Efficient Bayesian Estimation for Localization and Mapping

Yonhon Ng

Australian National University

A thesis submitted for the degree of
Doctor of Philosophy
The Australian National University

May 2019

Except where otherwise indicated, this thesis is my own original work.
The nature and extent of collaboration have been outlined in this thesis.

Yonhon Ng
18 May 2019

To my family.

"He who learns but does not think, is lost!
He who thinks but does not learn is in peril."

— Confucius[1]

"Education is not the learning of facts,
it's rather the training of the mind to think. "

— Albert Einstein[2]

"Human progress has always been driven by
a sense of adventure and unconventional thinking. "

— Andre Geim[3]

[1]Well known Chinese philosopher
[2]Winner of 1921 Nobel Prize in Physics
[3]Winner of 2010 Nobel Prize in Physics

Declaration

My doctoral studies have been conducted under the guidance and supervision of Assoc. Prof. Jonghyuk Kim, Assoc. Prof. Changbin (Brad) Yu and Assoc. Prof. Hongdong Li. This thesis does not contain materials which has been accepted for the award of any other degree or diploma from any university.

Acknowledgments

There are many people who I would like to thank for their time, patience, and support over the last several years during my PhD degree.

First, I would like to express my sincerest gratitude to my supervisors Assoc. Prof. Jonghyuk (Jon) Kim, Assoc. Prof. Changbin (Brad) Yu and Assoc. Prof. Hongdong Li. It is a great privilege to have the opportunity to work with such high calibre academics in their respective research fields. Their research expertise and willingness to share their knowledge are vital for the completion of my thesis.

I would like to thank my supervisor Jon who has provided close guidance and support to me during the final year of my PhD. Jon is always very patient, and is willing to spare his precious time with me to discuss research issues even if we do not have a prior appointment. Beside the academic help, he also shows great concern about his students' well-being and future plan.

I am grateful to my supervisor Brad, who is also the supervisor for my undergraduate honours project. Brad has organised frequent group meetings where his students have a chance to present their work and share ideas. Some of these meetings were jointly organized by researchers from the University of Technology Sydney and Defence Science and Technology group. I have benefited greatly by participating in these group meetings where my presentation skill is honed and is often inspired by the high quality presentations given by the other speakers. He has also given a lot of freedom to his students to work on research that interests them the most while helping them to succeed along the way.

I would like to also express my appreciation towards my supervisor Hongdong. He has helped me a lot in topics related to computer vision. His knowledge and passion in his work have always been inspirational for me. With his encouragement, I have had the opportunity to become a tutor for the robotics and computer vision courses which I thoroughly enjoyed. The design of teaching materials and interaction with bright undergraduate students has taught me a great deal. It has helped to strengthen my understanding of the subjects and has incubated my interest in teaching.

I have had a wonderful time at ANU, and I credit this to the amazing colleagues whom I have the opportunity to work with. They are Mengbin (Ben) Ye, Dr. Yun Hou, Dr. Junming Wei, Dr. Xiaolei Hou, Dr. Zhiyong Sun, Dr. Qingchen Liu, Zhixun Li, Benjamin Nizette, Edwin Davis, Dr. Jiaolong Yang, Dr. Gao Zhu, Dr. Pan Ji, Mina Henein, Cristian Rodriguez, Yi Zhou, Yang Liu, Yiran Zhong, Jun Zhang, Liu Liu, Dr. Yifei Huang and many more.

Last but not least, I am greatly indebted to all the tremendous support, love and understanding from my family during all these years. I would like to thank my

parents, Hockweng Ng and Siewlee Ang who has always believed in me, and my brother Yonsen Ng whom I have always looked up to. Thank you.

