Using pseudo-sensors to improve egomotion estimation for mobile autonomy

by

Valentin Peretroukhin

A thesis submitted in conformity with the requirements
for the degree of Doctor of Philosophy
Graduate Department of Institute for Aerospace Studies
University of Toronto

© Copyright 2019 by Valentin Peretroukhin

Abstract

Using pseudo-sensors to improve egomotion estimation for mobile autonomy

Valentin Peretroukhin
Doctor of Philosophy
Graduate Department of Institute for Aerospace Studies
University of Toronto
2019

The ability to estimate *egomotion*, that is, to track one's own pose through an unknown environment, is at the heart of safe and reliable mobile autonomy. By inferring pose changes from sequential sensor measurements, egomotion estimation forms the basis of mapping and navigation pipelines, and permits mobile robots to self-localize within environments where external localization sources are intermittent or unavailable. Visual and inertial egomotion estimation, in particular, have become ubiquitous in mobile robotics due to the availability of high-quality, compact, and inexpensive sensors that capture rich representations of the world. To remain computationally tractable, 'classical' visual-inertial pipelines (like visual odometry and visual SLAM) make simplifying assumptions that, while permitting reliable operation in ideal conditions, often lead to systematic error. In this thesis, we present several data-driven *pseudo-sensors* that serve to complement conventional pipelines by inferring latent information from the same data stream. Our approach retains much of the benefits of traditional pipelines, while leveraging high-capacity hyper-parametric models to extract complementary information that can be used to improve uncertainty quantification, correct for systematic bias, and improve robustness to difficult-to-model deleterious effects. We validate our *pseudo-sensors* on several kilometres of sensor data collected in sundry settings such as urban roads, indoor labs, and planetary analogue sites in the Canadian high arctic.

Epigraph

A little learning is a dangerous thing;
drink deep, or taste not the Pierian
spring: there shallow draughts
intoxicate the brain, and drinking
largely sobers us again.

Alexander Pope

The universe is no narrow thing and the order within it is not constrained by any latitude in its conception to repeat what exists in one part in any other part. Even in this world more things exist without our knowledge than with it and the order in creation which you see is that which you have put there, like a string in a maze, so that you shall not lose your way. For existence has its own order and that no man's mind can compass, that mind itself being but a fact among others.

Cormac McCarthy

Elephants don't play chess.

Rodney Brooks

To all those who encouraged (or, at least, *never discouraged*) my intellectual wanderlust.

Acknowledgements

This document would not have been possible without the generous support and guidance of my supervisor[1], the perennial love of my family and friends[2], and the limitless patience of my lab mates[3]. Thank you all.

[1] as well as all my collaborators and academic mentors
[2] especially the support and encouragement of Elyse
[3] in humouring my insatiable need for debate and banter

Contents

Notation

a	:	Symbols in this font are real scalars.
$\mathbf{a}$	:	Symbols in this font are real column vectors.
$\mathbf{A}$	:	Symbols in this font are real matrices.
$\mathcal{N}(\boldsymbol{\mu}, \mathbf{R})$	:	Normally distributed with mean $\boldsymbol{\mu}$ and covariance $\mathbf{R}$.
$E[\cdot]$	:	The expectation operator.
$\underrightarrow{\boldsymbol{\mathcal{F}}}_a$	:	A reference frame in three dimensions.
$(\cdot)^\wedge$	:	An operator associated with the Lie algebra for rotations and poses. It produces a matrix from a column vector.
$(\cdot)^\vee$	:	The inverse operation of $(\cdot)^\wedge$
$\mathbf{1}$	:	The identity matrix.
$\mathbf{0}$	:	The zero matrix.
$\mathbf{p}_a^{c,b}$	:	A vector from point b to point c (denoted by the superscript) and expressed in $\underrightarrow{\boldsymbol{\mathcal{F}}}_a$ (denoted by the subscript). This vector can be in homogenous coordinates depending on context.
$\mathbf{C}_{a,b}$	:	The 3×3 rotation matrix that transforms vectors from $\underrightarrow{\boldsymbol{\mathcal{F}}}_b$ to $\underrightarrow{\boldsymbol{\mathcal{F}}}_a$: $\mathbf{p}_a^{c,b} = \mathbf{C}_{a,b}\mathbf{p}_b^{c,b}$.
$\mathbf{T}_{a,b}$	:	The 4×4 transformation matrix that transforms homogeneous points from $\underrightarrow{\boldsymbol{\mathcal{F}}}_b$ to $\underrightarrow{\boldsymbol{\mathcal{F}}}_a$: $\mathbf{p}_a^{c,a} = \mathbf{T}_{a,b}\mathbf{p}_b^{c,b}$.

Chapter 1

Introduction

> To be sure, a writer cannot begin with a thesis; he must rather use his writerly sensitivity to intuit what is going on, even if he cannot understand its implications.
>
> — Gary Morson, *How the great truth dawned*

1.1 Autonomy and humanity through the ages

Autonomous systems, in some form, have been imagined and realized for the bulk of recorded history. In ancient Greek mythology, the god Hephaestus, the 'patron of invention and technology' (Mayor, 2019) was said to create talking mechanical handmaidens, while early Hindu and Buddhist texts tell of *yantakara* that lived in Greece and created machines that helped in trade and farming. The secret methods of the *yantakara* (the early 'roboticists') were closely guarded, and mechanical assassins were said to pursue and kill any person who revealed their techniques[1] (Mayor, 2019). Fitzgibbon et al. (2007)

Figure 1.1: A 'robot' rebellion from Karel Capek's 1920 play, *Rossum's Universal Robots*.

[1] Please be careful distributing this thesis.

Since the industrial revolution, the idea of an autonomous machine–one that requires no, or very minimal, human intervention or oversight to operate–has been imagined in different ways. Depending on one's perspective, autonomous machines have perennially promised to either usher in a utopia of freedom, or threatened to bring about an age of job loss and social upheaval that worsens socioeconomic divisions. Much like the Luddites of the 19th century, the social critics of the 21st century have continued the dialectic to understand the social ramifications of modern *yantakara* and their newly-created autonomous hand-maidens.

These controversial origins are embedded even within the modern name of for the academic field, *robotics*. The word *robot* comes from the title of a science fiction play, R.U.R.: Rossum's Universal Robots, written by the Czech playwright Karel Capek in 1920 (see Figure 1.2). In naming the play, the word *robot* was derived from the Slavic term for slave, *rab*, and its Czech derivative for serf labour, *rabota*, while the name *Rossum* was inspired by the Czech word for reason, or intellect. Indeed, the concept of enslaved or embodied intelligence is at the heart of modern definitions of the discipline of robotics (Redfield, 2019). Much of the popular culture surrounding robots (e.g., Shelley's *Frankenstein*, Asimov's *I, Robot*, Kubrick's and Clarke's *2001: A Space Odyssey*) also paints a complicated picture of humanity's relationship with such enslaved machines. In this thesis, I focus on improving a specific part of a modern *mobile* autonomy pipeline, while minimizing the use of term *robot* to avoid maelstrom of philosophical and ethical problems that it connotes. I hope my work aids the march of technological progress towards a future which finds some Hegelian synthesis of autonomy and humanity—a future in which human-in-the-loop autonomous systems augment and improve the lot of many people while still negotiating and constantly considering the social costs that come with technological innovation.

1.2 Mobile Autonomy and State Estimation

While the looms and railroads of the industrial revolution were spurred by the discovery of steam engines and electricity, modern *mobile* autonomy was largely born out of the technological arms race of the cold war and the constraints and challenges associated with long-distance flight and extraterrestrial travel (see Grewal and Andrews (2010) for a history of one of the seminal algorithms in mobile autonomy, the Kalman filter). Indeed, much of the work on modern perception algorithms has its origins in the automated compilation of cold-war-era reconnaissance imagery and the design of extraterrestrial rovers like the Mars Exploration Rovers, *Spirit* and *Opportunity* (Scaramuzza and Fraundorfer, 2011). Similarly, much of the planning and control algorithms originate in American and Soviet defence-funded research (Nilsson, 1984; Thrun et al., 2006).

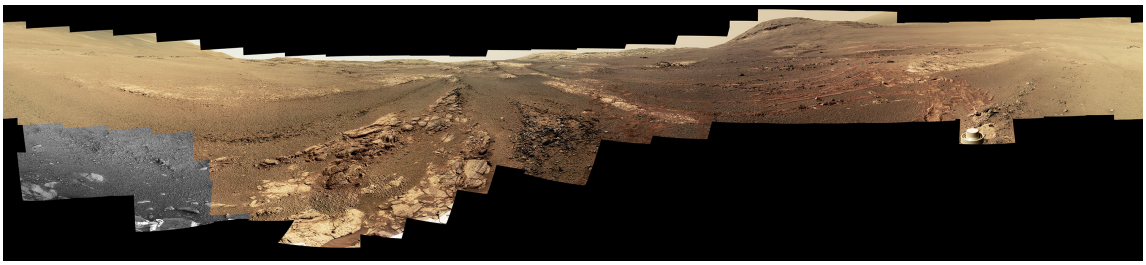
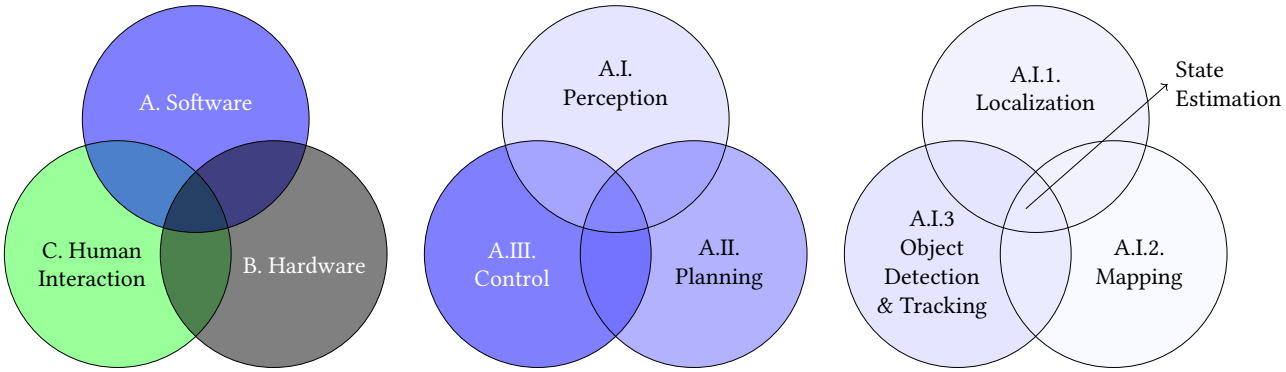


Figure 1.2: The last 360 degree panorama taken by the PanCam apparatus of the Mars Exploration Rover, *Opportunity*, at its final resting place on Mars, the western rim of the Endeavour Crater. Contact with *Opportunity* was lost shortly after this was captured, due to a severe dust storm (credit: NASA/JPL-Caltech/Cornell/ASU).



(a) Research strands in mobile autonomy.

(b) The components of software research.

(c) The components of perception.

Figure 1.3: Venn diagrams of modern mobile autonomy.

Once confined to carefully-controlled factory floors, autonomous mobile platforms have now began to show great promise in improving the safety of human transport, reducing the burden of repetitive, arduous jobs, and more efficiently leveraging limited resources for environmental monitoring. This newly-realized potential can be attributed to several factors: improvements in the cost and efficiency of computing devices (in terms energy efficiency, processing power, and overall size), the availability of relatively cheap, compact, high-quality sensors and rapid prototyping tools, and the development of open-source hardware, software platforms and datasets (e.g., the Robot Operating System, the KITTI Self-Driving Car dataset (Geiger et al., 2013)).

Despite decades of research, mobile autonomy as a field still has nebulous demarcations between subfields. I have attempted to provide a general overview of the field through a series of Venn diagrams in Figure 1.3. At the highest level, the field can be roughly divided into those researchers who study and develop software, those who study and develop hardware, and those who study and analyze the interaction between autonomous systems (composed of both software and hardware) and humans (Figure 1.3a). There is, of course, a plethora

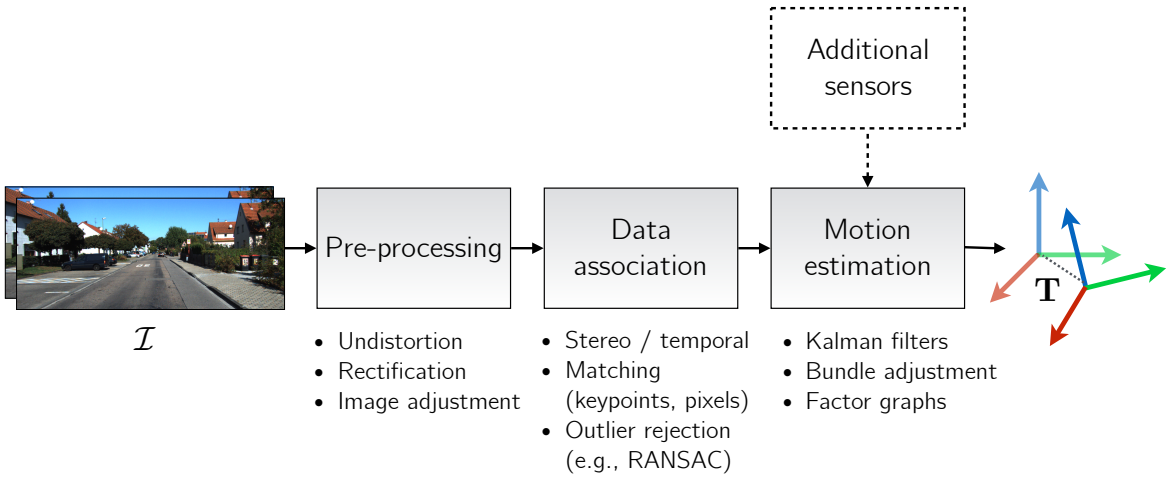


Figure 1.4: A 'classical' visual odometry pipeline consists of several distinct components that have interpretable inputs and outputs.

of overlap between all three of these rough categories. Within the software realm, there has historically been a distinction between those who study algorithms that deal with the perception of the interoceptive and exteroceptive data, those who study how to use that data to plan action, and those who study how to use those plans to control a system to execute that action (Figure 1.3b).

Within perception, which is the focus of this thesis, there are three general directions of research: localization, mapping and object detection and tracking. Localization and mapping can refer to self-localization, or egomotion estimation, which deals with the problem of estimating the pose of a moving platform through an unknown world, SLAM, simultaneous localization and mapping, which deals with the former problem and mapping simultaneously and finally, it can refer to localization within a known map given some hitherto unseen observation of that environment. The field of object detection and tracking (whether that be static objects like stop signs, or moving objects like humans, animals or vehicles) uses much of the same underlying mathematics as the former two, but has historically been a separate strand of research. Broadly, the overlap of all three of these pursuits within the field of autonomy and robotics is referred to as *state estimation* (Barfoot, 2017).

1.3 The *State* of State Estimation

Central to *classical* state estimation algorithms (which, in this context, refers to the bulk of state estimation research published prior to 2016) is the idea of a pipeline. A pipeline consists several distinguishable blocks that have interpretable inputs and outputs. By carefully pro-

cessing information contained within sensor data, pipelines facilitate the construction of complex state estimation architectures that can fuse observations from sensors of varied modality to create rich models of the external world and infer the state of a mobile platform within it. My thesis focuses on egomotion estimation: the problem of accurately and consistently estimating the relative pose of a moving platform. For this task, a variety of different sensors may be useful (e.g., lidar, stereo cameras, or inertial measurement units), and each may allow for various components of a state estimation pipeline. For cameras, egomotion estimation is typically referred to as *visual odometry* or VO for short. A typical VO pipeline is illustrated in Figure 3.1.

Modern visual egomotion estimation pipelines (e.g., Leutenegger et al. (2015); Cvišić and Petrović (2015); Tsotsos et al. (2015)) have achieved impressive localization accuracy on trajectories spanning several kilometres by carefully extracting and tracking sparse visual features (using *hand-crafted* algorithms) across consecutive images. Simultaneously, significant effort has gone to developing localization pipelines that eschew sparse features in favour of *dense* visual data (Alcantarilla and Woodford, 2016; Forster et al., 2014), typically relying on loss functions that use direct pixel intensities. There are also those that fuse the two (Forster et al., 2014) modalities into a semi-direct approach.

In the last five years, a significant part of the state estimation literature has also focused on the idea of replacing classical pipelines with parametric modelling through deep convolutional neural networks (CNNs) and data-driven training. Although initially developed for image classification (LeCun et al., 2015), CNN-based measurement models have been applied to numerous problems in geometric state estimation (e.g., homography estimation (DeTone et al., 2016), single image depth reconstruction (Garg et al., 2016), camera re-localization (Kendall and Cipolla, 2016), place recognition (Sünderhauf et al., 2015)). A number of recent CNN-based approaches have also tackled the problem of egomotion estimation, often purporting to obviate the need for classical visual localization pipelines by learning pose changes *end-to-end*, directly from image data (e.g., Melekhov et al. (2017), Handa et al. (2016), Oliveira et al. (2017)).

Despite this surge of excitement, significant debate has emerged within the robotics and computer vision communities regarding the extent to which deep models should replace existing geometric state estimation algorithms. Owing to their representational power, deep models may move the onerous task of selecting 'good' (i.e., robust to environmental vagaries and sensor motion) visual features from the roboticist to the learned model. By design, deep models also provide a straight-forward formulation for using *dense* data while being flexible in their loss function, and taking full advantage of modern computing architecture to minimize run time. Despite these potential benefits, current deep regression techniques for state

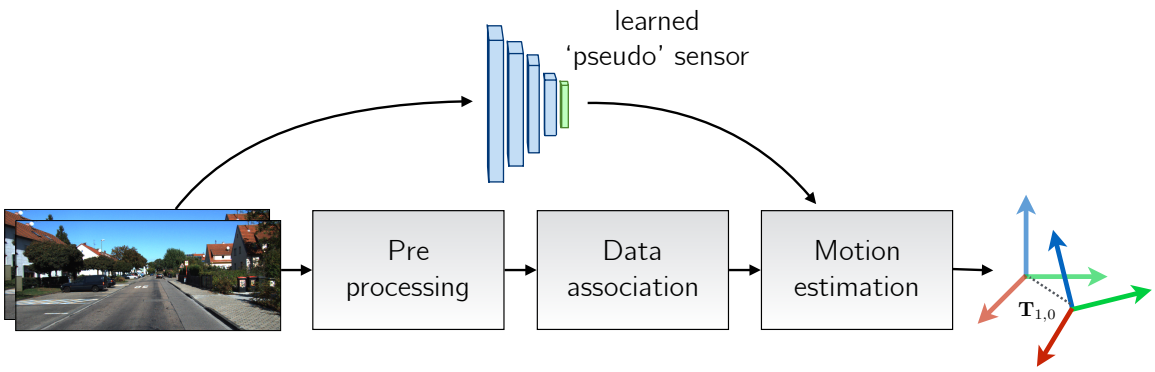


Figure 1.5: A learned *pseudo-sensor* extracts latent information from the same data stream.

estimation often generalize poorly to new environments, come with few analytical guarantees, and provide only point estimates of latent parameters. Furthermore , there is new evidence that separating different tasks into interpretable components (e.g., optical flow, scene segmentation) works better than end-to-end learning for action (Zhou et al., 2019). Table 1.1 summarizes the benefits of using deep models (as opposed to classical pipelines) along several salient criteria.

