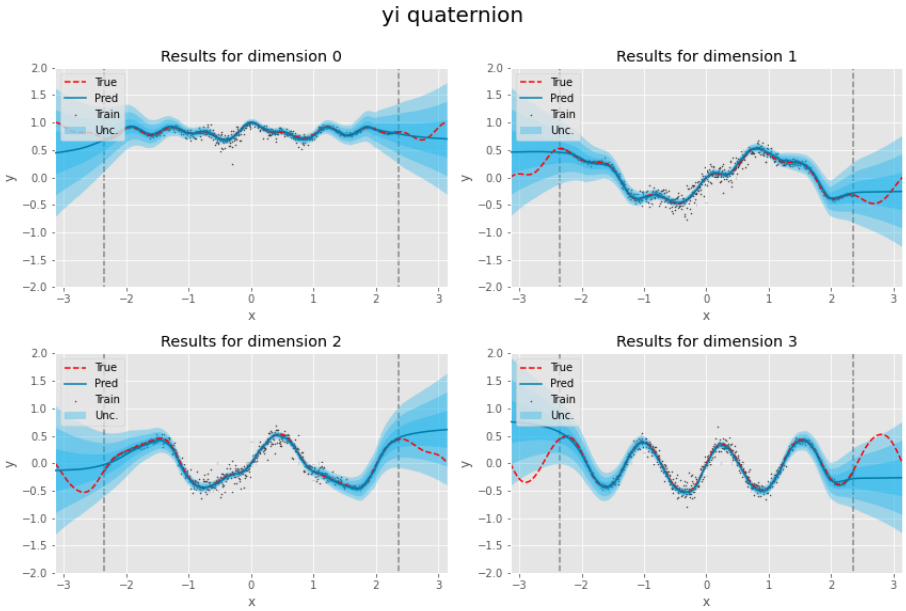


Figure 4.17: Overall results with Gaussian Noise for the SO(3) case

To sample the quaternions using this method, we first computed the following Euler angles :

$$\begin{aligned} roll(t) &= sin(t), \\ pitch(t) &= sin(3t), \\ yaw(t) &= sin(5t). \end{aligned} \tag{4.11}$$

Then, we use Equation 3.25 to compute the raw quaternion samples $q_{raw}(t)$. The next step is to add noise in the data. To do that, we sample an angle $\alpha(t)$ such that $\alpha(t) \sim \mathbb{N}(0, \sigma_\alpha^2(t))$ where σ_α is the density probability function $\phi_{\mu,\sigma}$ of a normal distribution with parameters $\mu = 0$ and $\sigma = 3$. More precisely, we sample the following noise quaternion $q_{noise}(t)$ using:

$$\begin{gathered} q_{noise}(t) = \left[cos\left(\frac{\alpha(t)}{2}\right),\ sin\left(\frac{\alpha(t)}{2}\right) u \right], \\ \text{where } u \text{ is a vector randomly sampled in } [1,1]^3 \text{ and } u \neq [0,0,0] \\ \text{and } \alpha(t) \sim \mathbb{N}(0, \sigma_\alpha^2(t)), \\ \sigma_\alpha(t) = \phi_{\mu,\sigma}(t) \text{ with } \mu = 0 \text{ and } \sigma = 3. \end{gathered} \tag{4.12}$$

The resulting noisy samples are obtained using:

$$q_{noisy}(t) = q_{raw}(t) \cdot q_{noise}(t) \tag{4.13}$$

where $\cdot$ is the group operator in $SO(3)$.

We used the following parameters for training the model $\lambda = 0.03$ for regularization, $lr = 5e^{-4}$ and $epochs = 2000$.