Abstract

This thesis addresses the theoretical and practical development of efficient Bayesian filtering algorithms for use in robotic localization and mapping. Full Bayesian filters generally require an infinite number of parameters to maintain the full conditional probability density function (PDF), which is computationally intractable. The extended Kalman filter, Gaussian sum and particle filter are commonly used to address the above problem. The limitations of these methods are the inherent trade-off between accuracy and computational complexity, and difficulty in ensuring consistent estimation. This thesis investigates the use of degenerate Gaussian density functions to approximate the nonlinear measurement densities arising in various sensing systems, such as conical density in bearing sensors, or spherical density in ranging sensors. There are four main contributions:

First, we propose the Minimal Iterative Gaussian Estimator (MIGE), which utilizes a degenerate Gaussian density to approximate the nonlinear measurement likelihood. A degenerate Gaussian allows uncertainty to be infinite along some directions, allowing the representation of cylindrical and planar likelihood functions. A minimal parametric representation of the Gaussian likelihood function is developed, which allows for simple measurement likelihood update. Through Monte Carlo simulation, we show improved accuracy and consistency for bearing-only localization, while using the least amount of memory and computational time, when compared to existing popular filters.

Second, the MIGE algorithm is applied to improve the performance of Time Difference of Arrival (TDOA) and Frequency Difference of Arrival (FDOA) localization. TDOA is a differenced range measurement forming a hyperboloid distribution. FDOA is a pseudo bearing measurement forming conical distributions for a stationary emitter. Existing methods typically utilize linearization methods by computing Jacobians. The MIGE-based method is shown to better approximate the measurement density. Outliers may also be present in real-data experiments, which may degrade estimator's performance. It is shown that MIGE can effectively handle the outliers by utilizing a bounding box method. Simulations and experiments using real data collected from sensors (receivers) and a target (radio-station) demonstrate the improved localization accuracy.

Third, the MIGE algorithm is applied to improve the performance of visual simultaneous localization and mapping (SLAM). The visual mapping process requires three-dimensional triangulation of scene points. We apply MIGE by utilizing the cylindrical degenerate Gaussian for the triangulation with minimal parametric representation. Next, the Bayes Dense Flow (BDF) algorithm is proposed for a SLAM front-end module to address the difficulty of feature-limited scenes in a probabilistic framework. A new Mahalanobis eight-point algorithm is also proposed, which min-

imises the Mahalanobis distance of the epipolar line to each optical flow estimate. By combining the BDF, Mahalanobis eight-point algorithm and MIGE, a robust visual odometry is designed. The visual odometry is then combined with an existing SLAM back-end, called robust linear pose-graph. The resulting visual SLAM is shown to be more accurate for a standard dataset and our own UAV dataset, effectively handling feature-limited scenes, pure rotational motion and large camera height variations.

Lastly, with the robustly estimated camera pose from our visual SLAM method, it is possible to estimate a smooth camera trajectory for digital video stabilization. We propose a method using window-based weighted pair-wise rotation average to obtain a smooth rotational motion. Improved video stabilization performance is shown with the proposed method.

Contents

List of Figures

Chapter 1

Introduction

This chapter begins with discussing issues that motivate this research work relating to the topic of developing a new approximate Bayesian estimator for localization and mapping tasks. It also presents the objectives, contributions and thesis structure.

1.1 Motivations

The work of this thesis falls into the broad research field of stochastic filtering, widely studied by different research communities for many decades. Stochastic filtering is applied to many engineering fields such as robot localization and navigation, sensor network, target tracking and so on, where one wants to estimate the states of a nonlinear dynamic system from noisy measurements and imperfect knowledge.

Stochastic filtering was first established by the pioneering work of Wiener and Hopf [1931] and Kolmogorov [1941]. Their work then led to the well-known Kalman filter [Kalman, 1960] (and subsequently Kalman-Bucy filter [Kalman and Bucy, 1961]). The Kalman filter is a stochastic filter designed to solve the state estimation problem for a linear dynamical system with Gaussian perturbation. The effectiveness of the Kalman filter was evidenced by its application in the navigation system of the Apollo lunar mission [Schmidt, 1981]. The Kalman filter has also been applied in various scientific areas, such as communications, economics, finance, biology and so on.

A general class of stochastic filters is the Bayesian filter, where the random processes are modelled as conditional probability densities. These probability densities are then combined using Bayesian theory by Thomas Bayes [Bayes and Price, 1763]. Bayesian filtering techniques were first developed for state estimation in Ho and Lee [1964]. The Kalman filter was also derived under Bayesian framework in [Meinhold and Singpurwalla, 1983].

Most estimation problems involve nonlinear dynamic or measurement process, where the assumptions used by the Kalman filter are not valid, and indeed the estimation problems considered in this thesis are all nonlinear. Full Bayesian estimators in general, require an infinite number of parameters to describe the underlying conditional probability density function (PDF). Thus, approximate Bayesian methods were developed to tackle nonlinear estimation problems.