Table 1.1: The benefits and downsides to using learning.

	Classical Pipelines	**Deep Models**
Maturity	Decades of literature & domain knowledge	Nascent in autonomy
Interpretability	Good, each component has interpretable input and output	Poor, often with no interpretable intermediate outputs
Uncertainty	Foundational to probabilistic robotics	Some methods (bootstrap, BCNNs)
Robustness	Relatively good	Highly dependant on training data
Flexibility	Limited by components	Limited by training data

1.4 The Learned Pseudo-Sensor

To retain the benefits of classical state estimation pipelines while leveraging the representational power of modern data-drive learning techniques, we introduce the paradigm of the

learned *pseudo-sensor*. Instead of completely replacing the classical pipeline, this thesis presents four general ways in which machine learning can be used to train a hyper-parametric model that extracts latent information from an existing visual data stream. By fusing the output of these sensors with the output of the pipeline, we can make the final egomotion estimates more accurate and more robust to difficult-to-model effects (Figure 1.5). In this thesis, we present two different ways in which this fusion can be performed. The first, which is used in Predictive Robust Estimation (PROBE, and its follow up PROBE-GK, **??**), is to use it to learn a heteroscedastic noise model that can be incorporated into a maximum-likelihood loss function which can be minimized traditional non-linear optimization techniques. The second (used by Sun-BCNN, DPC, and HydraNet, Chapter 2 and **????**) produce geometric quantities (probabilistic estimates of an illumination source, $\mathrm{SE}(3)$ corrections to existing egomotion estimates, and independent probabilistic rotation estimates, respectively), that can be fused with the original pipeline through a general factor graph framework.

1.5 Original Contributions

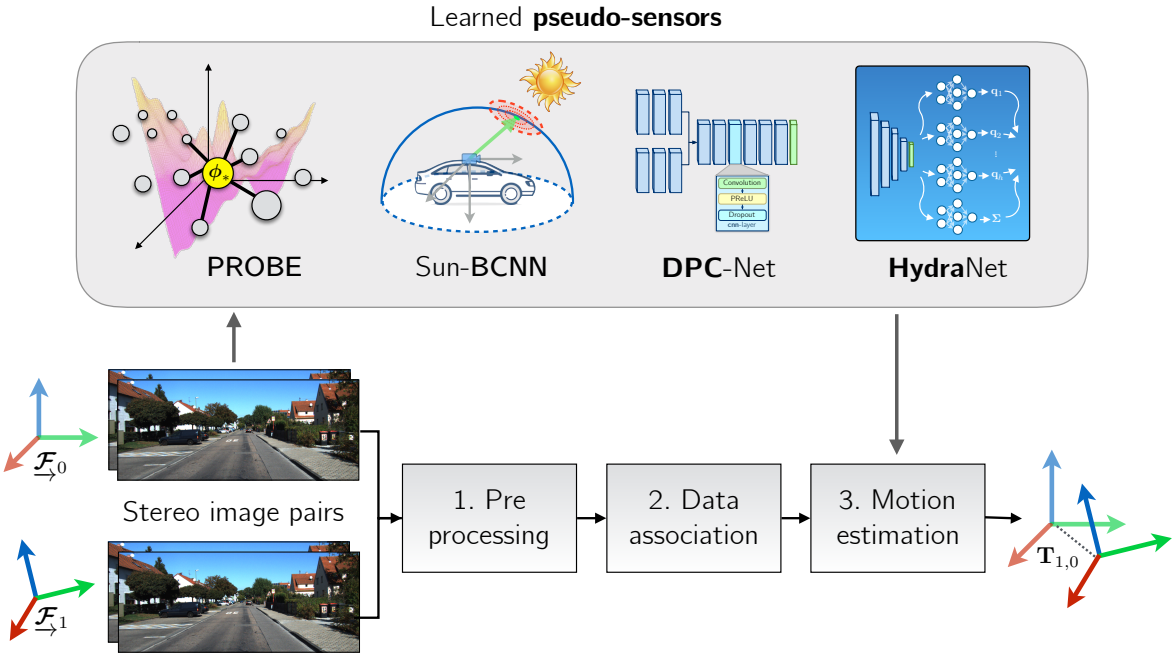


Figure 1.6: My thesis presents four different types of pseudo-sensors to improve egomotion estimation.

My thesis presents four different types of pseudo-sensors to improve visual odometry: PROBE, Sun-BCNN, DPC-Net, and $\mathrm{SO}(3)$ HydraNet:

1. **Predictive Robust Estimation (PROBE)**,
   Predictive Robust Estimation (**??**) uses k-NN regression (original PROBE) or Generalized Kernels (Vega-Brown et al., 2014) (PROBE-GK) to train a predictive model for heteroscedastic measurement covariance of stereo reprojection errors to improve the accuracy and consistency of an indirect stereo visual odometry pipeline. It is associated with two publications:

   - Peretroukhin, V., Clement, L., Giamou, M., and Kelly, J. (2015a). PROBE: Predictive robust estimation for visual-inertial navigation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'15)*, pages 3668–3675, Hamburg, Germany
   - Peretroukhin, V., Vega-Brown, W., Roy, N., and Kelly, J. (2016a). PROBE-GK: Predictive robust estimation using generalized kernels. In *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, pages 817–824

2. **Virtual Sun Sensor using a Bayesian Convolutional Neural Network**
   Sun-BCNN (Chapter 2) is a technique to infer a probabilistic estimate of the direction of the sun from a single RGB image using a Bayesian Convolutional Neural Networks (BCNN). The method works much like dedicated sun sensors (Lambert et al., 2012), but requires no additional hardware, and can provide mean and covariance estimates that can be readily incorporated into existing visual odometry frameworks. Initial exploratory work was published at ISER 2016, and the BCNN improvement was presented at ICRA 2017. An additional journal paper summarizing the work of the prior two papers, adding data from the Canadian High Arctic and Oxford, and investigating the effect of cloud cover and transfer learning was published in the International Journal of Robotics' Research, Special Issue on Experimental Robotics at the end of 2017.

   - Clement, L., Peretroukhin, V., and Kelly, J. (2017). Improving the accuracy of stereo visual odometry using visual illumination estimation. In Kulic, D., Nakamura, Y., Khatib, O., and Venture, G., editors, *2016 International Symposium on Experimental Robotics*, volume 1 of *Springer Proceedings in Advanced Robotics*, pages 409–419. Springer International Publishing, Berlin Heidelberg. Invited to Journal Special Issue
   - Peretroukhin, V., Clement, L., and Kelly, J. (2017a). Reducing drift in visual odometry by inferring sun direction using a bayesian convolutional neural network. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'17)*, pages 2035–2042, Singapore

- Peretroukhin, V., Clement, L., and Kelly, J. (2018). Inferring sun direction to improve visual odometry: A deep learning approach. *International Journal of Robotics Research*

3. **Deep Pose Corrections (DPC-Net)**

   Deep Pose Correction (**??**) is an approach to improving egomotion estimates through pose corrections learned through deep regression. DPC takes as its starting point an efficient, classical localization algorithm that computes high-rate pose estimates. To it, it adds a Deep Pose Correction Network (DPC-Net) that learns low-rate, 'small' *corrections* from training data that are then fused with the original estimates. DPC-Net does not require any modification to an existing localization pipeline, and can learn to correct multi-faceted errors from estimator bias, sensor mis-calibration or environmental effects. It is associated with a journal publication:

   - Peretroukhin, V. and Kelly, J. (2018). DPC-Net: Deep pose correction for visual localization. *IEEE Robotics and Automation Letters*.

4. **Estimating Rotation through Deep Probabilistic Inference with HydraNet**

   Finally, HydraNet (**??**) is a multi-headed network structure that can regress probabilistic estimates of rotation (elements of the matrix Lie group, $\mathrm{SO}(3)$ accounting for both aleatoric and epistemic uncertainty. This uncertainty can then be used to fuse the output of HydraNet with the output of classical egomotion pipelines in a probabilistic factor graph formulation.

   - Peretroukhin, V., Wagstaff, B., and Kelly, J. (2019). Deep probabilistic regression of elements of so(3) using quaternion averaging and uncertainty injection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'19) Workshop on Uncertainty and Robustness in Deep Visual Learning*, pages 83–86, Long Beach, California, USA

Chapter 2

Sun-BCNN

2.1 Motivation

A crucial competency of any autonomous mobile robot is the ability to estimate its own motion through an operating environment. While there exists a rich body of literature on the topic of motion estimation using a variety of techniques such as lidar-based point cloud matching (**?**) and visual-inertial odometry (Leutenegger et al., 2015), egomotion estimation is fundamentally a process of dead-reckoning and will accumulate unbounded error over time. This accumulated error, or drift, can be limited by incorporating global information into the motion estimation problem. This frequently takes the form of a globally consistent map, loop closure detection, or reliance on additional sensors such as GPS to make corrections to the estimated trajectory. In many situations, however, a globally consistent map may be unavailable or prohibitively expensive to compute, loop closures may not occur, or GPS may be unavailable or inaccurate. In such cases, it can be advantageous to rely on environmental cues such as the sun, which can easily provide global orientation information since it is readily detectable and its apparent motion in the sky is well described by ephemeris models.

For visual odometry (VO) in particular, the addition of global orientation information can limit the growth of drift error to be linear rather than superlinear with distance traveled (**?**). Sun-based orientation corrections have been successfully used in planetary analogue environments (**??**) as well as on board the Mars Exploration Rovers (MERs) (**??**). In particular, **?** showed that incorporating sun sensor and inclinometer measurements directly into the motion estimation pipeline (as opposed to periodically updating the vehicle heading, as in earlier work) can significantly reduce VO drift over long trajectories.

In this work, we seek to answer the question of whether similar reductions in VO drift can be obtained solely from the image stream already being used to compute VO. The main idea here is that by reasoning over more than just the geometric information available from a

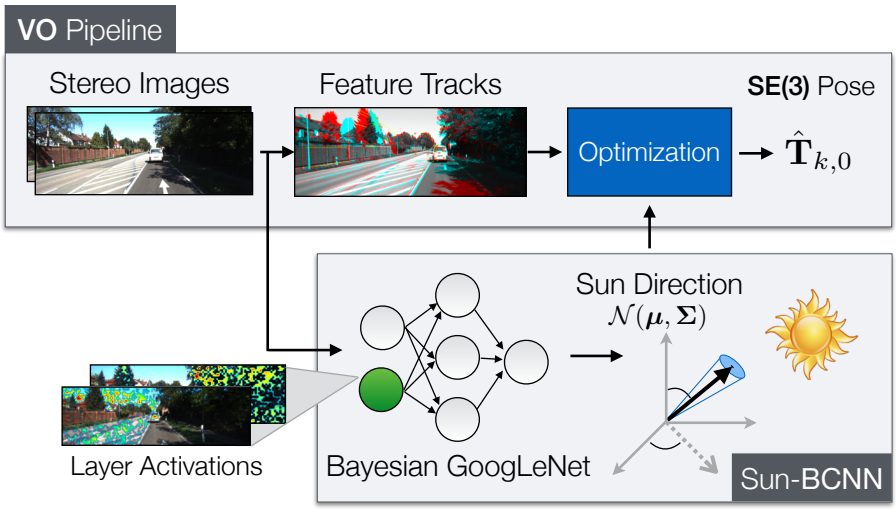


Figure 2.1: Our method uses a Bayesian Convolutional Neural Network (BCNN) to estimate the direction of the sun and to produce a principled uncertainty estimate for each prediction. We incorporate this *virtual sun sensor* into a stereo visual odometry pipeline to reduce estimation error.

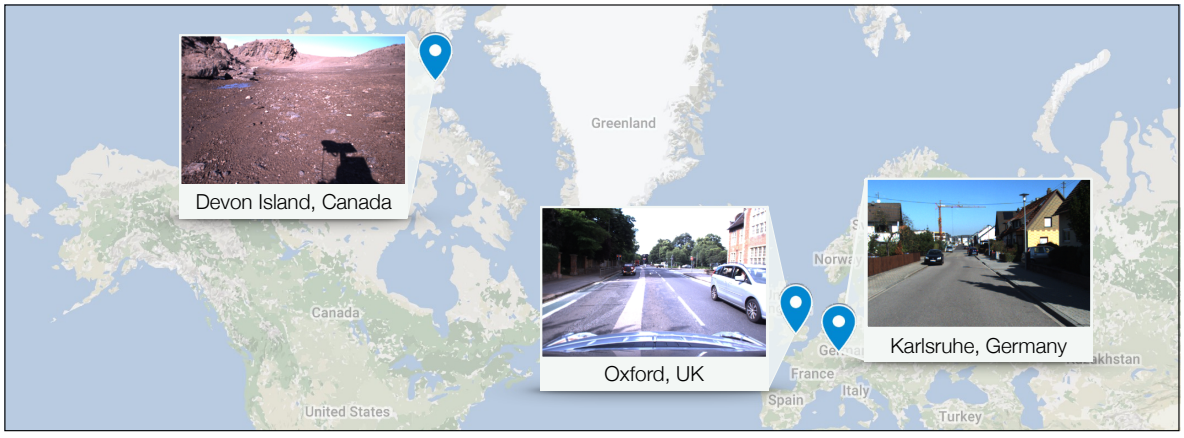


Figure 2.2: We train and test Sun-BCNN in a variety of environments ranging from urban driving in Europe to remote planetary analogue sites in the Canadian High Arctic. (Map data: Google, INEGI, ORION-ME.)

standard RGB camera, we can improve existing VO techniques without needing to rely on a dedicated sun sensor or specially oriented camera. In particular, we leverage recent advances in Bayesian Convolutional Neural Networks (BCNNs) to demonstrate how we can build and train a deep model capable of inferring the direction of the sun from a single RGB image. Moreover, we show that our network, dubbed Sun-BCNN, can produce a covariance estimate for each observation that obviates the need for a hand-tuned or empirically computed static covariance typically used for data fusion in a motion estimation pipeline.

Our main contributions can be summarized as follows:

1. We apply a Bayesian CNN to the problem of sun direction estimation, incorporating the resulting covariance estimates into a visual odometry pipeline;

2. We demonstrate that a Bayesian CNN with dropout layers after each convolutional and fully-connected layer can achieve state-of-the-art accuracy at test time;

3. We learn a 3D unit-length sun direction vector, appropriate for full 6-DOF pose estimation;

4. We present experimental results on over 30 km of visual navigation data in urban (**?**) and planetary analogue (**?**) environments;

5. We investigate the sensitivity of our Bayesian CNN-based sun estimator (Sun-BCNN) to cloud cover, camera and environment changes, and measurement parameterization; and

6. We release Sun-BCNN as open-source [1].

The remainder of this paper begins with a discussion of related work, followed by an overview of the theory underlying BCNNs and a discussion of our model architecture, implementation, and training procedure. We then outline our chosen visual odometry pipeline, which is based on a two-frame bundle adjustment optimization, and describe how observations of the sun can be incorporated directly into the motion estimation problem following the technique of **?**. Finally, we present several sets of experiments designed to test and validate both Sun-BCNN and our sun-aided VO pipeline in variety of environments. These include experiments on 21.6 km of urban driving data from the KITTI odometry benchmark training set (**?**), as well as a further 10 km traverse through a planetary analogue site taken from the Devon Island Rover Navigation Dataset collected in a planetary analogue site in the Canadian High Arctic (**?**). We investigate the possibility of model generalization between different cameras

[1] https://github.com/utiasSTARS/sun-bcnn-vo.

and environments, and further explore the sensitivity of Sun-BCNN to cloud cover during training and testing, using data from the Oxford Robotcar Dataset (**?**). We also examine the impact of different methods for computing the mean and covariance of a norm-constrained vector on the accuracy and consistency of the estimated sun directions.

2.2 Related Work

Visual odometry (VO), a technique to estimate the motion of a moving platform equipped with one or more cameras, has a rich history of research including a notable implementation onboard the Mars Exploration Rovers (MERs) (Scaramuzza and Fraundorfer, 2011). Modern approaches to VO can achieve estimation errors below 1% of total distance traveled (Geiger et al., 2013). To achieve such accurate and robust estimates, modern techniques use careful visual feature pruning (Cvišić and Petrović, 2015), adaptive robust methods (**?**Peretroukhin et al., 2016a), or operate directly on pixel intensities (**?**).

Independent of the estimator, VO exhibits superlinear error growth, and is particularly sensitive to errors in orientation (**?**Cvišić and Petrović, 2015). One way to reduce orientation error is to incorporate observations of a landmark whose position or direction in the navigation frame is known *a priori*. The sun is an example of such a known directional landmark. Accordingly, hardware sun sensors have been used to improve the accuracy of VO in planetary analogue environments (e.g., the Sinclair Interplanetary SS-411 sun sensor used by **?** and **?**), while the MERs articulated their Pancam apparatus to directly image the sun (**??**). More recently, software-based alternatives have been developed that can estimate the direction of the sun from a single image, making sun-aided navigation possible without additional sensors or a specially-oriented camera (**?**). Some of these methods have been based on hand-crafted illumination cues such as shadows and variation in sky brightness (**??**), while others have attempted to learn such cues from data using deep Convolutional Neural Networks (CNNs) (Ma et al., 2016).

Convolutional Neural Networks (CNNs) have been applied to a wide range of classification, segmentation, and learning tasks in computer vision (LeCun et al., 2015). Recent work has shown that CNNs can learn orientation information directly from images by modifying the loss functions of existing discrete classification-based CNN architectures into continuous regression losses (Ma et al., 2016; **?**; Kendall and Cipolla, 2016). Despite their success in improving prediction accuracy, most existing CNN-based models do not report uncertainty estimates, which are important in the context of data fusion.

For classification, it is possible to restrict CNN model outputs to a certain range (e.g., using a softmax function) and interpret these values as the model's confidence in its output. As **?**

noted, however, this can be misleading because these values can be unjustifiably large for test points far away from training data. To address this, **?** showed that it is possible to achieve covariance outputs that better quantify model uncertainty for classification and regression tasks, with only minor modifications to existing CNN architectures. An early application of this uncertainty quantification was presented by Kendall and Cipolla (2016) who used it to improve their prior work (**?**) on camera pose regression.