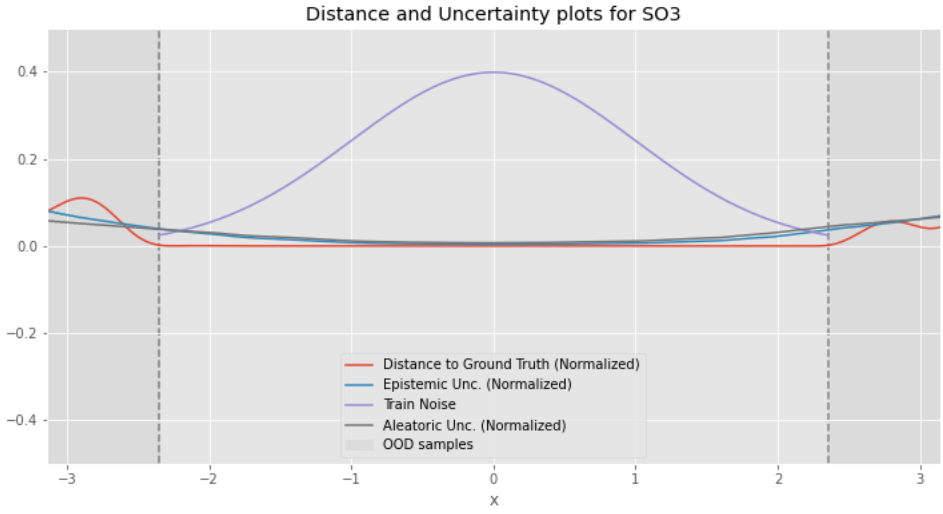


Figure 4.18: Plotting Geodesic Distance to Ground Truth, Gaussian Noise, Epistemic and Aleatoric Uncertainties for the SO(3)

We can see on Figure 4.17 that the model correctly regresses the target functions and the epistemic uncertainty is consistent as it is higher for OOD samples that it is for in-distribution samples. But on Figure 4.18, we can see that the aleatoric uncertainty does not follow the distribution of the applied noise.

Conclusion

It seems that the model can be applied to the $SO(3)$ case. But the method does not provide consistent aleatoric uncertainty measures. It may be coming from the way we sampled the data or the hyper-parameters tuning, or that the Deep Evidential loss $\mathcal{L}_i$ is not robust enough and should be used under stricter assumptions. In any case, the Deep Evidential framework fails to predict reliable uncertainty measure in $SO(3)$.

4.7 3D Special Euclidean Group SE(3)

4.7.1 Predicting multiple uncertainty measures

We consider a target output $y \in SE(3)$ such that $y = [T, q]$ where $T \in \mathbb{R}^3$ and $q \in SO(3)$ respectively represent the translation an rotation parts of a 3D transformation. The input data t is sampled such that $t_{train} \sim \mathbb{U}([-0.75\pi, 0.75\pi])$ and $t_{test} \sim \mathbb{U}([-\pi, \pi])$.

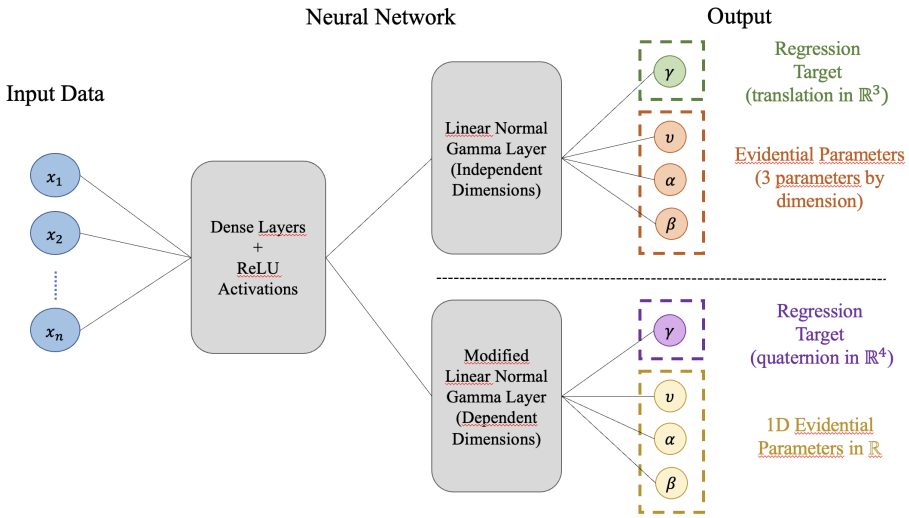


Figure 4.19: Modified Deep Evidential NN in SE(3)