The most popular efficient non-linear estimator is the Extended Kalman Filter

(EKF) [Sorenson, 1985]. The EKF approximates the non-linearities of the system by computing Jacobians. However, the EKF is known to be inconsistent, where the level of uncertainty in the state estimate is often underestimated [Huang et al., 2010; Li and Mourikis, 2013]. The Gaussian sum filter [Sorenson and Alspach, 1971; Alspach and Sorenson, 1972] is another nonlinear estimator, where the nonlinear PDF is approximated by weighted sum of multiple Gaussian PDFs. The Gaussian sum filter has an inherent trade-off between the computational complexity and accuracy, where using more Gaussian densities can better approximate the true PDF but incurs higher computational and memory cost.

An alternative to Gaussian sum filter is the sequential Monte Carlo based Bayesian filter commonly known as particle filter [Gordon et al., 2002], where the PDF is represented using a set of particles. However, similar to Gaussian sum, the particle filter needs to retain high number of particles to ensure accurate approximation of the PDF, which in turn leads to higher computational cost. Another nonlinear Bayesian estimator approximately captures the PDF using a grid-based method [Arulampalam et al., 2002]. The discrete conditional probabilities at different locations are computed and recursively updated. However, the accuracy of grid-based methods are limited by the resolution, and the memory requirement is generally the highest among the previously discussed methods.

The performance of the nonlinear filters is inherently poor when subjected to noise that corrupts the measurements and is often computationally intensive. It is also worth noting that the discussed filters have not exploited the geometrical aspects of the measurement likelihood. For example, the bearing-sensing has a conical likelihood function extending to infinity, while range-sensing has a spherical likelihood. Within a small local region, these likelihoods can be approximated as cylindrical or planar likelihood function using a degenerate Gaussian.

The degenerate Gaussian has not drawn much attention in the filtering and estimation domain. Sola et al. [2012] reviewed a number of past work that got around representing infinite uncertainty of degenerate Gaussian using some over-parametrisation. However, in some applications (e.g. dense visual reconstruction), the additional memory and computational requirement may be undesirable. In this context, the main interest of this thesis is to construct and apply an efficient estimator exploiting the geometrical aspects of a measurement likelihood. We name this estimator as Minimal Iterative Gaussian Estimator (MIGE). It utilizes a degenerate Gaussian function to approximate a nonlinear likelihood function arising in various sensing problems in target-tracking, localization and SLAM. A degenerate Gaussian function allows infinite uncertainty along some directions. The standard Gaussian parameters of mean and covariance are ill-defined in these functions, and thus we propose a new parametrization method consisting of a minimal set of coefficients for the quadratic terms (which gives rise to the "minimal" term of the MIGE). The estimator also improves the estimation accuracy by iteratively refining the suitable covariance values for the non-degenerate directions (which gives rise to the iterative term of the MIGE). The performance of this estimator is evaluated by applying it to several simulated experiments and complex real-world scenarios with real measure-

ment data, which shows that the new estimator can achieve demonstrably improved accuracy and estimator consistency while requiring the least amount of memory and computational resources, when compared to existing popular filtering methods including those discussed above.

1.2 Objectives

The objective of this thesis is to design an efficient approximate Bayesian estimator and deploy it to some real-world applications. Our focus is on improving the estimator consistency compared to EKF, while requiring low computational and memory resources.

In designing our efficient approximate Bayesian estimator, we consider the following estimation problems:

- **Noisy measurement with outliers**: The sensors used to make measurements of the target are not perfect. There may also be scenarios where the sensor's accuracy is limited by the environment. Some measurement process may also be error prone such that certain measurements have an error significantly larger than the rest of the measurements. These measurements are known as outliers, and need to be handled properly to ensure correct convergence of the estimator. Thus, the estimator should be designed to appropriately handle the measurements' noise and outliers. In this thesis, the inlier (non-outlier) measurement noise is assume to follow zero mean Gaussian distribution, while outlier have large unbounded noise that needs to be discarded. Sensors that were used in the study are: software defined radios (SDRs), camera and inertial measurement unit.