We build on on previous work by **?**, who demonstrated empirically that techniques for single-image sun estimation based on hand-crafted models (**?**) and Convolutional Neural Networks (CNNs) (Ma et al., 2016) could be incorporated into a stereo visual odometry pipeline to reduce estimation error in the manner of **?**. We also build on the work of Peretroukhin et al. (2017b), who presented preliminary experimental results comparing Sun-BCNN against the method of **?** and its VO-informed variant (**?**) as well as the Sun-CNN of Ma et al. (2016) on the KITTI odometry benchmark (**?**Geiger et al., 2013), both in terms of raw measurement accuracy and in terms of their impact on VO accuracy.

While our method is similar in spirit to the work of Ma et al. (2016), who built a CNN-based sun sensor as part of a relocalization pipeline, our model makes three important improvements: 1) in addition to a point estimate of the sun direction, we output a principled covariance estimate that is incorporated into our estimator; 2) we produce a full 3D sun direction estimate with azimuth and zenith angles that is better suited to 6-DOF robot pose estimation problems (as opposed to only the azimuth angle and 3-DOF estimator used by Ma et al. (2016)); and 3) we incorporate the sun direction covariance into a VO estimator that accounts for growth in pose uncertainty over time (unlike **?**). Furthermore, our Bayesian CNN includes a dropout layer after every convolutional and fully connected layer (as outlined by **?** but not done by Kendall and Cipolla (2016)), which produces more principled covariance outputs.

2.3 Indirect Sun Detection using a Bayesian Convolutional Neural Network

We use a Bayesian Convolutional Neural Network (BCNN) to infer the direction of the sun and an associated uncertainty, and refer to our model as Sun-BCNN. We motivate the choice of a deep model through the empirical findings of **?** and Ma et al. (2016), who demonstrated that a CNN-based sun detector can substantially outperform hand-crafted models such as that of **?** both in terms of measurement accuracy and in its application to a VO task.

We choose a deep neural network structure based on GoogLeNet (**?**) due to its use in

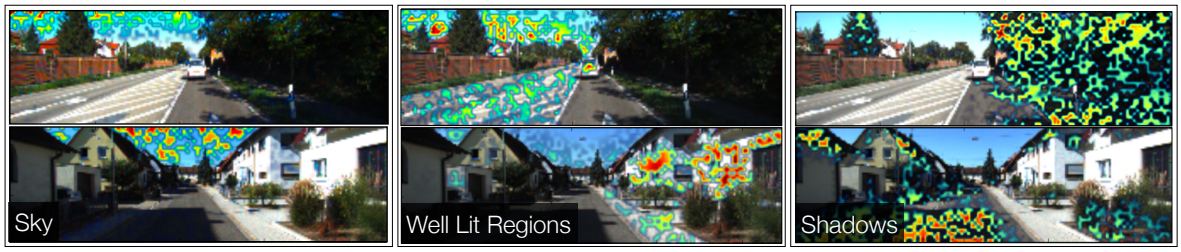


Figure 2.3: Three conv1 layer activation maps superimposed on two images from the KITTI odometry benchmark (?) 00 and 04 for three selected filters. Each filter picks out salient parts of the image that aid in sun direction inference.

past work that adapted it for orientation regression (Kendall and Cipolla, 2016; ?). Unlike Ma et al. (2016), we choose to transfer weights trained on the MIT Places dataset (?) rather than ImageNet (?). We believe the MIT Places dataset is a more appropriate starting point for localization tasks than ImageNet since it includes outdoor scenes and is concerned with classifying physical locations rather than objects.

2.3.1 Cost Function

We train Sun-BCNN by minimizing the cosine distance between the unit-norm target sun direction vector $\mathbf{s}_k$ and the predicted unit-norm sun direction vector $\hat{\mathbf{s}}_k$, where k indexes the images in the training set:

$$\mathcal{L}\left(\hat{\mathbf{s}}_k\right) = 1 - \left(\hat{\mathbf{s}}_k \cdot \mathbf{s}_k\right). \tag{2.1}$$

Note that in our implementation, we do not formulate the cosine distance loss explicitly, but instead minimize half the square of the tip-to-tip Euclidian distance between $\mathbf{s}_k$ and $\hat{\mathbf{s}}_k$, which is equivalent to Equation (2.1) since both vectors have unit length:

$$\begin{aligned}\frac{1}{2}\left\|\hat{\mathbf{s}}_k - \mathbf{s}_k\right\|^2 &= \frac{1}{2}\left(\left\|\hat{\mathbf{s}}_k\right\|^2 + \left\|\mathbf{s}_k\right\|^2 - 2(\hat{\mathbf{s}}_k \cdot \mathbf{s}_k)\right) \\ &= 1 - \left(\hat{\mathbf{s}}_k \cdot \mathbf{s}_k\right) \\ &= \mathcal{L}\left(\hat{\mathbf{s}}_k\right).\end{aligned}$$

We ensure that our network output, $\hat{\mathbf{s}}_k$, has a unit norm by appending a normalization layer to the network.

2.3.2 Uncertainty Estimation

Following recent work on Bayesian Convolutional Neural Networks (BCNNs) (???), we modify our model architecture to enable the computation of principled covariance estimates as-

sociated with each predicted sun direction. To achieve computationally tractable Bayesian inference with a CNN architecture, BCNNs exploit a connection between stochastic regularization (e.g., dropout, a widely used technique in deep learning to mitigate overfitting) and approximate variational inference of a Bayesian Neural Network. We outline the technique here briefly, and refer the reader to **?** for more details.

The method begins with a prior $p(\boldsymbol{w})$ on the weights in a deep neural network and attempts to compute a posterior distribution $p(\boldsymbol{w}|\mathbf{X}, \mathbf{S})$ given training inputs $\mathbf{X} = \{\mathbf{x}_k\}$ and targets $\mathbf{S} = \{\mathbf{s}_k\}$. This posterior can be used to compute a predictive distribution for test samples but is generally intractable. To overcome this, the BCNN approach notes that CNN training with stochastic regularization can be viewed as variational inference if we define a variational distribution $q(\boldsymbol{w})$ as:

$$q(\boldsymbol{w}_i) = \mathbf{M}_i \text{diag}\left(\{b_j^i\}_{j=1}^{K_i}\right), \tag{2.2}$$

$$b_j^i \in \text{Bernoulli}(p_i). \tag{2.3}$$

Here, i indexes a particular layer in the neural network with K_i weights, $\mathbf{M}$ are the weights to be optimized, b_j^i are Bernoulli distributed binary variables, and p_i is the dropout probability for weights in layer i.

With this variational distribution $q(\boldsymbol{w})$, training a CNN with dropout is analogous to minimizing $\text{KL}(p(\boldsymbol{w}|\mathbf{X}, \mathbf{S}) \,||\, q(\boldsymbol{w}))$, the Kullback-Leibler (KL) divergence between the variational distribution and the true posterior. At test time, the first two moments of the predictive distribution are approximated using Monte Carlo integration over the weights $\boldsymbol{w}$:

$$\mathbb{E}\left[\hat{\mathbf{s}}^*\right]_k = \hat{\bar{\mathbf{s}}}_k^* \approx \frac{1}{N}\sum_{n=1}^{N} \hat{\mathbf{s}}_k^*(\mathbf{x}_k^*, \boldsymbol{w}^n) \tag{2.4}$$

$$\mathbb{E}\left[\hat{\mathbf{s}}_k^* \; \hat{\mathbf{s}}^{*\,T}_k\right] \approx \tau^{-1}\mathbf{1} + \frac{1}{N}\sum_{n=1}^{N} \hat{\mathbf{s}}_k^*(\mathbf{x}_k^*, \boldsymbol{w}^n) \; \hat{\mathbf{s}}_k^*(\mathbf{x}_k^*, \boldsymbol{w}^n)^T - \hat{\bar{\mathbf{s}}}_k^* \; \hat{\bar{\mathbf{s}}}_k^{*\,T}, \tag{2.5}$$

where $\mathbf{1}$ is the identity matrix, and $\boldsymbol{w}^n$ is a sample from $q(\boldsymbol{w})$ (obtained by sampling the network with dropout). The model precision, τ, is computed as

$$\tau = \frac{pl^2}{2M\lambda}, \tag{2.6}$$

where p is the dropout probability, l is the characteristic length scale, M is the number of samples in the training data, and λ is the weight decay.

Following **?**, we build our BCNN by adding dropout layers after every convolutional and fully connected layer in the network. We then retain these layers at test time to sample the network stochastically, following the technique of Monte Carlo Dropout, and obtain the relevant statistical quantities using Equations (2.4) and (2.5).

2.3.3 Implementation and Training

We implement our network in Caffe (**?**), using the `L2Norm` layer from the Caffe-SL fork[2] to enforce a unit-norm constraint on the final output. We train the network using stochastic gradient descent, setting all dropout probabilities to 0.5, performing 30,000 iterations with a batch size of 64, and setting the initial learning rate to be between 10^{-3} and 10^{-4}. Training requires approximately 2.5 hours on an NVIDIA Titan X GPU. Interestingly, Figure 2.3 shows that some convolutional filters learned by Sun-BCNN on the KITTI dataset appear to correspond to illumination variations reminiscent of the visual cues designed by **?**.

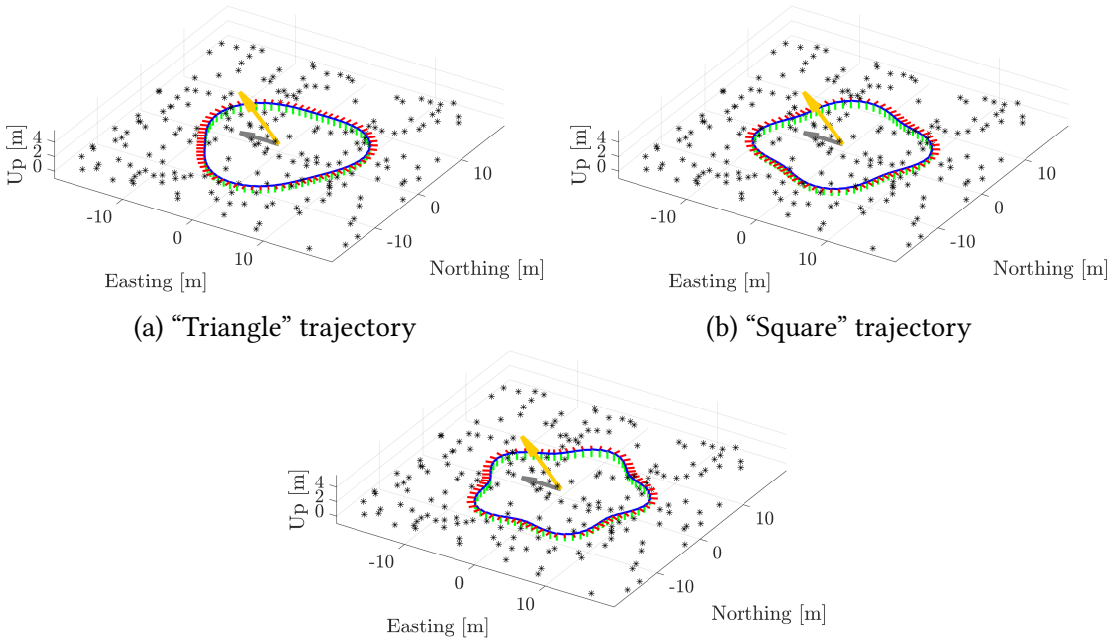
Data Preparation & Transfer Learning

We resize images from their original size to $[224 \times 224]$ pixels to achieve the image size expected by GoogleLeNet. We experimented with preserving the aspect ratio of the original image and padding zeros to the top and bottom of the resized image, but found that preserving the vertical resolution (as done by Ma et al. (2016)) results in better test-time accuracy. We do not crop or rotate the images, nor do we augment the dataset in any other way.

Model Precision

We find an empirically optimal model precision τ (see Equation (2.6)) by optimizing the Average Normalized Estimation Error Squared (ANEES) across the entire test set for each dataset. While this hyperparameter should in principle be tuned using a validation set, we omit this step to keep our training procedure consistent with that of Ma et al. (2016). We note that the BCNN uncertainty estimates are affected by two significant factors: 1) variational inference is known to underestimate predictive variance (**?**); and 2) we assume the observation noise is homoscedastic. As noted by **?**, the BCNN can be made heteroscedastic by learning the model precision during training, but this extension is outside the scope of this work.

[2] https://github.com/wanji/caffe-sl



(a) "Triangle" trajectory

(b) "Square" trajectory

(c) "Star" trajectory

Figure 2.4: One loop of the "Triangle", "Square", and "Star" trajectories, consisting primarily of translation and yaw rotation. Landmarks are shown as black asterisks, and the simulated sun direction is indicated with a yellow arrow along with its projection, in grey, on the EN-plane.

Data Partitioning

We partition our data into training and testing sets using a leave-one-out approach based on temporally disjoint sequences of images. That is, given N sequences, the model tested on sequence i is trained with sequences $\{1, 2, ..., N\} \setminus i$. This process varies based on the dataset, and we discuss the specifics in the experimental discussion corresponding to each. In contrast to randomly holding out a subset of the data, this method minimizes the similarity of training and testing data for temporally correlated image streams.

2.4 Simulation Experiments

We assess the benefit of incorporating sun observations of varying quality by conducting a series of simulation experiments consisting of a stereo camera moving along loopy trajectories of varying shapes through a simulated field of point landmarks, with a single static directional landmark representing the sun. Figure 2.4 shows several such loopy trajectories.

We simulate the sun at $45°$ of zenith and an arbitrary azimuth angle, and corrupt observations of the ground truth sun vector with artificial noise such that the mean angular distance (a non-negative quantity) between the observed and true sun direction is $0°$, $10°$, $20°$, and

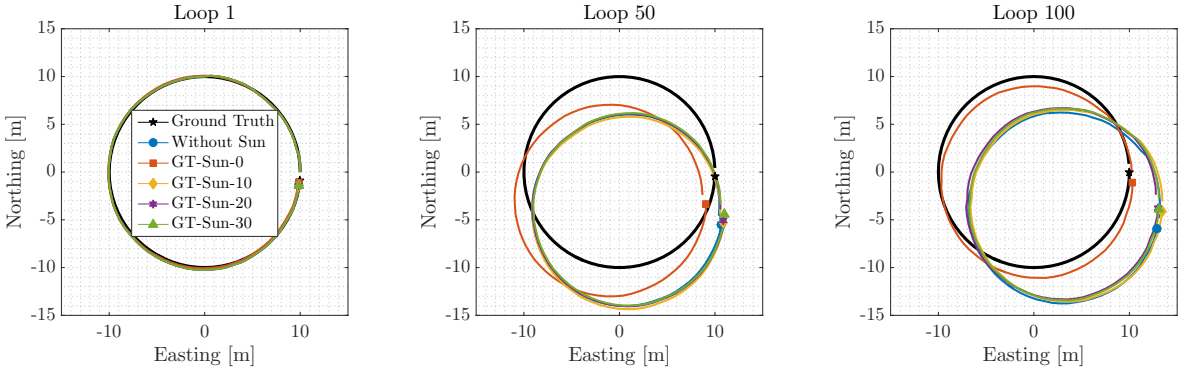
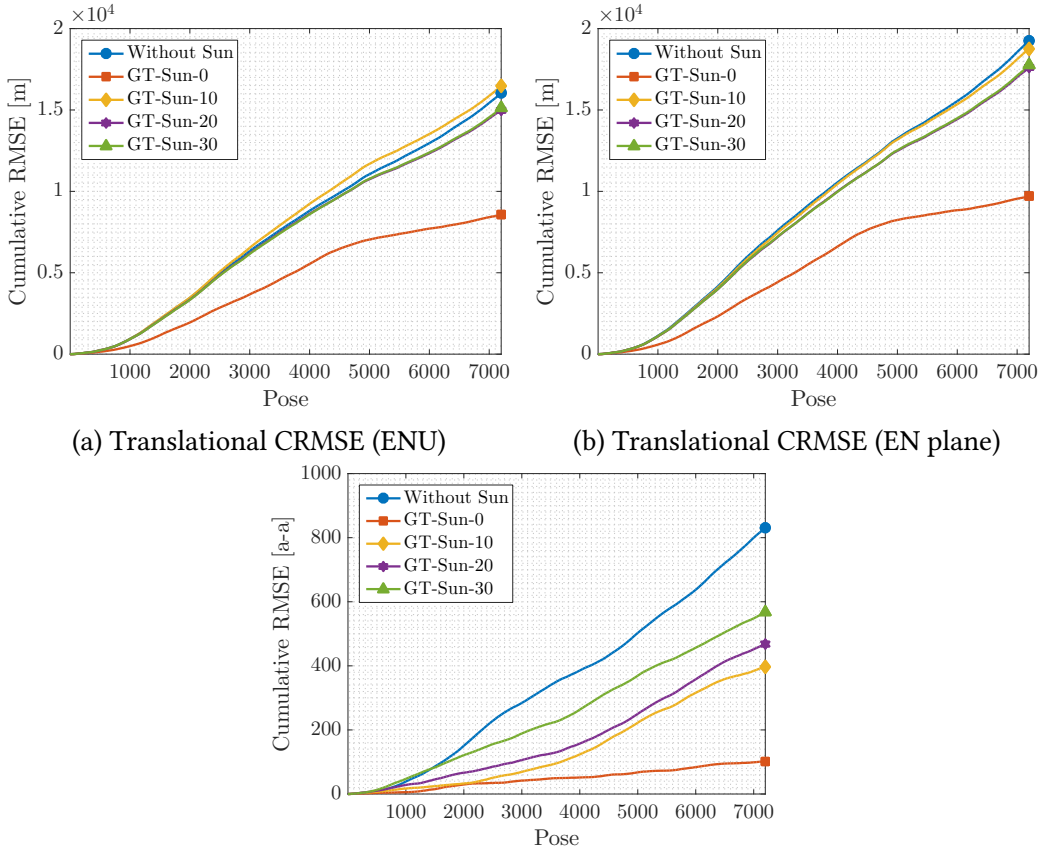


Figure 2.5: Selected segments of a 100-loop "Circle" trajectory, without sun corrections, and with sun corrections corrupted by varying levels of artificial Gaussian noise. The effect of VO drift can be clearly seen, as well as the benefit of incorporating observations of a directional landmark such as the sun.



(a) Translational CRMSE (ENU)

(b) Translational CRMSE (EN plane)

(c) Rotational CRMSE

Figure 2.6: Cumulative root mean squared error (CRMSE) of a simulated 100-loop circular trajectory, without sun corrections, and with sun corrections corrupted by varying levels of artificial Gaussian noise. The accumulated estimation error is greatly reduced by incorporating observations of the sun, and the benefit decreases as these observations become noisier.

30°, labeling these conditions *GT-Sun-0*, *GT-Sun-10*, *GT-Sun-20*, and *GT-Sun-30*, respectively. In our experiments, we treated the measurement noise as an additive quantity sampled from a zero-mean isotropic 3D Gaussian distribution, and renormalized the resulting vectors to enforce the unit-norm constraint.