Experimental Setup

We used the same method to sample q as used in the previous section about $SO(3)$. More precisely, we used Equation 4.11, Equation 4.12 and Equation 4.13 to sample the quaternions here. For the translation part, we used the following equations:

$$\begin{aligned} T_1 &= t^2 + n_{t_1}, \\ T_2 &= -t^2 + n_{t_2}, \\ T_3 &= t^3 + n_{t_3}, \end{aligned} \tag{4.14}$$

$$\text{with } n_{t_1}, n_{t_2}, n_{t_3} \sim \mathcal{N}(0, \sigma_t^2) \text{ and } \sigma_t = 0.4.$$

To train the model, we respectively used $\lambda_t = 0.01$ and $\lambda_q = 0.01$ for regularization on translation and rotation, $lr = 5e^{-4}$ and $epochs = 1500$.

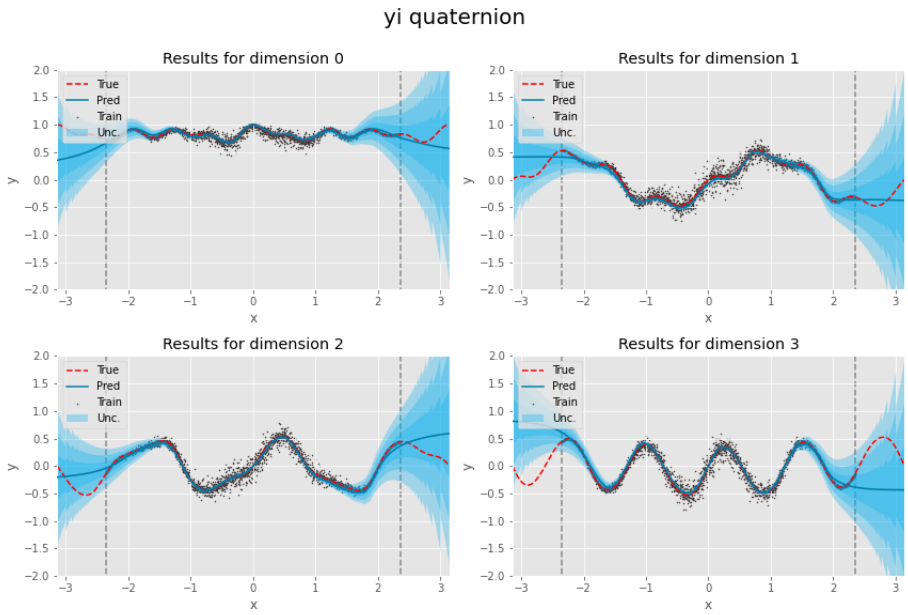


Figure 4.20: Overall results with Gaussian Noise for the SE(3) case : Rotation Part

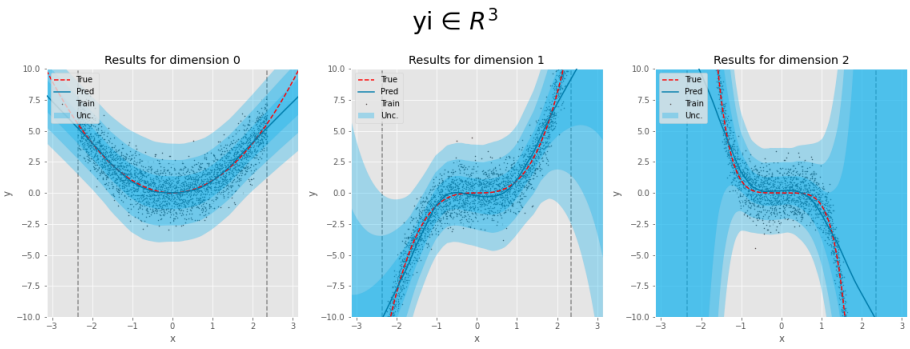


Figure 4.21: Overall results with Gaussian Noise for the SE(3) case : Translation Part

We can see on Figure 4.21 and Figure 4.20 that the model predicts accurate results. It seems that the model is more confident on the predicted orientation part than the predicted translation part. On Figure 4.22, we can notice that the predicted epistemic uncertainty is consistent as it is higher for OOD samples that it is for in-distribution samples. But the aleatoric uncertainty does not follow the distribution of the applied noise.

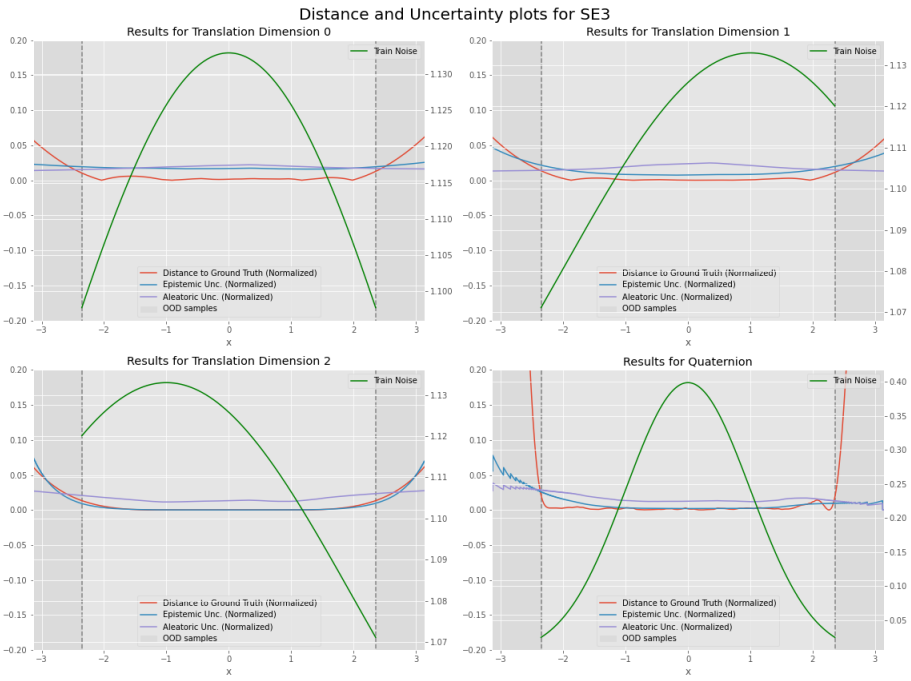


Figure 4.22: Plotting Geodesic Distance to Ground Truth, Gaussian Noise, Epistemic and Aleatoric Uncertainties for the SE(3) case

Conclusion

The Deep Evidential Framework may seem promising as the model learns the trajectories as we can see on Figure 4.20 and Figure 4.21, meaning that the model correctly regress the target functions in $SE(3)$. But the method only gives a consistent epistemic uncertainty while the predicted aleatoric uncertainties are not reliable, as seen on Figure 4.22. This may be coming from the toy examples we chose: there may some limitations due to how we sampled the training data or added the noise. Another thing is that we made the assumption that translation and rotation were independent, but it may be a strong assumption leading to limitations in the results. Further work could be done on removing this assumption and developing a multivariate framework as proposed by [27].

4.7.2 Predicting a single dimension of uncertainty

This time, we consider that the dimensions are not independent. Thus, we predict only one set of evidential uncertainty parameters α, β, ν for each sample $y_i \in SE(3)$ and we also apply the same noise on each dimension of y_i. The input data t is sampled such that $t_{train} \sim \mathbb{U}([-0.75\pi, 0.75\pi])$ and $t_{test} \sim \mathbb{U}([-\pi, \pi])$ as we use trigonometric functions to sample the data points for the rotation part.