We simulate the sun at 45° of zenith and an arbitrary azimuth angle, and corrupt observations of the ground truth sun vector with artificial noise such that the mean angular distance (a non-negative quantity computed from the dot product) between the ground truth and noisy sun vectors is 0°, 10°, 20°, or 30°. We label these conditions *GT-Sun-0*, *GT-Sun-10*, *GT-Sun-20*, and *GT-Sun-30*, respectively. We generated these noisy measurements by first sampling 3-vectors from an isotropic zero-mean multivariate Gaussian distribution, then adding these vectors to the ground truth sun vector, and finally normalizing the result to unit length. We chose the covariance of this distribution to yield the desired average angular distance in each case. Note that although the distribution from which we sample noise vectors is zero-mean, the average angular distances will not be zero-mean because angular distance is non-negative.

Our choice to add noise in $\mathbb{R}^3$ and re-normalize was motivated by the fact that this process yields approximately Gaussian error distributions over the azimuth and zenith error angles, which is an important property assumed by our VO pipeline to produce maximum likelihood motion estimates based on the fusion of multiple data sources. We note that these distributions are less Gaussian-like for larger covariances (due to the geometry of the unit 2-sphere) and for ground truth vectors near singularities (e.g., zero zenith).

We also experimented with sampling simulated measurements from a Von Mises-Fisher distribution (**?**), which is approximately analogous to an isotropic Gaussian distribution that respects the geodesics on the unit 2-sphere. However, we observed that the resulting distributions on azimuth and zenith error were severely non-Gaussian, which violated the assumption of zero-mean Gaussian noise in our VO pipeline and interfered with our VO experiments.

Since our VO pipeline does not incorporate loop closures, the effects of drift in the VO solution can be clearly seen by examining individual loops in the camera trajectory. Figure 2.5 shows three loops from the "Circle" trajectory, demonstrating that the VO solution drifts significantly from the true trajectory by the 100th loop. Figure 2.6 plots the translational and rotational cumulative root mean squared error (CRMSE) for this trajectory, which measures the growth in total estimation error over time. Figure 2.6c in particular highlights the significant effect of sun sensing on rotational error, where we see a clear progression in estimation error as the sun direction observations become more noisy.

?? shows that while all four simulation trajectories display consistent and predictable reductions in rotational average root mean squared error (ARMSE), this is not always the case for translational ARMSE. This is because translational errors are only partially induced by ro-

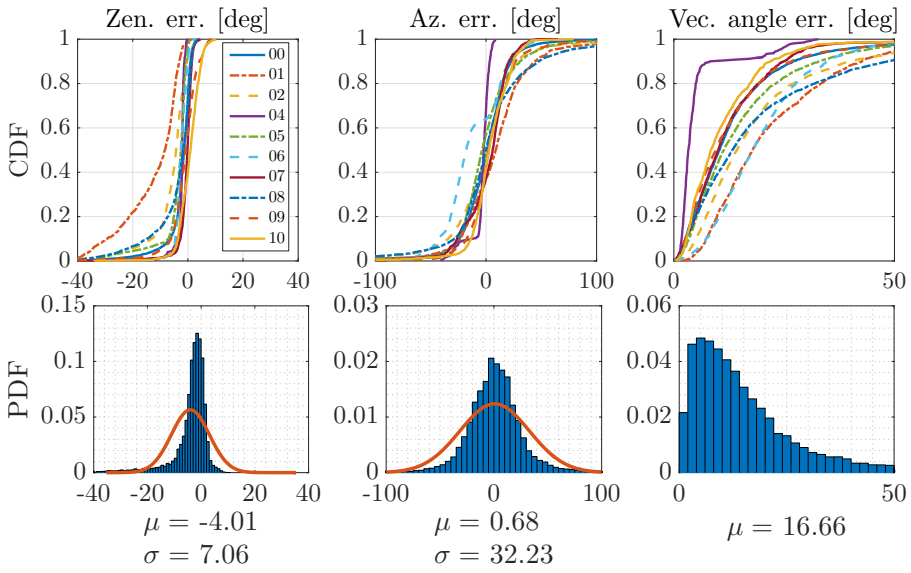


Figure 2.7: Distributions of azimuth error, zenith error, and angular distance for Sun-BCNN compared to ground truth over each test sequence in the KITTI dataset. *Top row*: Cumulative distributions of errors for each test sequence individually. *Bottom row:* Histograms and Gaussian fits of aggregated errors.

tational errors, with the remainder made up of 'sliding' motions orthogonal to the direction of travel. These non-rotational errors are highly dependent on the specific trajectory, where more or less of the observed feature tracks can be explained by a sliding motion instead of a rotation. Due to the coupling of translational and rotational errors, correcting for rotational error in such cases may actually worsen the translational error (e.g., on the "Triangle" sequence).

While we do not implement this in our work, we speculate that incorporating an appropriate motion model into our VO formulation would significantly mitigate the impact of these errors by, for example, imposing a nonholonomic constraint on a ground vehicle or accounting for the dynamics of a quadcopter.

2.5 Urban Driving Experiments: The KITTI Odometry Benchmark

We investigated the performance of Sun-BCNN on the KITTI odometry benchmark training set (?Geiger et al., 2013), which consists of 21.6 km of urban driving data[3]. Importantly, the

[3]Because we rely on the first pose reported by the GPS/INS system, we used the raw (rectified and synchronized) sequences corresponding to each odometry sequence. However, the raw sequence `2011_09_26_drive_0067` corresponding to odometry sequence `03` was not available on the KITTI website

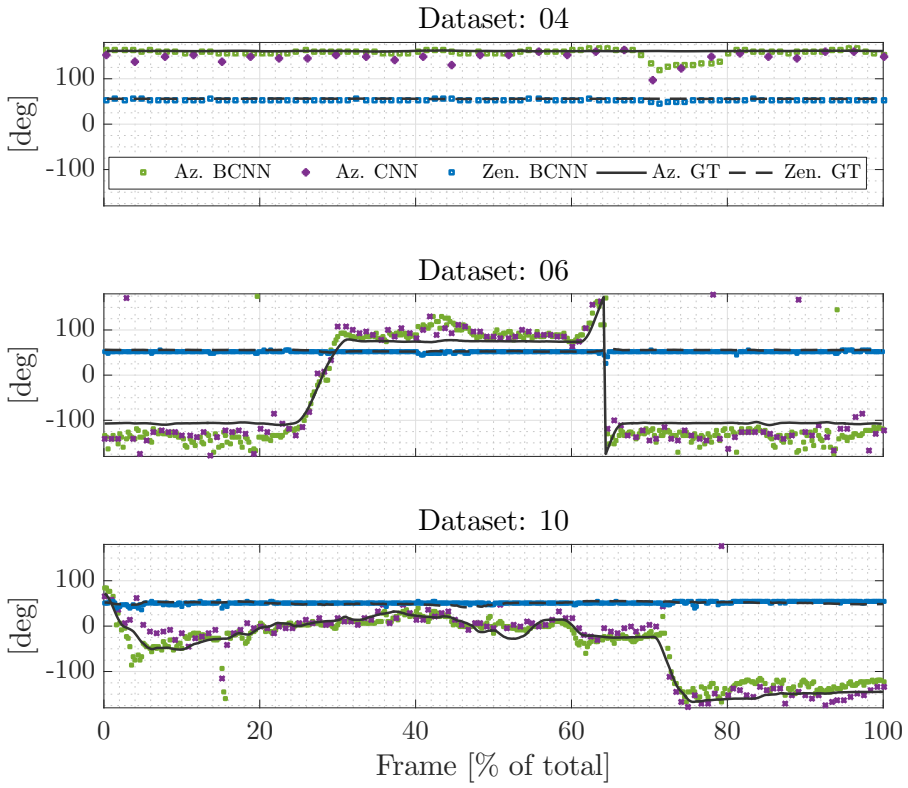


Figure 2.8: Azimuth (Sun-CNN and Sun-BCNN) and zenith (Sun-BCNN only) predictions over time for KITTI test sequences 04, 06 and 10. Sun-CNN is trained and tested on every tenth image, whereas Sun-BCNN is trained and tested on every image. In our VO experiments, we use the Sun-BCNN predictions of every tenth image to make a fair comparison.

Table 2.1: Test Errors for Sun-BCNN on KITTI odometry sequences with estimates computed at every image.

	Zenith Error [deg]			**Azimuth Error [deg]**			**Vector Angle Error [deg]**			
Sequence	Mean	Median	Stdev	Mean	Median	Stdev	Mean	Median	Stdev	**ANEES**[1]
00	-2.59	-1.37	5.15	-0.33	0.81	25.61	13.56	10.31	13.14	1.00
01	-12.53	-8.31	10.33	8.95	8.83	33.67	22.16	17.85	15.00	1.38
02	-6.13	-4.26	7.38	-1.03	0.74	37.61	19.69	14.32	18.25	1.40
04	-2.42	-2.11	1.64	-3.89	-2.18	9.14	5.33	3.29	6.44	0.30
05	-4.31	-2.51	6.18	-0.74	-3.80	29.81	15.66	11.33	14.80	1.05
06	-2.48	-2.52	2.27	-12.22	-17.86	25.78	19.78	17.72	11.35	1.93
07	-0.69	-0.16	3.26	1.25	5.98	20.27	12.44	10.05	9.97	0.97
08	-4.46	-1.61	8.14	3.66	-0.14	41.73	19.90	13.30	19.59	1.04
09	-1.35	-0.75	5.60	4.78	2.36	23.84	13.09	9.48	12.66	0.73
10	0.59	0.95	3.90	3.64	2.61	19.15	11.23	8.34	9.83	1.08
All	-4.01	-2.26	7.06	0.68	0.53	32.23	16.66	12.08	15.91	-

[1] We compute Average Normalized Estimation Error Squared (ANEES) values with all sun directions that fall below a cosine distance threshold of 0.3 (relative to ground truth) and set $\tau^{-1} = 0.015$.

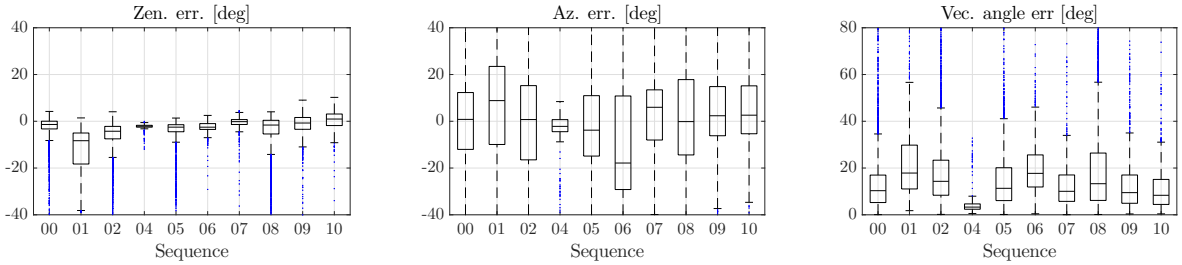
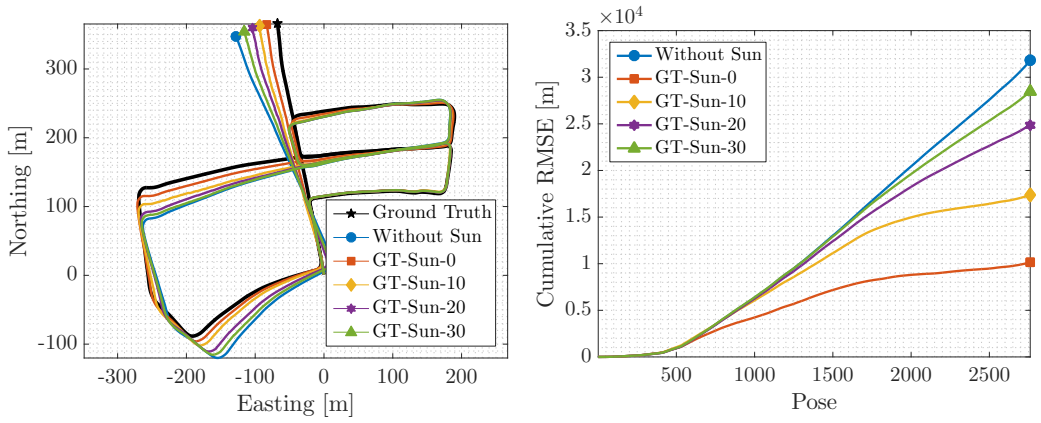
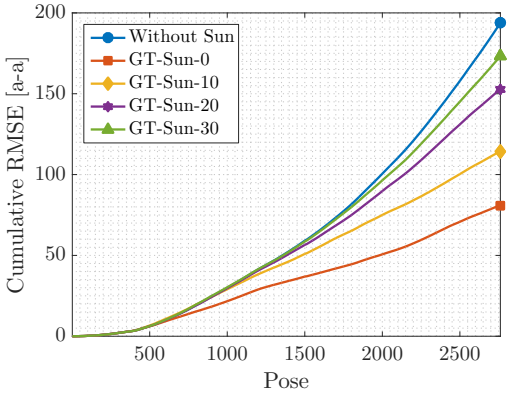


Figure 2.9: Box-and-whiskers plot of final test errors on all ten KITTI odometry sequences (c.f. Table 2.1).



(a) VO and ground truth trajectories

(b) Translational CRMSE (EN-plane)



(c) Rotational CRMSE

Figure 2.10: VO results for KITTI odometry sequence 05 using simulated sun measurements at every tenth pose. We observe a clear progression in cumulative root mean squared error (CRMSE) in translation and rotation as noise in the simulated sun measurements increases.

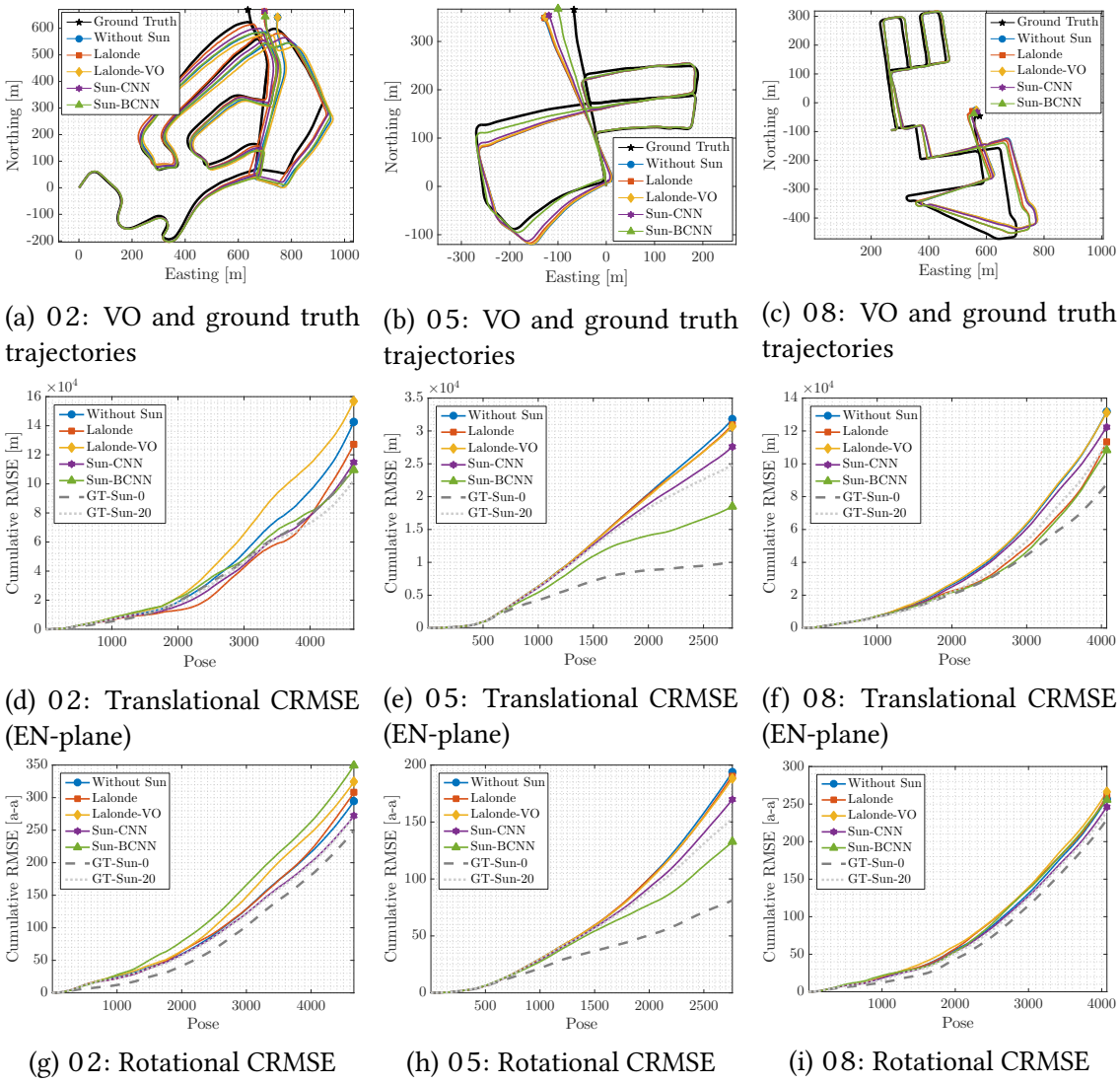
dataset includes 6-DOF ground truth poses obtained from an accurate GPS/INS tracking system, as well as calibrated transformations between this sensor and the colour stereo pair we use for sun estimation and VO in our experiments. This allows us to create a training set of ground truth sun vectors for each image by querying the solar ephemeris model at each ground truth pose and rotating the resulting vector from the GPS/INS frame $\underrightarrow{\mathcal{F}}_0$ (which is an ENU coordinate system) into the camera coordinate frame $\underrightarrow{\mathcal{F}}_k$. For each of our experiments, we trained Sun-BCNN on nine benchmark sequences and tested on the remaining sequence. This procedure is consistent with that of Ma et al. (2016), against whose Sun-CNN we directly compare, and allows us to evaluate each sequence using the maximum amount of training data.

Figure 2.11: Sun BCNN predictions and associated ground truth sun directions on the KITTI sequence 05. *Top two rows*: Sun BCNN produces accurate predictions in a variety of azimuth values. *Bottom row*: Poor results occur rarely due to shadow ambiguities.

2.5.1 Sun-BCNN Test Results

Once trained, we analyzed the accuracy and consistency of the Sun-BCNN mean and covariance estimates. We obtained the mean estimated sun vector by evaluating Equation (2.4) with $N = 25$ and then re-normalized the resulting vector to preserve unit length. To obtain the required covariance on azimuth and zenith angles, we sampled the vector outputs, converted them to azimuth and zenith angles using **??**, and then applied Equation (2.5). We investigate the impact of this parametrization (as opposed to working in azimuth and zenith coordinates directly) later in this paper. As shown in Table 2.1, we chose a value for the model precision τ such that the Average Normalized Estimation Error Squared (ANEES) of each test sequence is close to one (i.e., the estimator is consistent).

at the time of writing, so we omit sequence 03 from our analysis.