Experimental Setup

We use for this sections the Neural Network architecture presented in Figure 3.2, but the last Linear Normal Gamma Layer has $outdim = 7$ as we predict a 1D-uncertainty parametrized by $\alpha, \beta, \nu \in \mathbb{R}$ for each sample $y_i \in SE(3)$. We use the loss $\mathcal{L}_i$ defined in Equation 3.6 using the distance function in $SE(3)$ defined in Equation 3.31. Since we are predicting a single uncertainty over all dimensions, we applied the same noise to all the dimensions. We used the same method to sample q as used in the previous section about $SO(3)$. More precisely, we used Equation 4.11, Equation 4.12 and Equation 4.13 to sample the quaternions here. For the translation part, we used the following equations:

$$\begin{aligned} T_1 &= t^2 + n_t, \\ T_2 &= -t^2 + n_t, \\ T_3 &= t^3 + n_t, \\ \text{with } n_t &\sim \mathcal{N}(0, \sigma_t^2) \text{ and } \sigma_t = 1. \end{aligned} \tag{4.15}$$

We use the following set of hyper-parameters for training: $\lambda_d = 0.1$ to weigh rotation in Equation 3.16, $\lambda = 0.01$ for regularization, $lr = 5e^{-4}$ and $epochs = 2000$. The results are shown on Figure 4.23.

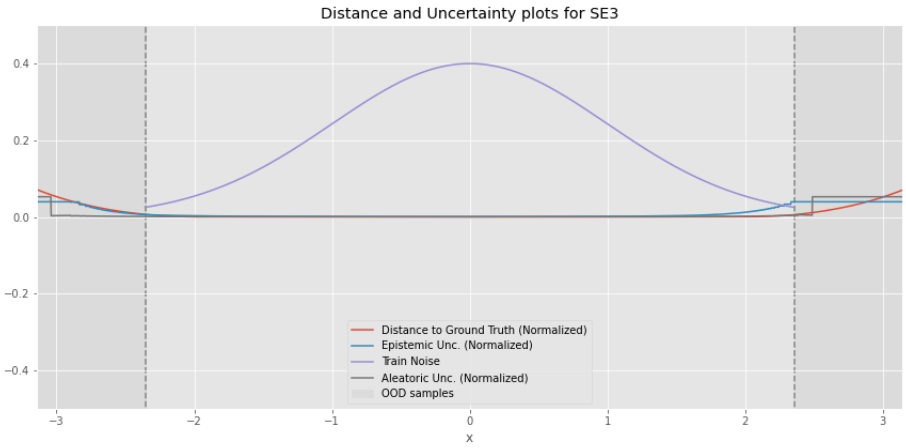


Figure 4.23: Plotting Geodesic Distance to Ground Truth, Gaussian Noise, Epistemic and Aleatoric Uncertainties for the SE(3) case - 1D uncertainty case

We decided to predict a 1D uncertainty over all dimensions with hopes that the model would predict more reliable uncertainty measures than when it predicts multiple uncertainties. As we can see on Figure 4.23, it seems that the Deep Evidential framework once again fails to predict a reliable aleatoric uncertainty.

Conclusion

In short, the method outputs somewhat consistent regression predictions in $SE(3)$ case: indeed, the distance to ground truth is close to 0 for in-distribution samples and increases for OOD samples. Regarding the 1D uncertainty measure, the epistemic also seems somewhat consistent with the distribution of data, while the predicted aleatoric uncertainty is wrong. But in any case, the Deep Evidential framework does not guarantee reliable uncertainty prediction in $SE(3)$.

CHAPTER 5
CONCLUSION AND FUTURE WORK

The goal of this work was to first reproduce the experiments from the base paper [1], and second, build a general framework to extend the Deep Evidential method to higher dimension manifolds and groups of the Lie algebra.

We were able to reproduce the base experiments in $\mathbb{R}$, but noticed that the Deep Evidential method is unstable and heavily depends on both the hyper-parameters, as well as the toy examples at hand. In particular, we noticed that the prediction of the aleatoric uncertainty is the most problematic, as it does not give consistent result even when used in the same parameters as the base paper.