(a) 02: VO and ground truth trajectories

(b) 05: VO and ground truth trajectories

(c) 08: VO and ground truth trajectories

(d) 02: Translational CRMSE (EN-plane)

(e) 05: Translational CRMSE (EN-plane)

(f) 08: Translational CRMSE (EN-plane)

(g) 02: Rotational CRMSE

(h) 05: Rotational CRMSE

(i) 08: Rotational CRMSE

Figure 2.12: VO results for KITTI odometry sequences 02, 05, and 08 using estimate sun directions at every tenth pose. *Top row*: Estimated and ground truth trajectories in the Easting-Northing (EN) plane. *Middle row*: Translational cumulative root mean squared error (CRMSE) in the EN-plane. *Bottom row*: Rotational CRMSE. Sun-BCNN significantly reduces the estimation error on sequence 05, while the Lalonde (?), Lalonde-VO (?), and Sun-CNN (Ma et al., 2016) methods provide modest reductions in estimation error. The remaining sequences are less clear, but Sun-BCNN generally provides some benefit.

Table 2.2: Comparison of translational and rotational average root mean squared error (ARMSE) on KITTI odometry sequences with and without sun direction estimates at every tenth image. The best result (excluding simulated sun sensing) is highlighted in bold.

Sequence[1]	00	01[2]	02	04	05	06	07	08	09	10
Length [km]	3.7	2.5	5.1	0.4	2.2	1.2	0.7	3.2	1.7	0.9
Trans. ARMSE [m]										
Without Sun	4.33	198.52	28.59	2.48	9.90	3.35	4.55	28.05	10.44	5.54
GT-Sun-0	5.40	114.69	23.83	2.23	4.84	3.50	1.58	31.55	8.21	3.67
GT-Sun-10	4.85	123.84	25.34	2.45	5.84	2.80	2.94	28.47	8.65	4.81
GT-Sun-20	4.78	136.60	22.33	2.46	8.16	3.03	3.90	27.54	8.68	5.45
GT-Sun-30	4.83	157.14	27.30	2.48	8.93	3.44	4.62	26.73	10.10	5.28
Lalonde	**3.81**	200.34	28.13	**2.47**	9.88	3.36	4.61	29.70	10.49	**5.48**
Lalonde-VO	4.87	199.03	29.41	2.48	9.74	**3.30**	4.52	27.82	11.06	5.59
Sun-CNN	4.36	192.50	**26.58**	2.48	8.92	3.38	4.30	**26.99**	10.15	5.58
Sun-BCNN	4.44	**188.46**	26.89	2.48	**8.50**	4.10	**4.21**	27.71	**10.13**	5.61
Trans. ARMSE (EN-plane) [m]										
Without Sun	4.53	230.73	30.66	1.81	11.50	3.68	5.44	32.37	11.65	5.95
GT-Sun-0	3.41	136.76	24.12	1.46	3.67	3.96	1.80	21.51	7.77	3.71
GT-Sun-10	5.05	149.36	24.79	1.79	6.29	2.73	3.51	22.41	8.90	5.09
GT-Sun-20	5.14	164.37	22.04	1.80	9.01	3.13	4.66	27.58	8.86	5.81
GT-Sun-30	5.12	188.61	22.65	1.83	10.31	3.83	5.50	27.65	11.16	5.58
Lalonde	**3.95**	232.66	27.30	**1.81**	11.20	3.70	5.52	27.84	11.41	**5.87**
Lalonde-VO	5.38	231.33	33.68	1.82	11.13	**3.61**	5.42	32.24	12.41	6.00
Sun-CNN	4.56	224.91	24.65	1.82	9.99	3.74	5.16	30.09	11.21	5.99
Sun-BCNN	4.68	**220.54**	**23.58**	1.82	**6.70**	4.78	**5.05**	**26.59**	**10.97**	6.03
Rot. ARMSE $(\times 10^{-3})$ **[axis-angle]**										
Without Sun	23.88	185.30	63.18	12.97	70.18	23.24	49.96	63.13	26.77	21.54
GT-Sun-0	11.20	38.82	53.48	11.75	29.38	17.66	20.37	56.39	17.00	12.60
GT-Sun-10	17.05	64.51	58.78	12.86	41.47	18.90	34.05	54.89	19.71	14.26
GT-Sun-20	18.84	94.65	58.03	12.91	55.39	19.67	43.34	58.82	20.99	25.87
GT-Sun-30	23.40	121.21	57.79	13.01	62.73	23.96	49.92	56.74	25.63	20.15
Lalonde	**21.10**	188.06	66.02	**12.96**	69.00	23.27	50.49	64.22	26.27	**20.49**
Lalonde-VO	27.91	185.52	69.52	12.98	68.09	**22.79**	49.74	65.35	28.82	22.10
Sun-CNN	24.05	177.45	**58.32**	13.00	61.48	23.34	47.77	**60.55**	**26.19**	21.99
Sun-BCNN	26.96	**175.21**	75.02	13.00	**47.96**	23.80	**47.57**	62.85	26.29	20.85

[1] Because we rely on the timestamps and first pose reported by the GPS/INS system, we use the raw (rectified and synchronized) sequences corresponding to each odometry sequence. However, the raw sequence `2011_09_26_drive_0067` corresponding to odometry sequence 03 was not available on the KITTI website at the time of writing, so we omit sequence 03 from our analysis.

[2] Sequence 01 consists largely of self-similar, corridor-like highway driving which causes difficulties when detecting and matching features using `libviso2`. The base VO result is of low quality, although we note that including global orientation from the sun nevertheless improves the VO result.

Figure 2.9 and **??** plot the error distributions for azimuth, zenith, and angular distance for all ten KITTI odometry sequences, while **??** shows three characteristic plots of the azimuth and zenith predictions over time. We see that the errors in azimuth and zenith are strongly peaked around zero and are reasonably well described by a Gaussian distribution, which are important properties assumed by our VO pipeline to produce maximum likelihood motion estimates based on the fusion of multiple data sources. Note that the error distribution in zenith is slightly biased towards negative values due to the presence of a long tail on the negative side of the mean. This is an artifact of the azimuth-zenith parameterization when the sun zenith is small (i.e., when the sun is high in the sky), since zenith angles are defined on $[0, \pi]$. In practice, we attempt to reduce the influence of the long negative tail by imposing a robust Huber loss on the sun measurement errors in our optimization problem.

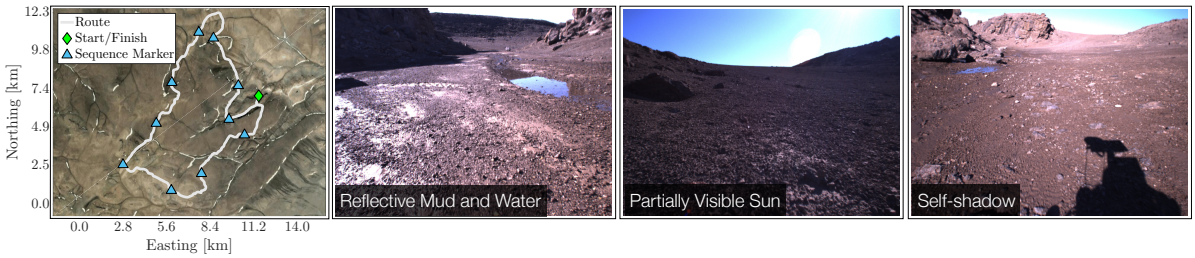


Figure 2.13: GPS track and sample images from the Devon Island traverse, with the start of each sequence highlighted. The Devon Island dataset is conducive to visual sun sensing due to the presence of strong environmental shadows, reflective surfaces such as mud and water, occasionally visible sun, and self-shadowing by the sensor platform. (Map data: Google, DigitalGlobe)

Table 2.1 summarizes the Sun-BCNN test errors numerically. Sun-BCNN achieved median vector angle errors of less than 15 degrees on every sequence except sequence 01 and 06, which were particularly difficult in places due to challenging lighting conditions. It is interesting to note that sequences 00 and 06 also have higher than average ANEES values, which indicates that the estimator is overconfident in its estimates despite their low quality. We suspect this behaviour stems from the assumption of homoscedastic noise in the BCNN, which treats all input images as being equally amenable to sun estimation across the entire sequence.

2.5.2 Visual Odometry Experiments

We evaluated the influence of the estimated sun directions and covariances obtained from Sun-BCNN on the KITTI odometry benchmark using the sun-aided VO pipeline previously described. To place these results in context, we compare them against the results obtained

using simulated sun measurements with varying levels of noise, the method of **?** and its VO-informed variant (**?**), and the Sun-CNN of Ma et al. (2016).

Simulated Sun Sensing

In order to gauge the effectiveness of incorporating sun information in each sequence, and to determine the impact of measurement error, we constructed several sets of simulated sun measurements by computing ground truth sun vectors and artificially corrupting them with varying levels of zero-mean Gaussian noise. We obtained these ground truth sun vectors by transforming the ephemeris vector into each camera frame using ground truth vehicle poses. Using the same convention as our experiments with simulated trajectories, we created four such measurement sets with $0°$, $10°$, $20°$, and $30°$ mean angular distance from ground truth.

Figure 2.10 shows the results we obtained using simulated sun measurements on sequence 05, in which the basic VO suffers from substantial orientation drift.[4] Incorporating absolute orientation information from the simulated sun sensor allows the VO to correct these errors, but the magnitude of the correction decreases as sensor noise increases, consistent with the results of our simulation experiments. As shown in **??**, which summarizes our VO results for all ten sequences, this is typical of sequences where orientation drift is the dominant source of error.

While the VO solutions for sequences such as 00 do not improve in terms of translational ARMSE, **??** shows that rotational ARMSE nevertheless improves on all ten sequences when low-noise simulated sun measurements are included. This implies that the estimation errors of the basic VO solutions for certain sequences are dominated by non-rotational effects, and that the apparent benefit of the Lalonde method on translational ARMSE in sequence 00 is likely coincidental.

Vision-based Sun Sensing

Figure 2.11 illustrates the behaviour of Sun-BCNN on four characteristic images from test sequence 05 by overlaying the Sun-BCNN predictions and associated ground truth sun directions for each image. The two frames in the top row both contain strong shadows which typically result in very accurate sun predictions. Conversely, the bottom row highlights two examples of rare situations where ambiguous shadows lead to very inaccurate predictions. As previously mentioned, we mitigate the influence of these outlier measurements by imposing a robust Huber loss on the sun measurement errors in our optimizer.

[4] In order to make a fair comparison to the Sun-CNN of Ma et al. (2016), who compute sun directions for every tenth image of the KITTI odometry benchmark, we subsample the sun directions obtained through each other method to match.

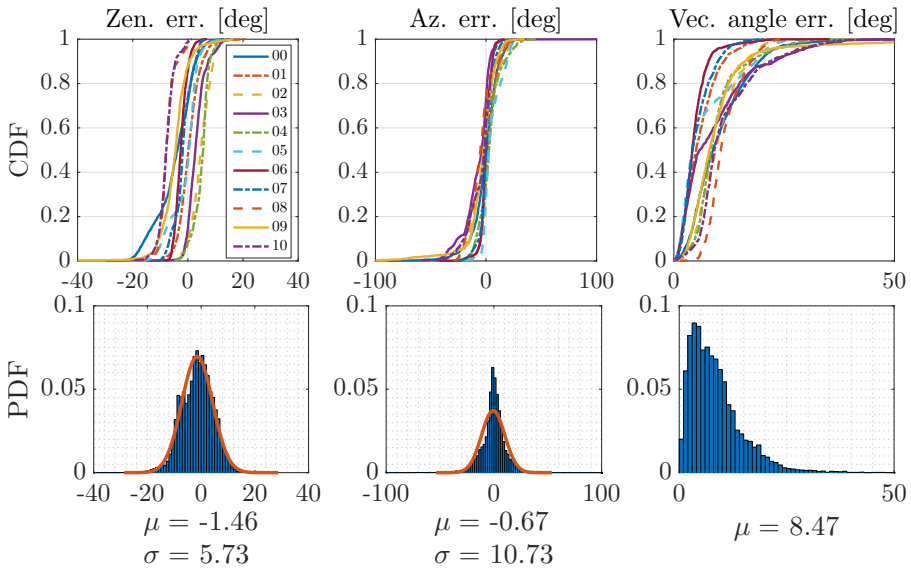


Figure 2.14: (Devon Island) Distributions of azimuth error, zenith error, and angular distance for Sun-BCNN compared to ground truth over each test sequence. *Top row*: Cumulative distributions of errors for each test sequence individually. *Bottom row:* Histograms and Gaussian fits of aggregated errors.

Figure 2.12 shows the results we obtained for sequences 02, 05, and 08 using the Sun-CNN of Ma et al. (2016), which estimates only the azimuth angle of the sun, our Bayesian Sun-BCNN which provides full 3D estimates of the sun direction as well as a measure of the uncertainty associated with each estimate, and the method of **?** in its original and VO-informed (**?**) forms, which provide 3D estimates of the sun direction without reasoning about uncertainty. A selection of results using simulated sun measurements are also displayed for reference. All four sun detection methods succeed in reducing the growth of total estimation error on this sequence, with Sun-BCNN reducing both translational and rotational error growth significantly more than the other three methods. Both Sun-CNN and Sun-BCNN outperform the two Lalonde variants, consistent with the results of Ma et al. (2016) and **?**.

?? shows results for all ten sequences using each method. With few exceptions, the VO results using Sun-BCNN achieve improvements in rotational and translational ARMSE comparable to those achieved using the simulated sun measurements with between 10 and 30 degrees average error. As previously noted, sequences such as 00 do not benefit significantly from sun sensing since rotational drift is not the dominant source of estimation error in these cases. Nevertheless, these results indicate that CNN-based sun sensing is a valuable tool for improving localization accuracy in VO and an improvement that comes without the need for additional sensors or a specially oriented camera.

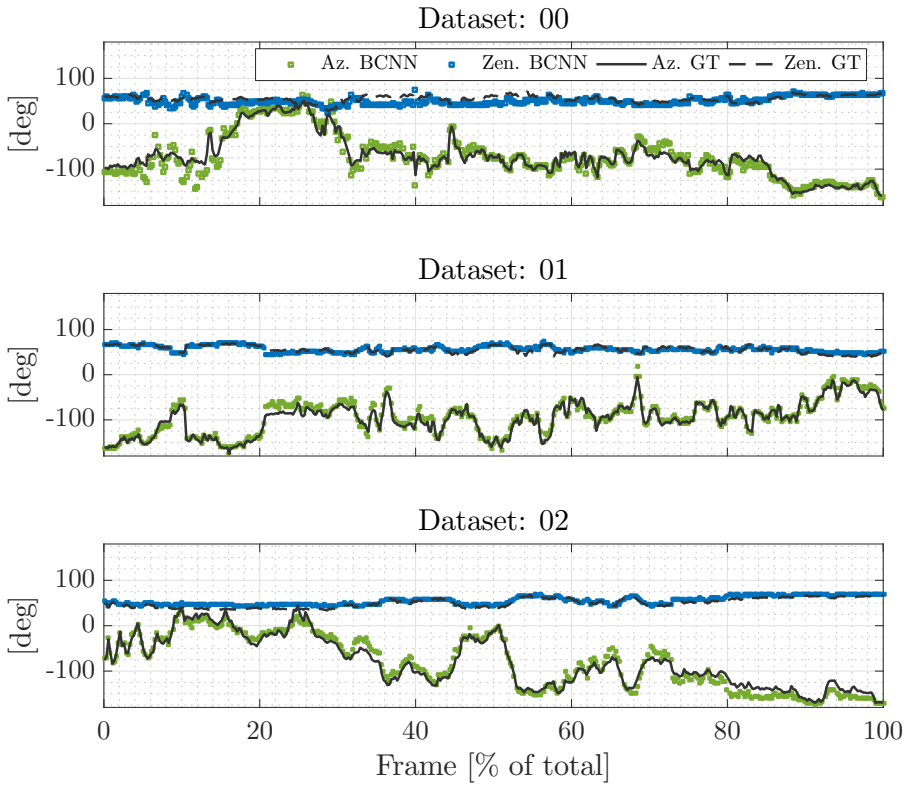


Figure 2.15: Azimuth (Sun-BCNN azimuth and zenith predictions over time for Devon Island test sequences 00, 01 and 12. Sun-BCNN is trained and tested on all frames (in our VO experiments, we use the Sun-BCNN predictions of every tenth image to make a fair comparison).

2.6 Planetary Analogue Experiments: The Devon Island Rover Navigation Dataset

In addition to urban driving, we further investigate the usefulness of Sun-BCNN in the context of planetary exploration using the Devon Island Rover Navigation Dataset (**?**), which consists of various sensor data collected using a mobile sensor platform traversing a 10 km loop on Devon Island in the Canadian High Arctic (Figure 2.13). The rugged landscape of Devon Island (Figure 2.13) is a significant departure from the structured urban environment of Karlsruhe. Unlike the KITTI odometry benchmark, the Devon Island dataset provides ground truth vehicle orientations for only a small number of images, which means that our previous method of generating ground truth sun vectors using ground truth poses is not applicable. However, the sensor platform used to collect the dataset was equipped with a hardware sun sensor and inclinometer, both of which were used by **?** to correct VO drift. For our purposes, we ignore the inclinometer and use the sun sensor measurements as training targets for Sun-BCNN.

The Devon Island environment contains many features one might expect to be amenable to visual sun detection. As shown in Figure 2.13, the dataset contains strong environmental shadows, stretches of wet terrain featuring reflective mud and water, and some self-shadowing from the sensor platform itself. At times the sun is partially visible to the camera, although these images tend to be saturated and do not immediately allow for accurate localization of the sun in the image.

For the purposes of our experiments, we partition the dataset into 11 sequences of approximately 1 km each, chosen such that the full pose of the vehicle at the beginning of each sequence is available from the ground truth data (see Figure 2.13). In aggregate, the sequences contain 13257 poses with associated sun sensor measurements. We apply a similar training and testing procedure as for the KITTI dataset, with the exception that we now withhold one sequence for validation and hyper-parameter tuning in addition to the sequence withheld for testing. This leaves nine sequences remaining to form the training sets for each test and validation pair.

2.6.1 Sun-BCNN Test Results

As in our experiments with the KITTI odometry benchmark, we obtained the mean estimated sun vector by evaluating Equation (2.4) with $N = 25$ and re-normalizing the resulting vector to preserve unit length. To obtain the required covariance on azimuth and zenith angles, we again sampled the vector outputs, converted them to azimuth and zenith angles using **??**, and then applied Equation (2.5). As shown in **??**, we chose a value for the model precision τ such

Table 2.3: Test Errors for Sun-BCNN on Devon Island odometry sequences with estimates computed at every image.