We developed a general framework to extend the Deep Evidential Regression approach introduced by [1] to multiple groups and manifolds of the Lie algebra. We showed that the stability and success of the method on each manifolds highly depends on its structure and the distance function used to compute the Deep Evidential Loss $\mathcal{L}_i$.

The method introduced by [1] seemed promising as it was presented as a way to predict two types of uncertainties (aleatoric and epistemic). This area of study is major and could allow great advances in the field of robotics as the most challenging part of 6D Pose Estimation in the field of robotics is that we often are in a safety-critical context. In this context, it is crucial that the model not only predicts accurate poses, but gives a reliable uncertainty measure over its predictions, resist to adversarial attacks to prevent malicious hacking and detect In- and OOD samples [5] (as the model is less likely to perform well on OOD data).

We used the Deep Evidential Framework in the context of 6D Pose Estimation in $SE(3)$, as well as other manifolds of the Lie algebra, but never got reliable results for both the aleatoric and epistemic uncertainties. Once again, the different models had more difficulty to predict an aleatoric uncertainty consistent with the way we applied noise on data.

Future work could still be done on applying the framework introduced to real data and state-of-the-art Deep Neural Networks for 6D Pose Estimation, such as [8, 7], and compare the uncertainty performance with other similar methods (e.g. the Bingham Loss [10]).

In conclusion, we showed that the Deep Evidential method has several limitations as it seems unstable, heavily depends on hyper-parameter tuning and does not predict reliable uncertainty measures. The next steps would be to consider using the Multivariate Loss introduced in [27]. Indeed, [27] is a continuation of the base paper [1] that also mentions and proposes solution to have a more stable and robust uncertainty estimation. Moreover, the fact that they predict multivariate uncertainties would enable to consider any set of examples.

REFERENCES

[1] A. Amini, W. Schwarting, A. Soleimany, and D. Rus, "Deep evidential regression," *Advances in Neural Information Processing Systems*, vol. 33, pp. 14 927–14 937, 2020.

[2] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *CoRR*, vol. abs/1810.04805, 2018. arXiv: 1810.04805.

[3] M. Bakator and D. Radosav, "Deep learning and medical diagnosis: A review of literature," *Multimodal Technologies and Interaction*, vol. 2, no. 3, 2018.

[4] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of deep learning techniques for autonomous driving," *Journal of Field Robotics*, vol. 37, no. 3, pp. 362–386, 2020. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/rob.21918.

[5] J. Gawlikowski *et al.*, "A survey of uncertainty in deep neural networks," *CoRR*, vol. abs/2107.03342, 2021. arXiv: 2107.03342.

[6] Y. Gal and Z. Ghahramani, *Dropout as a bayesian approximation: Representing model uncertainty in deep learning*, 2015.

[7] H. Wang, S. Sridhar, J. Huang, J. Valentin, S. Song, and L. J. Guibas, "Normalized object coordinate space for category-level 6d object pose and size estimation," *CoRR*, vol. abs/1901.02970, 2019. arXiv: 1901.02970.

[8] C. Wang *et al.*, "Densefusion: 6d object pose estimation by iterative dense fusion," *CoRR*, vol. abs/1901.04780, 2019. arXiv: 1901.04780.

[9] M. Abdar *et al.*, "A review of uncertainty quantification in deep learning: Techniques, applications and challenges," *Information Fusion*, vol. 76, pp. 243–297, 2021.

[10] I. Gilitschenski, R. Sahoo, W. Schwarting, A. Amini, S. Karaman, and D. Rus, "Deep orientation uncertainty learning based on a bingham loss," in *International Conference on Learning Representations*, 2020.

[11] G. Shi *et al.*, "Fast uncertainty quantification for deep object pose estimation," *CoRR*, vol. abs/2011.07748, 2020. arXiv: 2011.07748.

[12] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *Proceedings of The 33rd International Con-*

ference on Machine Learning, M. F. Balcan and K. Q. Weinberger, Eds., ser. Proceedings of Machine Learning Research, vol. 48, New York, New York, USA: PMLR, 20–22 Jun 2016, pp. 1050–1059.