	Zenith Error [deg]			**Azimuth Error [deg]**			**Vector Angle Error [deg]**			
Sequence	Mean	Median	Stdev	Mean	Median	Stdev	Mean	Median	Stdev	**ANEES**[2]
00	-4.77	-3.77	6.82	-0.65	0.69	12.41	10.48	8.86	6.96	1.27
01	0.47	0.21	3.91	2.96	2.31	7.01	5.97	5.06	4.01	0.59
02	4.66	4.68	3.52	-0.72	-1.32	11.78	10.02	9.51	4.76	1.37
03	3.09	2.70	3.41	-7.47	-4.03	12.88	9.39	5.83	8.75	1.11
04	4.93	5.53	2.90	3.27	2.72	10.09	9.78	8.41	5.60	0.89
05	-1.01	0.46	4.97	5.26	2.46	8.23	7.19	4.15	6.60	0.92
06	-2.45	-2.58	2.23	-0.23	-0.30	5.07	4.72	4.17	3.16	0.31
07	-1.80	-1.87	3.28	0.47	0.20	6.45	5.23	4.25	3.38	0.41
08	-7.46	-7.88	2.85	-4.93	-5.14	10.30	11.61	10.63	3.96	1.33
09	-4.72	-4.46	5.27	-3.91	-2.13	14.61	9.90	8.02	8.56	0.86
10	-7.69	-7.82	2.92	-4.81	-1.54	10.80	11.79	9.19	7.52	0.91
All	-1.46	-1.23	5.73	-0.67	-0.14	10.73	8.47	7.15	6.31	-

[1] We compute Average Normalized Estimation Error Squared (ANEES) values with all sun directions that fall below a cosine distance threshold of 0.3 (relative to ground truth) and set $\tau^{-1} = 0.01$.

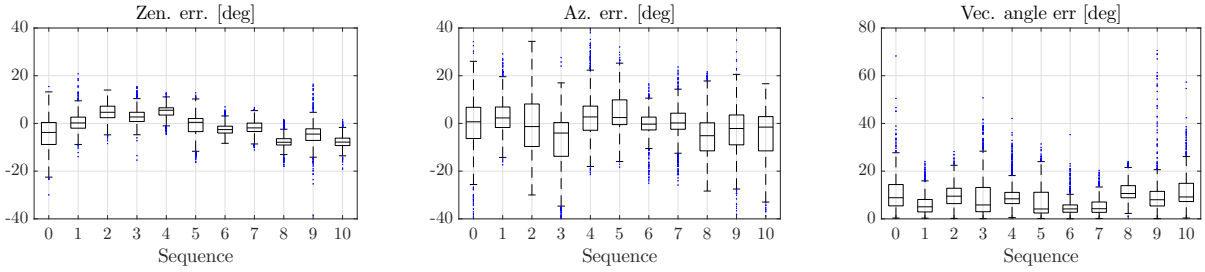


Figure 2.16: Box-and-whiskers plot of final test errors on Devon Island odometry sequences (c.f. ??).

that the Average Normalized Estimation Error Squared (ANEES) of each test sequence is close to one (i.e., the estimator is consistent).

???? plot the error distributions for azimuth, zenith, and angular distance for all 11 Devon Island odometry sequences, while **??** shows three characteristic plots of the azimuth and zenith predictions over time. We see that the errors in azimuth and zenith are strongly peaked around zero and are better described by a Gaussian distribution than in the case of KITTI (c.f. **??**), which as we previously mentioned are important properties assumed by our VO pipeline to appropriately fuse data. The distribution of zenith errors in the Devon Island dataset does not exhibit the same bias and long tail we observed in the KITTI dataset. This is likely because the sun is much lower in the sky (i.e., the zenith angle is further from zero) in the Devon Island dataset than in the KITTI dataset, so there is no clipping of the distribution near zero zenith.

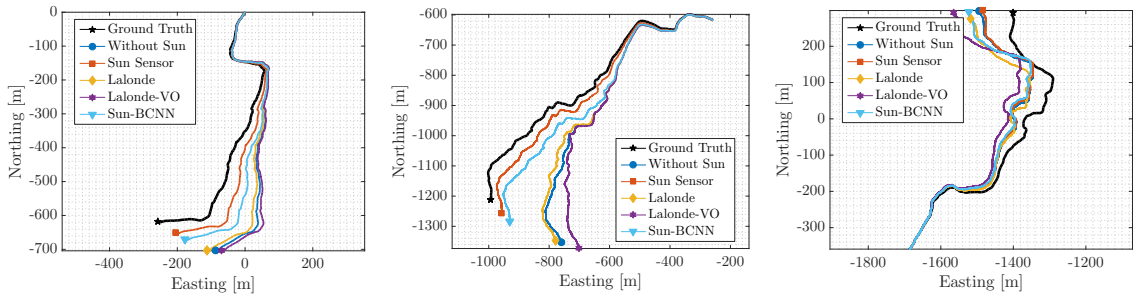
?? summarizes the test errors and ANEES of each sequence numerically, while **????** plot the error distributions for azimuth, zenith, and angular distance for each sequence. **??** shows three characteristic plots of the azimuth and zenith predictions over time. Sun-BCNN achieved median vector angle errors of less than 10 degrees on every sequence except sequence 08. Consistent with the results we observed in the KITTI experiments, the sequences with the highest median vector angle error (sequences 02 and 08) also have the highest ANEES values, again indicating that the homoscedastic noise assumption is perhaps ill suited to this environment.

2.6.2 Visual Odometry Experiments

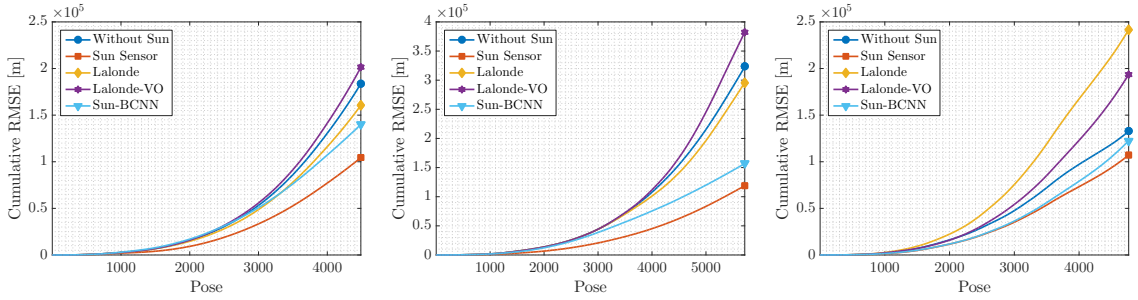
As in our KITTI benchmark experiments, we compare visual odometry results on each of our 11 test sequences both with sun-based orientation corrections and without. Notably, we do not report results using simulated sun measurements since we are unable to generate these measurements without ground truth vehicle orientations for every image. We also do not report results using the Sun-CNN of Ma et al. (2016) since we do not have access to their model. However, we do compare the results obtained using Sun-BCNN to those obtained using the hardware sun sensor as well as the Lalonde (**?**) and Lalonde-VO (**?**) methods.

?? shows sample VO results on three sequences from the Devon Island dataset using no sun measurements, the hardware sun sensor, Sun-BCNN, and the Lalonde variants. While the Lalonde methods struggle in this environment, Sun-BCNN yields significant improvements in VO accuracy, nearly on par with those obtained using the hardware sun sensor.

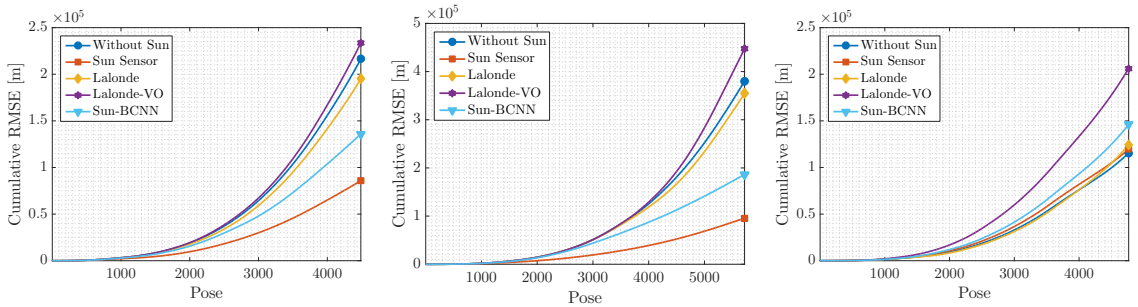
?? summarizes these results numerically for all 11 sequences in the dataset. While the addition of sun sensing using either the hardware sensor or Sun-BCNN generally results in significant reductions in error, we note that in certain cases (e.g., sequence 05), sun sensing



(a) 00: VO trajectories (EN-plane) (b) 01: VO trajectories (EN-plane) (c) 05: VO trajectories (EN-plane)



(d) 00: Translational CRMSE (e) 01: Translational CRMSE (f) 01: Translational CRMSE



(g) 00: Translational CRMSE (EN-Plane) (h) 01: Translational CRMSE (EN-Plane) (i) 05: Translational CRMSE (EN-Plane)

Figure 2.17: VO results for Devon Island sequences 00, 01, and 05 using estimated sun directions. *Top row*: Estimated and ground truth trajectories in the EN-plane. *Bottom rows*: Translational cumulative root mean squared error (CRMSE). Sun-BCNN significantly reduces the estimation error on sequences where the sun sensing has an impact (c.f. **??**).

Table 2.4: Comparison of average root mean squared error (ARMSE) on Devon Island sequences with and without sun direction estimates using both a hardware sun sensor and vision-based methods. The best result using a vision-based method is bolded.

Sequence	00	01	02	03	04	05	06	07	08	09	10
Length [km]	0.9	1.1	1.0	1.0	0.9	1.0	1.1	1.0	0.9	0.7	0.6
Trans. ARMSE [m]											
Without Sun	40.93	56.51	41.58	42.04	30.52	27.82	58.91	40.04	47.22	11.39	12.94
Hardware Sun Sensor	23.26	20.79	9.79	22.03	30.79	22.47	24.14	29.59	47.97	6.26	8.50
Lalonde	35.77	51.74	53.32	47.00	39.55	50.70	94.77	59.37	**45.78**	10.03	16.23
Lalonde-VO	44.83	66.91	44.17	59.84	42.87	40.62	52.16	36.04	50.52	11.34	16.74
Sun-BCNN	**31.17**	**27.45**	**16.00**	**26.02**	**29.34**	**25.70**	**33.43**	**32.25**	50.80	**4.27**	**14.92**
Trans. ARMSE (EN-plane) [m]											
Without Sun	48.20	66.49	43.58	45.92	31.08	24.23	43.01	22.33	40.85	9.30	15.59
Hardware Sun Sensor	19.13	16.74	8.99	21.18	28.27	25.08	29.27	21.76	28.89	5.14	9.70
Lalonde	43.45	62.03	36.21	49.44	**20.13**	**26.13**	53.22	**18.10**	35.62	6.01	18.45
Lalonde-VO	52.05	78.26	40.20	59.09	50.12	43.28	53.62	42.71	49.99	11.74	20.17
Sun-BCNN	**30.28**	**32.65**	**9.62**	**14.32**	33.26	30.62	**36.44**	23.18	**13.53**	**4.45**	**14.75**

has little or no impact on the VO result. We suspect that the translation errors in these cases are dominated by non-rotational effects, similarly to those observed in our experiments with the KITTI dataset, although it is difficult to be certain in the absence of rotational ground truth. As previously mentioned, the incorporation of a motion prior in the VO estimator would likely reduce the impact of these errors.

2.7 Sensitivity Analysis

In this section we analyze the sensitivity of our model to cloud cover, investigate the possibility of model transfer between urban and planetary analogue environments, and examine the impact of different methods for computing the mean and covariance of a norm-constrained vector on the accuracy and consistency of the estimated sun directions.

2.7.1 Cloud Cover

Given that both the KITTI and Devon Island datasets were collected in sunny conditions, it is natural to wonder whether and to what extent Sun-BCNN is affected by cloud cover. As shown in **??**, Sun-BCNN relies in part on shadows and other local illumination variations to estimate the direction of the sun. Since the diffuse nature of daylight in cloudy conditions tends to soften shadows and other shading variations, one might expect Sun-BCNN to perform worse in cloudy conditions. Accordingly, we investigated the effect of cloud cover on Sun-BCNN

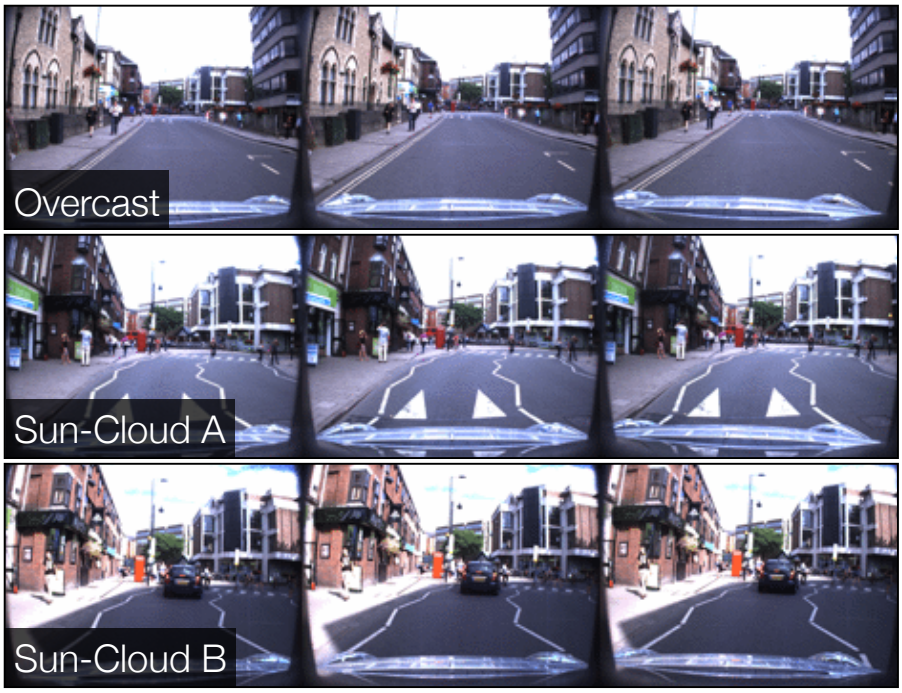


Figure 2.18: Sample images of approximately the same location taken from three different Oxford Robotcar sequences we used to investigate the effect of cloud cover on Sun-BCNN.

using selected sequences from the Oxford Robotcar Dataset (**?**), which consists of 1000 km of urban driving along a consistent route but in varying weather conditions and at varying times over the course of a year.

Procedure

We selected three sequences collected within a two hour period on the same day (namely 2014-07-14-14-49-50, 2014-07-14-15-16-36, and 2014-07-14-15-42-55), which consist of the same route observed under different lighting conditions. Figure 2.18 presents sample images from each of these sequences, which we label *Overcast*, *Sun-Cloud A*, and *Sun-Cloud B*, respectively. To evaluate the performance of Sun-BCNN in each of these conditions, we partition each sequence into a randomly selected set of training (80%), validation (10%) and test (10%) images, and then train and test Sun-BCNN on each of the nine train-test permutations.

Results

Figure 2.19 shows the results of these experiments with box and whisker plots for azimuth, zenith and vector angle errors while **??** summarizes the results numerically. We obtained the most accurate test predictions using the model trained on *Sun-Cloud B*, the sequence with the

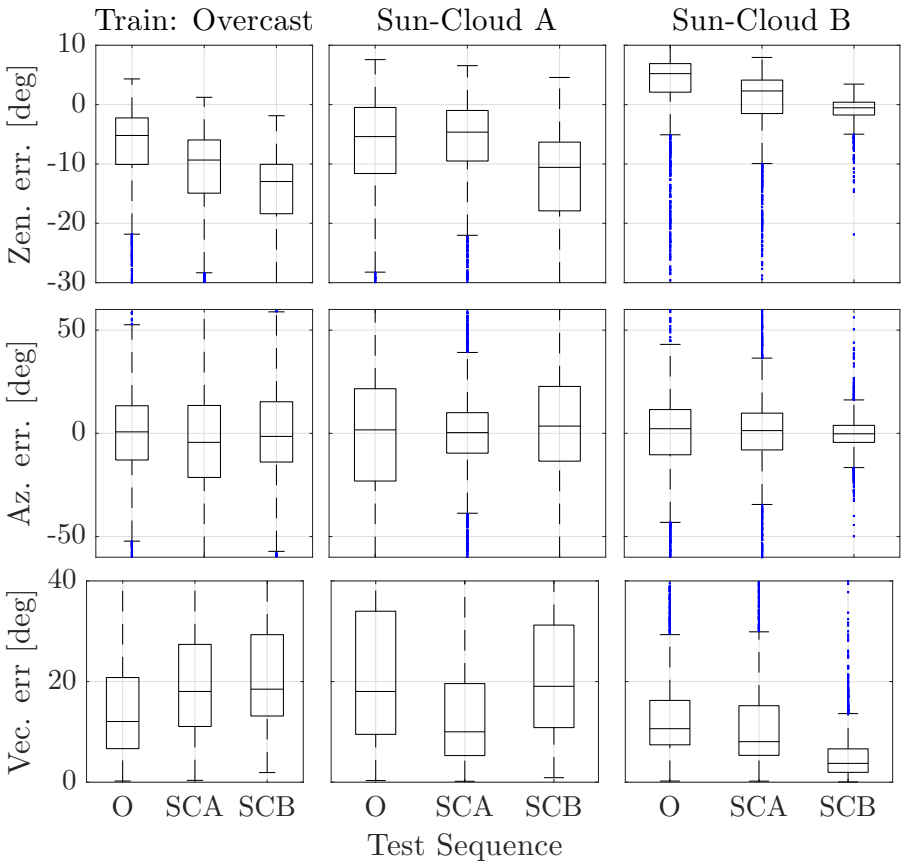


Figure 2.19: Box-and-whiskers plot for zenith, azimuth and vector angle errors for nine different combinations of train-test sequences taken from the Oxford Robotcar dataset. Each column corresponds to a different training sequence, and each plot contains three different test sequences. In the bottom legend, we use the labels O: *Overcast*, SCA: *Sun-Cloud A*, SCB: *Sun-Cloud B*.

least amount of cloud cover. Notably, this model produced vector angle errors on the *Overcast* test set that were lower than those trained with its own *Overcast* training set. Moreover, we note that the *Sun-Cloud A* model achieved similar test errors when applied to the *Sun-Cloud B* test set as when applied to the *Overcast* test set. Similarly, the *Sun-Cloud B* model achieved similar test errors when applied to the *Sun-Cloud A* test set as when applied to the *Overcast* test set. From this we can conclude the following: 1) that Sun-BCNN can still perform well in the presence of cloud cover; and 2) that training in environments illuminated by strong directional light (i.e., sunny conditions) can significantly improve sun estimation accuracy in different test conditions.

2.7.2 Model Generalization

It may also be natural to ask how well a Sun-BCNN model trained in an urban environment performs in a planetary analogue environment and vice versa. This would provide some indication of whether the model generalizes to new environments or if a philosophy of place-specific excellence (e.g., the place-specific visual features of **?**) is more appropriate for the task of illumination estimation.

Procedure

We attempted to answer this question by creating three larger datasets from combinations of the sequences used in our previous experiments:

1. KITTI odometry sequences 00 - 10;
2. Devon Island sequences 00 - 10; and
3. the previously discussed *Overcast*, *Sun-Cloud A*, and *Sun-Cloud B* sequences from the Oxford Robotcar dataset.

We randomly partitioned each dataset into training (90%) and test (10%) sets. We then trained three separate Sun-BCNN models on each training set, and evaluated each trained model on each of the three test sets.

Results

?? shows the results of these experiments with box and whisker plots for azimuth, zenith and vector angle errors while **??** summarizes the results numerically. We see that none of the three models generalize well to environments other than the one in which they were trained, yielding large and significantly biased test errors. We note, however, that the Oxford model

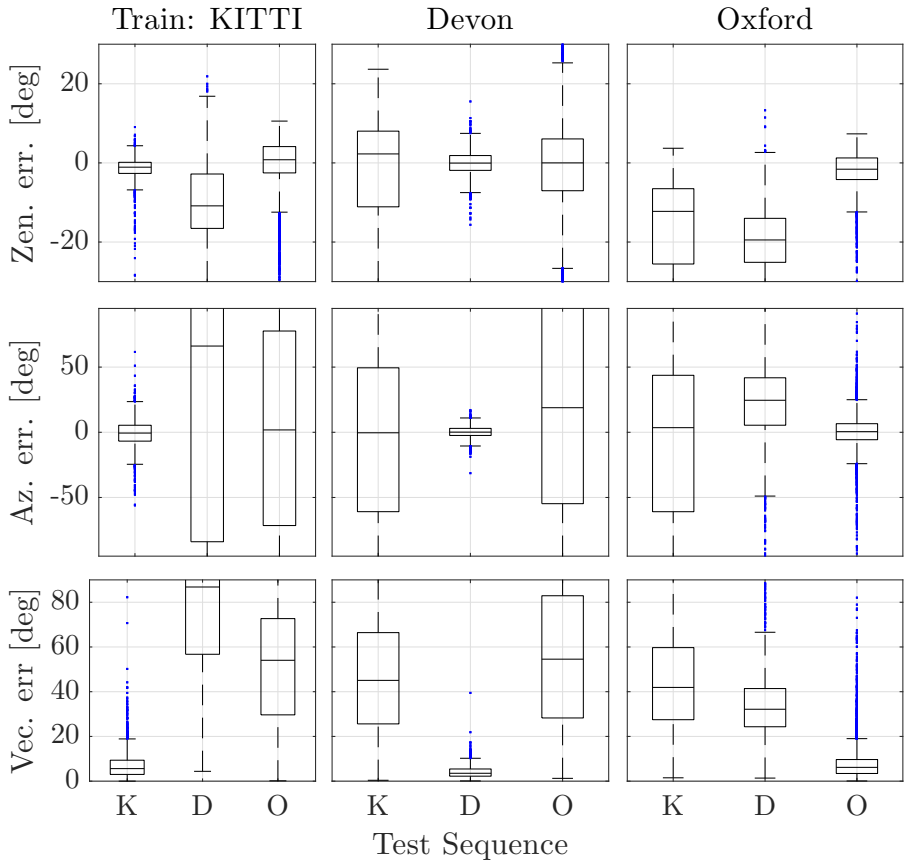


Figure 2.20: Box-and-whiskers plot for zenith, azimuth and vector angle errors for nine different combinations of train-test datasets. Each column corresponds to a different training sequence, and each plot contains three different test sequences. In the bottom legend, we use the labels K: KITTI, D: Devon Island, O: Oxford. All three models produce large biased errors when applied to other datasets, likely due to variations in optical properties and parameter settings across cameras.

was the least egregious offender, and speculate that this may be because the Oxford sequences contain significantly more training images than the other two datasets (approximately 3 times as many as the KITTI odometry benchmark and 5 times as many as the Devon Island dataset).

A possible explanation for the poor generalization of these models is the fact that each dataset was collected using different cameras with different optical properties and parameter settings. We believe these differences affect Sun-BCNN's ability to recover an accurate estimate of a three dimensional direction vector, since metrically important quantities such as the principal point and focal length of the sensor can vary significantly from camera to camera. Furthermore, differences in dynamic range may also significantly affect the ability of Sun-BCNN to treat shading variations consistently.

Table 2.5: Test Errors for Sun-BCNN on three different Oxford Robotcar sequences collected on the same day with different lighting conditions.

		Zenith Error [deg]			**Azimuth Error [deg]**			**Vector Error [deg]**		
Train	**Test**	Mean	Median	Std.	Mean	Median	Std.	Mean	Median	Std.
Overcast[1]	Overcast	-7.12	-5.20	7.04	-0.66	0.72	29.36	15.22	12.06	11.73
	Sun-Cloud A	-11.58	-9.34	7.94	-5.71	-4.37	37.21	21.19	18.03	14.07
	Sun-Cloud B	-15.23	-12.96	8.00	0.05	-1.49	38.83	23.36	18.49	15.05
Sun-Cloud A[2]	Overcast	-7.17	-5.39	9.05	-0.67	1.68	51.27	23.66	18.03	18.11
	Sun-Cloud A	-6.49	-4.64	7.88	0.29	0.35	27.42	14.31	10.02	12.75
	Sun-Cloud B	-12.89	-10.58	8.94	1.87	3.51	40.41	23.45	19.06	16.75
Sun-Cloud B[3]	Overcast	3.34	5.22	6.46	-0.32	2.24	26.07	13.95	10.63	11.32
	Sun-Cloud A	-0.14	2.30	7.36	-1.08	1.34	28.54	13.76	8.06	14.60
	Sun-Cloud B	-0.84	-0.54	2.07	-0.36	-0.22	9.00	5.11	3.73	5.13

[1] 2014-07-14-14-49-50 [2] 2014-07-14-15-16-36 [3] 2014-07-14-15-42-55

Table 2.6: Test Errors for Sun-BCNN on different training and test datasets.

		Zenith Error [deg]			**Azimuth Error [deg]**			**Vector Error [deg]**		
Train	**Test**	Mean	Median	Std.	Mean	Median	Std.	Mean	Median	Std.
KITTI	KITTI	-1.49	-1.08	2.99	-0.64	-0.60	11.46	7.16	5.61	6.23
	Devon Island	-9.27	-10.86	9.97	26.78	66.15	113.23	81.32	86.82	33.48
	Oxford	-0.02	0.80	6.59	-0.44	1.81	91.30	52.39	54.05	29.46
Devon Island	KITTI	-2.37	2.27	14.30	-5.58	-0.38	78.01	48.16	45.06	27.85
	Devon Island	-0.08	-0.05	3.20	0.20	0.12	5.52	4.24	3.52	2.96
	Oxford	-1.35	0.00	11.57	17.12	18.85	96.86	55.52	54.55	29.88
Oxford	KITTI	-17.05	-12.25	13.19	-6.94	3.55	77.70	44.66	41.91	23.00
	Devon Island	-20.07	-19.47	9.81	20.92	24.56	45.52	35.16	32.15	16.07
	Oxford	-1.96	-1.59	4.60	0.19	0.48	15.08	8.08	6.16	7.68

2.7.3 Mean and Covariance Computation

In our formulation, Sun-BCNN outputs a sampling of unit-norm 3D vectors. Due to the unit-norm constraint, it is not immediately clear how to apply Equations (2.4) and (2.5) to calculate the mean and covariance of these samples. In this section we present and empirically evaluate two possible procedures for each computation using the previously discussed combined datasets for KITTI, Devon Island, and Oxford.

Mean computation

Procedure We investigated two different methods for computing the mean of the sampled sun vectors, which we refer to as *Method I* and *Method II*.

1. In *Method I* (used in this work), we first evaluate Equation (2.4) directly on the constrained unit vectors produced by N stochastic passes through the BCNN. We then re-normalize the resulting mean vector to enforce unit length, and convert it to azimuth and zenith angles using **??**.

2. In *Method II*, we first convert each of the N unit vectors produced through stochastic passes through the BCNN to azimuth and zenith angles using **??**. We then evaluate Equation (2.4) on the angles themselves to obtain the mean in azimuth-zenith coordinates.

We evaluated both methods using the same combined datasets and partitioning scheme as in the transfer learning experiment previously presented.

Results Table 2.7 presents the azimuth, zenith and vector errors for the two mean computation methods. *Method I* produces lower vector errors and smaller standard deviations in azimuth and zenith on all three datasets.

Covariance Computation

Procedure We further investigated two different covariance computation methods, which we also refer to as *Method I* and *Method II*.

1. In *Method I*, we first evaluate Equation (2.5) directly on the constrained unit vectors produced by N stochastic passes through the BCNN, yielding a 3×3 covariance. We then compute a 2×2 covariance on azimuth and zenith by propagating the 3×3 covariance through a linearized **??**.

Table 2.7: A comparison of prediction errors from different mean estimation methods.

		Zenith Error [deg]			**Azimuth Error [deg]**			**Vector Error [deg]**		
Sequence	**Mean Type**	Mean	Median	Std.	Mean	Median	Std.	Mean	Median	Std.
KITTI	Method I	-1.50	-1.06	2.96	-0.56	-0.47	11.52	7.16	5.52	6.27
	Method II	-1.06	-0.76	2.44	-0.30	-0.37	30.18	11.49	5.95	18.60
Devon	Method I	-0.07	0.02	3.18	0.19	0.27	5.76	4.22	3.55	3.04
	Method II	0.04	0.09	3.17	1.11	0.26	24.62	9.19	4.05	20.22
Oxford	Method I	-1.97	-1.66	4.59	0.20	0.51	15.31	8.12	6.10	7.74
	Method II	-1.45	-1.27	3.95	-1.58	0.11	34.46	13.18	6.76	19.24

2. In *Method II* (used in this work), we first convert each of the N unit vectors produced by stochastic passes through the BCNN to azimuth and zenith angles, and then evaluate Equation (2.5) on the angles themselves.

We once again re-used the transfer learning datasets with the same partitioning scheme, and evaluated covariances on the test sets corresponding to each of the three models. To control for the effect of tuning the model precision τ, we replace the diagonal elements of each covariance matrix with the diagonal elements of the empirical covariance corresponding to the entire test set (computed based ground truth azimuth and zenith errors). We then compared the consistency of the cross-correlations of each method (i.e., the off-diagonal components of the covariance matrix) by computing ANEES values over the each model's corresponding test set using both mean computation methods.

Results **??** lists the ANEES values produced by each method of covariance computation when paired with each mean computation method. *Method I* covariances produced better ANEES values when paired with *Method I* mean estimation, but *Method II* covariances paired well with either mean estimation scheme.

Table 2.8: A comparison of ANEES values for different mean and covariance propagation methods.

Sequence	Covariance Type	Mean Type	ANEES
KITTI	Method I	Method I	0.95
		Method II	5.10
	Method II	Method I	1.40
		Method II	0.87
Devon	Method I	Method I	1.29
		Method II	10.05
	Method II	Method I	0.50
		Method II	0.85
Oxford	Method I	Method I	1.50
		Method II	2.14
	Method II	Method I	1.30
		Method II	0.89

Chapter 3

Conclusion

> Of what a strange nature is knowledge!
> It clings to a mind when it has once
> seized on it like a lichen on a rock.
>
> Mary Shelley, *Frankenstein; or, The Modern Prometheus*

And thus, we have reached the end. In this thesis, we developed a general framework for improving the performance of model-based visual-inertial localization pipelines by complementing them with learned probabilistic 'pseudo-sensors' that extract difficult-to-model latent information from sensor data. We presented four examples of such *pseudo-sensors*.

3.1 Summary of Contributions

Predictive Robust Estimation

The contributions of PROBE are:

1. a probabilistic model for sparse stereo visual odometry, leading to a predictive robust algorithm for inference on that model,
2. a procedure for training our model using pairs of stereo images with known relative transform, and
3. an iterative, expectation-maximization approach to train our model when the relative ground truth egomotion is unavailable.

Publications:

- Peretroukhin, V., Vega-Brown, W., Roy, N., and Kelly, J. (2016b). PROBE-GK: Predictive robust estimation using generalized kernels. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'16)*, pages 817–824, Stockholm, Sweden

- Peretroukhin, V., Clement, L., Giamou, M., and Kelly, J. (2015a). PROBE: Predictive robust estimation for visual-inertial navigation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'15)*, pages 3668–3675, Hamburg, Germany

- Peretroukhin, V., Clement, L., and Kelly, J. (2015b). Get to the point: Active covariance scaling for feature tracking through motion blur. In *Proceedings of the IEEE International Conference on Robotics and Automation Workshop on Scaling Up Active Perception*, Seattle, Washington, USA

Sun BCNN

- Peretroukhin, V., Clement, L., and Kelly, J. (2018). Inferring sun direction to improve visual odometry: A deep learning approach. *International Journal of Robotics Research*

- Peretroukhin, V., Clement, L., and Kelly, J. (2017a). Reducing drift in visual odometry by inferring sun direction using a bayesian convolutional neural network. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'17)*, pages 2035–2042, Singapore

- Clement, L., Peretroukhin, V., and Kelly, J. (2017). Improving the accuracy of stereo visual odometry using visual illumination estimation. In Kulic, D., Nakamura, Y., Khatib, O., and Venture, G., editors, *2016 International Symposium on Experimental Robotics*, volume 1 of *Springer Proceedings in Advanced Robotics*, pages 409–419. Springer International Publishing, Berlin Heidelberg. Invited to Journal Special Issue

In Sun-BCNN, our main contributions can be summarized as follows:

1. We apply a Bayesian CNN to the problem of sun direction estimation, incorporating the resulting covariance estimates into a visual odometry pipeline;

2. We demonstrate that a Bayesian CNN with dropout layers after each convolutional and fully-connected layer can achieve state-of-the-art accuracy at test time;

3. We learn a 3D unit-length sun direction vector, appropriate for full 6-DOF pose estimation;

4. We present experimental results on over 30 km of visual navigation data in urban (**?**) and planetary analogue (**?**) environments;

5. We investigate the sensitivity of our Bayesian CNN-based sun estimator (Sun-BCNN) to cloud cover, camera and environment changes, and measurement parameterization; and

6. We release Sun-BCNN as open-source software[1].

Deep Pose Corrections

In summary, the main contributions of DPC:

1. the formulation of a novel deep corrective approach to egomotion estimation,

2. a novel cost function for deep $\mathrm{SE}(3)$ regression that naturally balances translation and rotation errors, and

3. an open-source implementation of DPC-Net in `PyTorch`[2].

- Peretroukhin, V. and Kelly, J. (2018). DPC-Net: Deep pose correction for visual localization. *IEEE Robotics and Automation Letters*

Estimating Rotation through Deep Probabilistic Inference with HydraNet

The contributions of this work are:

1. a deep network structure we call *HydraNet* that builds on prior work **??** to produce meaningful uncertainties over unconstrained targets,

2. a loss formulation and mathematical framework that extends HydraNet to means and covariances of the rotation group $\mathrm{SO}(3)$,

3. and open source code for $\mathrm{SO}(3)$ regression[3].

- Peretroukhin, V., Wagstaff, B., and Kelly, J. (2019). Deep probabilistic regression of elements of so(3) using quaternion averaging and uncertainty injection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'19) Workshop on Uncertainty and Robustness in Deep Visual Learning*, pages 83–86, Long Beach, California, USA

[1] `https://github.com/utiasSTARS/sun-bcnn-vo`.
[2] See `https://github.com/utiasSTARS/dpc-net`.
[3] Repository redacted to protect author anonymity.

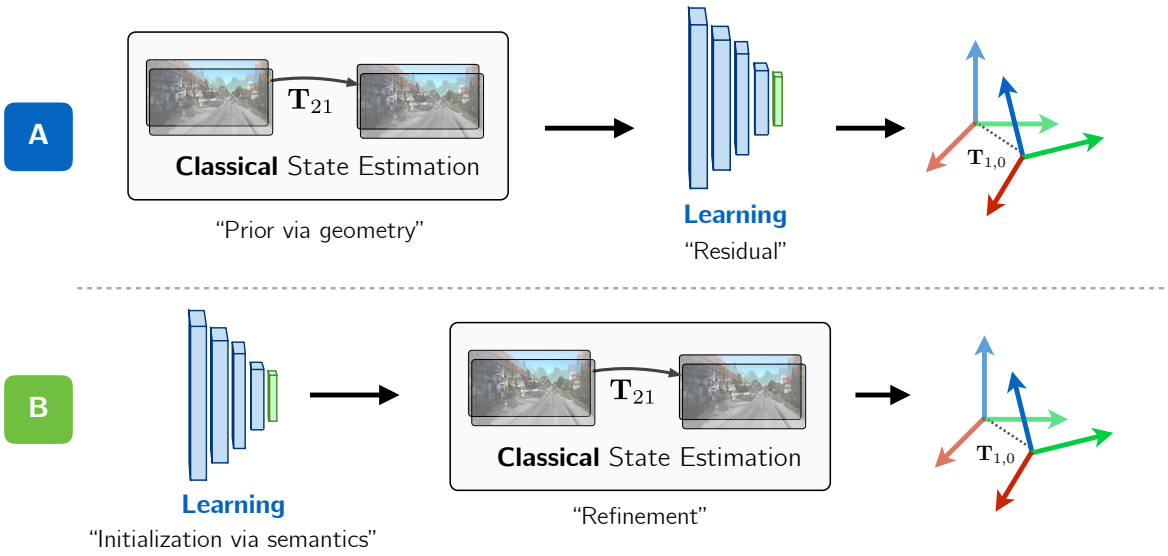


Figure 3.1: Two different ways to incorporate learning with classical pipelines. The first is by using pipelines as a *prior* which can then be corrected by learned approaches, while the second is use learning as an initialization which can then be *refined* by classical techniques.

3.2 Future Work

There are many avenues for future work. For instance, although the fusion of pipelines with pseudo-sensors can significantly improve localization performance in a given environment, there are few guarantees that the final estimates are accurate and consistent

A potential thread of future work could address this deficiency by developing more tightly-coupled perception systems that can be 'certified' to produce globally-optimal solutions while still being robust to adverse environmental effects. By incorporating these learned 'priors' into an optimization framework, I plan to use recently- developed theory on convex relaxations (e.g., that presented in **ToDo: add citations here**) to ensure that the final localization and mapping results are 'optimal', in the sense that they are the global minima of a maximum-likelihood-based loss, and 'safe', in the sense that they provide consistent uncertainty estimates even in the presence of unmodelled effects.