[13] M. Sensoy, L. Kaplan, and M. Kandemir, "Evidential deep learning to quantify classification uncertainty," in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31, Curran Associates, Inc., 2018.

[14] A. Subramanya, S. Srinivas, and R. V. Babu, "Confidence estimation in deep neural networks via density modelling," *CoRR*, vol. abs/1707.07013, 2017. arXiv: 1707.07013.

[15] M. Hein, M. Andriushchenko, and J. Bitterwolf, "Why relu networks yield high-confidence predictions far away from the training data and how to mitigate the problem," *CoRR*, vol. abs/1812.05720, 2018. arXiv: 1812.05720.

[16] A. Meinke and M. Hein, "Towards neural networks that provably know when they don't know," *CoRR*, vol. abs/1909.12180, 2019. arXiv: 1909.12180.

[17] L. Hansen and P. Salamon, "Neural network ensembles," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 10, pp. 993–1001, 1990.

[18] T. G. Dietterich, "Ensemble methods in machine learning," in *International workshop on multiple classifier systems*, Springer, 2000, pp. 1–15.

[19] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," *Advances in neural information processing systems*, vol. 30, 2017.

[20] A. Ashukha, A. Lyzhov, D. Molchanov, and D. Vetrov, "Pitfalls of in-domain uncertainty estimation and ensembling in deep learning," *arXiv preprint arXiv:2002.06470*, 2020.

[21] J. Gawlikowski *et al.*, "A survey of uncertainty in deep neural networks," *arXiv preprint arXiv:2107.03342*, 2021.

[22] Y. Gal *et al.*, "Uncertainty in deep learning,"

[23] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?" *Advances in neural information processing systems*, vol. 30, 2017.

[24] R. M. Neal, *Bayesian learning for neural networks*. Springer Science & Business Media, 2012, vol. 118.

[25] V. Peretroukhin, M. Giamou, D. M. Rosen, W. N. Greene, N. Roy, and J. Kelly, "A smooth representation of belief over SO(3) for deep rotation learning with uncertainty," *CoRR*, vol. abs/2006.01031, 2020. arXiv: 2006.01031.

[26] H. Deng, M. Bui, N. Navab, L. J. Guibas, S. Ilic, and T. Birdal, "Deep bingham networks: Dealing with uncertainty and ambiguity in pose estimation," *CoRR*, vol. abs/2012.11002, 2020. arXiv: 2012.11002.

[27] N. Meinert and A. Lavin, "Multivariate deep evidential regression," *CoRR*, vol. abs/2104.06135, 2021. arXiv: 2104.06135.

[28] A. Kendall and R. Cipolla, "Geometric loss functions for camera pose regression with deep learning," *CoRR*, vol. abs/1704.00390, 2017. arXiv: 1704.00390.

[29] T. D. Barfoot, *State Estimation for Robotics*, 1st. USA: Cambridge University Press, 2017, ISBN: 1107159393.

[30] J. J. Kuffner, "Effective sampling and distance metrics for 3d rigid body path planning," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, IEEE, vol. 4, 2004, pp. 3993–3998.

[31] D. Q. Huynh, "Metrics for 3d rotations: Comparison and analysis," *Journal of Mathematical Imaging and Vision*, vol. 35, no. 2, pp. 155–164, 2009.

[32] M. D. Shuster *et al.*, "A survey of attitude representations," *Navigation*, vol. 8, no. 9, pp. 439–517, 1993.

[33] F. L. Markley, Y. Cheng, J. L. Crassidis, and Y. Oshman, "Averaging quaternions," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 4, pp. 1193–1197, 2007.

[34] H. A. Ardakani and T. Bridges, "Review of the 3-2-1 euler angles: A yaw-pitch-roll sequence," *Department of Mathematics, University of Surrey, Guildford GU2 7XH UK, Tech. Rep*, 2010.

[35] J. M. Hernández-Lobato and R. Adams, "Probabilistic backpropagation for scalable learning of bayesian neural networks," in *International conference on machine learning*, PMLR, 2015, pp. 1861–1869.