3.3 Final Remarks

The German Philosopher Georg Hegel had a particular dialectic method that involved a triad: a thesis, antithesis and synthesis. Somewhat curiously, this *thesis* has been an attempt at a *synthesis* of the thesis posed by classical visual egomotion, and the antithesis posed by data-drive end-to-end learning techniques. My hope is that the synthesis based on the concept of

pseudo-sensors will prove to be a fruitful one within the field of state-estimation, and for the broader robotics community.

Appendices

Appendix A

Left and Middle Perturbations

A.1 Identities

$$\operatorname{Exp}\left(\left(\boldsymbol{\xi}+\delta\boldsymbol{\xi}\right)\right) \approx \operatorname{Exp}\left(\left(\boldsymbol{\mathcal{J}}\delta\boldsymbol{\xi}\right)\right)\operatorname{Exp}\left(\boldsymbol{\xi}\right), \tag{A.1}$$

$$\begin{aligned}\log\left(\mathbf{T}_1\mathbf{T}_2\right)^{\vee} &= \log\left(\operatorname{Exp}\left(\boldsymbol{\xi}_1\right)\operatorname{Exp}\left(\boldsymbol{\xi}_2\right)\right)^{\vee} \\ &\approx \begin{cases} \boldsymbol{\mathcal{J}}(\boldsymbol{\xi}_2)^{-1}\boldsymbol{\xi}_1+\boldsymbol{\xi}_2 & \text{if } \boldsymbol{\xi}_1 \text{ small} \\ \boldsymbol{\xi}_1+\boldsymbol{\mathcal{J}}(-\boldsymbol{\xi}_1)^{-1}\boldsymbol{\xi}_2 & \text{if } \boldsymbol{\xi}_2 \text{ small.} \end{cases}\end{aligned} \tag{A.2}$$

A.2 Perturbing $\mathrm{SE}(3)$

Consider the quantity,

$$\mathbf{T}_{ba} = \mathbf{T}_{bi}{\mathbf{T}_{ai}}^{-1} \tag{A.3}$$

where $\underrightarrow{\mathcal{F}}_i$ is some inertial frame.

A.2.1 Left Perturbation

Separating $\mathbf{T}_{ba}$ into a mean component, $\overline{\mathbf{T}}_{ba}$, and a small left perturbation,

$$\mathbf{T}_{ba} = \operatorname{Exp}\left(\delta\boldsymbol{\xi}^l_{ba}\right)\overline{\mathbf{T}}_{ba} = \operatorname{Exp}\left(\delta\boldsymbol{\xi}^l_{ba}\right)\operatorname{Exp}\left(\overline{\boldsymbol{\xi}}_{ba}\right) \tag{A.4}$$

Applying a logarithm to both sides,

$$\log\left(\mathbf{T}_{ba}\right)^{\vee} = \log\left(\operatorname{Exp}\left(\delta\boldsymbol{\xi}^l_{ba}\right)\operatorname{Exp}\left(\overline{\boldsymbol{\xi}}_{ba}\right)\right)^{\vee} \tag{A.5}$$

Using Equation (A.2),

$$\log\left(\mathbf{T}_{ba}\right)^{\vee} \approx \overline{\boldsymbol{\xi}}_{ba} + \boldsymbol{\mathcal{J}}_{ba}^{-1}\delta\boldsymbol{\xi}_{ba}^{l} \tag{A.6}$$

where $\boldsymbol{\mathcal{J}}_{ba} \triangleq \boldsymbol{\mathcal{J}}(\overline{\boldsymbol{\xi}}_{ba})$. This is exactly Equation 6.104 in Sean Anderson's thesis.

A.2.2 Middle Perturbation

Now consider the middle perturbation,

$$\mathbf{T}_{ba} = \operatorname{Exp}\left(\overline{\boldsymbol{\xi}}_{ba} + \delta\boldsymbol{\xi}_{ba}^{m}\right) \tag{A.7}$$

Immediately, we can take the logarithm of both sides and see that,

$$\log\left(\mathbf{T}_{ba}\right)^{\vee} = \overline{\boldsymbol{\xi}}_{ba} + \delta\boldsymbol{\xi}_{ba}^{m}, \tag{A.8}$$

where we now observe that $\delta\boldsymbol{\xi}_{ba}^{l} \approx \boldsymbol{\mathcal{J}}_{ba}\delta\boldsymbol{\xi}_{ba}^{m}$.

A.3 DPC $\mathrm{SE}(3)$ Loss

Using the notation in this document, the DPC derivation requires an expression for $\delta\boldsymbol{\xi}_b$, and assumes that $\boldsymbol{\xi}_a$ is constant.

A.3.1 Middle Perturbation

Consider,

$$\mathbf{T}_{ba} = \mathbf{T}_b{\mathbf{T}_a}^{-1} \tag{A.9}$$

where we have dropped the i frame for clarity. Middle perturbing $\mathbf{T}_{ba}$ and $\mathbf{T}_b$ and keeping $\mathbf{T}_a$ fixed (i.e., $\mathbf{T}_a = \overline{\mathbf{T}}_a$).

$$\operatorname{Exp}\left(\overline{\boldsymbol{\xi}}_{ba} + \delta\boldsymbol{\xi}_{ba}^{m}\right) = \operatorname{Exp}\left(\overline{\boldsymbol{\xi}}_{b} + \delta\boldsymbol{\xi}_{b}^{m}\right)\overline{\mathbf{T}}_a^{-1} \tag{A.10}$$

Using Equation (A.1) twice,

$$\operatorname{Exp}\left(\left(\boldsymbol{\mathcal{J}}_{ba}\delta\boldsymbol{\xi}_{ba}^{m}\right)\right)\operatorname{Exp}\left(\overline{\boldsymbol{\xi}}_{ba}\right) = \operatorname{Exp}\left(\left(\boldsymbol{\mathcal{J}}_{b}\delta\boldsymbol{\xi}_{b}^{m}\right)\right)\operatorname{Exp}\left(\overline{\boldsymbol{\xi}}_{b}\right)\overline{\mathbf{T}}_a^{-1} \tag{A.11}$$

Collecting terms, we have

$$\operatorname{Exp}\left(\left(\boldsymbol{\mathcal{J}}_{ba}\delta\boldsymbol{\xi}_{ba}^{m}\right)\right)\overline{\mathbf{T}}_{ba}=\operatorname{Exp}\left(\left(\boldsymbol{\mathcal{J}}_{b}\delta\boldsymbol{\xi}_{b}^{m}\right)\right)\overline{\mathbf{T}}_{b}\overline{\mathbf{T}}_{a}^{-1} \tag{A.12}$$

Right multiplying by $\overline{\mathbf{T}}_{ba}^{-1}$, we are left with

$$\operatorname{Exp}\left(\left(\boldsymbol{\mathcal{J}}_{ba}\delta\boldsymbol{\xi}_{ba}^{m}\right)\right)=\operatorname{Exp}\left(\left(\boldsymbol{\mathcal{J}}_{b}\delta\boldsymbol{\xi}_{b}^{m}\right)\right) \tag{A.13}$$

or

$$\boldsymbol{\mathcal{J}}_{ba}\delta\boldsymbol{\xi}_{ba}^{m}=\boldsymbol{\mathcal{J}}_{b}\delta\boldsymbol{\xi}_{b}^{m}. \tag{A.14}$$

Solving for $\delta\boldsymbol{\xi}_{ba}^{m}$,

$$\delta\boldsymbol{\xi}_{ba}^{m}=\boldsymbol{\mathcal{J}}_{ba}^{-1}\boldsymbol{\mathcal{J}}_{b}\delta\boldsymbol{\xi}_{b}^{m}. \tag{A.15}$$

Now inserting Equation (A.15) into Equation (A.8),

$$\log\left(\mathbf{T}_{ba}\right)^{\vee}\approx\overline{\boldsymbol{\xi}}_{ba}+\boldsymbol{\mathcal{J}}_{ba}^{-1}\boldsymbol{\mathcal{J}}_{b}\delta\boldsymbol{\xi}_{b}^{m} \tag{A.16}$$

This is exactly Equation 13 in the DPC-Net paper:

$$g(\boldsymbol{\xi}+\delta\boldsymbol{\xi})\approx\boldsymbol{\mathcal{J}}(g(\boldsymbol{\xi}))^{-1}\boldsymbol{\mathcal{J}}(\boldsymbol{\xi})\delta\boldsymbol{\xi}+g(\boldsymbol{\xi}). \tag{A.17}$$

A.3.2 Left Perturbation

Using the left perturbation, we can repeat the procedure of relating $\delta\boldsymbol{\xi}_{ba}^{l}$ and $\delta\boldsymbol{\xi}_{b}^{l}$ (by perturbing $\mathbf{T}_{ba}$ and $\mathbf{T}_{b}$ and keeping $\mathbf{T}_{a}$ fixed (i.e., $\mathbf{T}_{a}=\overline{\mathbf{T}}_{a}$).

$$\operatorname{Exp}\left(\delta\boldsymbol{\xi}_{ba}^{l}\right)\overline{\mathbf{T}}_{ba}=\operatorname{Exp}\left(\delta\boldsymbol{\xi}_{b}^{l}\right)\overline{\mathbf{T}}_{b}\overline{\mathbf{T}}_{a}^{-1} \tag{A.18}$$

from which we see immediately that $\operatorname{Exp}\left(\delta\boldsymbol{\xi}_{ba}^{l}\right)=\operatorname{Exp}\left(\delta\boldsymbol{\xi}_{b}^{l}\right)$ and therefore,

$$\delta\boldsymbol{\xi}_{ba}^{l}=\delta\boldsymbol{\xi}_{b}^{l} \tag{A.19}$$

Now using Equation (A.6), we have

$$\log\left(\mathbf{T}_{ba}\right)^{\vee}\approx\overline{\boldsymbol{\xi}}_{ba}+\boldsymbol{\mathcal{J}}_{ba}^{-1}\delta\boldsymbol{\xi}_{b}^{l} \tag{A.20}$$

A.3.3 Summary

Using the left perturbation, we have

$$\log\left(\mathbf{T}_{ba}\right)^{\vee} \approx \overline{\boldsymbol{\xi}}_{ba} + \boldsymbol{\mathcal{J}}_{ba}^{-1}\delta\boldsymbol{\xi}_{b}^{l} \tag{A.21}$$

Using the centre/middle perturbation, we have

$$\log\left(\mathbf{T}_{ba}\right)^{\vee} \approx \overline{\boldsymbol{\xi}}_{ba} + \boldsymbol{\mathcal{J}}_{ba}^{-1}\boldsymbol{\mathcal{J}}_{b}\delta\boldsymbol{\xi}_{b}^{m} \tag{A.22}$$

And we see the same earlier expression relating left and middle perturbations,

$$\delta\boldsymbol{\xi}^{l} \approx \boldsymbol{\mathcal{J}}\delta\boldsymbol{\xi}^{m} \tag{A.23}$$

A.3.4 Reconciliation

Consider the two update rules:

$$\mathbf{T}_{b} \leftarrow \operatorname{Exp}\left(\delta\boldsymbol{\xi}_{b}^{l}\right)\overline{\mathbf{T}}_{b} \tag{A.24}$$

$$\mathbf{T}_{b} \leftarrow \operatorname{Exp}\left((\overline{\boldsymbol{\xi}}_{b} + \delta\boldsymbol{\xi}_{b}^{m})\right) \tag{A.25}$$

But using Equation (A.1), the middle update becomes,

$$\operatorname{Exp}\left((\overline{\boldsymbol{\xi}}_{b} + \delta\boldsymbol{\xi}_{b}^{m})\right) \approx \operatorname{Exp}\left((\boldsymbol{\mathcal{J}}_{b}\delta\boldsymbol{\xi}_{b}^{m})\right)\operatorname{Exp}\left(\overline{\boldsymbol{\xi}}_{b}\right) = \operatorname{Exp}\left((\boldsymbol{\mathcal{J}}_{b}\delta\boldsymbol{\xi}_{b}^{m})\right)\overline{\mathbf{T}}_{b} = \operatorname{Exp}\left(\delta\boldsymbol{\xi}_{b}^{l}\right)\overline{\mathbf{T}}_{b} \tag{A.26}$$

So the middle perturbation does not require us to keep the mean in the group (as long as we avoid any degeneracies).

Bibliography

Alcantarilla, P. F. and Woodford, O. J. (2016). Noise models in feature-based stereo visual odometry.

Barfoot, T. D. (2017). *State Estimation for Robotics*. Cambridge University Press.

Clement, L., Peretroukhin, V., and Kelly, J. (2017). Improving the accuracy of stereo visual odometry using visual illumination estimation. In Kulic, D., Nakamura, Y., Khatib, O., and Venture, G., editors, *2016 International Symposium on Experimental Robotics*, volume 1 of *Springer Proceedings in Advanced Robotics*, pages 409–419. Springer International Publishing, Berlin Heidelberg. Invited to Journal Special Issue.

Cvišić, I. and Petrović, I. (2015). Stereo odometry based on careful feature selection and tracking. In *Proc. European Conf. on Mobile Robots (ECMR)*, pages 1–6.

DeTone, D., Malisiewicz, T., and Rabinovich, A. (2016). Deep image homography estimation.

Fitzgibbon, A. W., Robertson, D. P., Criminisi, A., Ramalingam, S., and Blake, A. (2007). Learning priors for calibrating families of stereo cameras. In *Proc. IEEE Int. Conf. Computer Vision (ICCV)*, pages 1–8.

Forster, C., Pizzoli, M., and Scaramuzza, D. (2014). SVO: Fast semi-direct monocular visual odometry. In *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, pages 15–22.

Garg, R., Carneiro, G., and Reid, I. (2016). Unsupervised CNN for single view depth estimation: Geometry to the rescue. In *European Conf. on Comp. Vision*, pages 740–756. Springer.

Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. (2013). Vision meets robotics: The KITTI dataset. *Int. J. Rob. Res.*, 32(11):1231–1237.

Grewal, M. S. and Andrews, A. P. (2010). Applications of kalman filtering in aerospace 1960 to the present [historical perspectives]. *IEEE Control Syst. Mag.*, 30(3):69–78.

Handa, A., Bloesch, M., Pătrăucean, V., Stent, S., McCormac, J., and Davison, A. (2016). gvnn: Neural network library for geometric computer vision. In *Computer Vision – ECCV 2016 Workshops*, pages 67–82. Springer, Cham.

Kendall, A. and Cipolla, R. (2016). Modelling uncertainty in deep learning for camera relocalization. In *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, pages 4762–4769.

Lambert, A., Furgale, P., Barfoot, T. D., and Enright, J. (2012). Field testing of visual odometry aided by a sun sensor and inclinometer. *J. Field Robot.*, 29(3):426–444.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436–444.

Leutenegger, S., Lynen, S., Bosse, M., Siegwart, R., and Furgale, P. (2015). Keyframe-based visual–inertial odometry using nonlinear optimization. *Int. J. Rob. Res.*, 34(3):314–334.

Ma, W.-C., Wang, S., Brubaker, M. A., Fidler, S., and Urtasun, R. (2016). Find your way by observing the sun and other semantic cues.

Mayor, A. (2019). *Gods and Robots*. Princeton University Press.

Melekhov, I., Ylioinas, J., Kannala, J., and Rahtu, E. (2017). Relative camera pose estimation using convolutional neural networks. In *Proc. Int. Conf. on Advanced Concepts for Intel. Vision Syst.*, pages 675–687. Springer.

Nilsson, N. J. (1984). Shakey the robot. Technical report, SRI INTERNATIONAL MENLO PARK CA.

Oliveira, G. L., Radwan, N., Burgard, W., and Brox, T. (2017). Topometric localization with deep learning.

Peretroukhin, V., Clement, L., Giamou, M., and Kelly, J. (2015a). PROBE: Predictive robust estimation for visual-inertial navigation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'15)*, pages 3668–3675, Hamburg, Germany.

Peretroukhin, V., Clement, L., and Kelly, J. (2015b). Get to the point: Active covariance scaling for feature tracking through motion blur. In *Proceedings of the IEEE International Conference on Robotics and Automation Workshop on Scaling Up Active Perception*, Seattle, Washington, USA.

Peretroukhin, V., Clement, L., and Kelly, J. (2017a). Reducing drift in visual odometry by inferring sun direction using a bayesian convolutional neural network. In *Proceedings of*

the IEEE International Conference on Robotics and Automation (ICRA'17), pages 2035–2042, Singapore.

Peretroukhin, V., Clement, L., and Kelly, J. (2017b). Reducing drift in visual odometry by inferring sun direction using a bayesian convolutional neural network. In *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*.

Peretroukhin, V., Clement, L., and Kelly, J. (2018). Inferring sun direction to improve visual odometry: A deep learning approach. *International Journal of Robotics Research*.

Peretroukhin, V. and Kelly, J. (2018). DPC-Net: Deep pose correction for visual localization. *IEEE Robotics and Automation Letters*.

Peretroukhin, V., Vega-Brown, W., Roy, N., and Kelly, J. (2016a). PROBE-GK: Predictive robust estimation using generalized kernels. In *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, pages 817–824.

Peretroukhin, V., Vega-Brown, W., Roy, N., and Kelly, J. (2016b). PROBE-GK: Predictive robust estimation using generalized kernels. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'16)*, pages 817–824, Stockholm, Sweden.

Peretroukhin, V., Wagstaff, B., and Kelly, J. (2019). Deep probabilistic regression of elements of so(3) using quaternion averaging and uncertainty injection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'19) Workshop on Uncertainty and Robustness in Deep Visual Learning*, pages 83–86, Long Beach, California, USA.

Redfield, S. (2019). A definition for robotics as an academic discipline. *Nature Machine Intelligence*, 1(6):263–264.

Scaramuzza, D. and Fraundorfer, F. (2011). Visual odometry [tutorial]. *IEEE Robot. Autom. Mag.*, 18(4):80–92.

Sünderhauf, N., Shirazi, S., Dayoub, F., Upcroft, B., and Milford, M. (2015). On the performance of ConvNet features for place recognition. In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Syst. (IROS)*, pages 4297–4304.

Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G., Lau, K., Oakley, C., Palatucci, M., Pratt, V., Stang, P., Strohband, S., Dupont, C., Jendrossek, L.-E., Koelen, C., Markey, C., Rummel, C., van Niekerk, J., Jensen, E., Alessandrini, P., Bradski, G., Davies, B., Ettinger, S., Kaehler, A., Nefian, A.,

and Mahoney, P. (2006). Stanley: The robot that won the DARPA grand challenge. *J. Field Robotics*, 23(9):661–692.

Tsotsos, K., Chiuso, A., and Soatto, S. (2015). Robust inference for visual-inertial sensor fusion. In *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, pages 5203–5210.

Vega-Brown, W. R., Doniec, M., and Roy, N. G. (2014). Nonparametric Bayesian inference on multivariate exponential families. In *Proc. Advances in Neural Information Proc. Syst. (NIPS) 27*, pages 2546–2554.

Zhou, B., Krähenbühl, P., and Koltun, V. (2019). Does computer vision matter for action?