- **Estimator consistency**: The estimator has to provide consistent results. The consistency of a static estimator is defined as the convergence towards the true value with increasing number of measurements. For a dynamic system, the estimator consistency is evaluated using the estimated uncertainty (*i.e.* covariance matrix in Gaussian assumption). More details can be found in Chapter 2.1.9.

- **Computational and memory efficiency**: The computational and memory requirement of the estimator needs to be low that will allow the estimator to be used on resources constrained system, which are increasingly common in robotics applications.

The efficient approximate Bayesian estimator proposed in this thesis has the potential to be applied to many localization and mapping tasks.

1.3 Contributions

This thesis explores solutions to the problem of localization and mapping tasks. The following is a summary of the main contributions of this thesis.

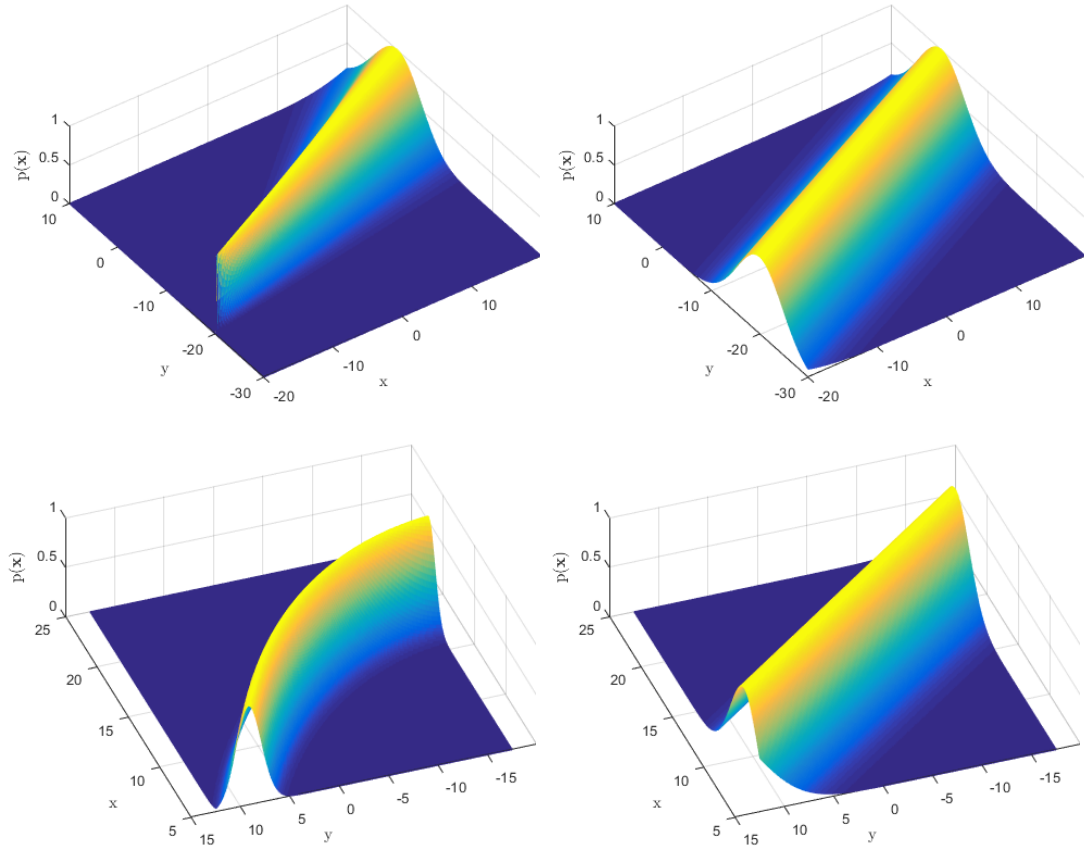

Figure 1.1: Examples of nonlinear likelihood distributions and their approximated likelihood using a degenerate Gaussian. From top left to bottom right: (a) bearing measurement likelihood; (b) approximated bearing measurement likelihood; (c) range measurement likelihood; (d) approximated range measurement likelihood.

- **Minimal Iterative Gaussian Estimator (MIGE)**: We propose a minimal representation, iterative estimator that utilises a single degenerate Gaussian to effectively approximate the measurement likelihood function. The degenerate Gaussian is a Gaussian function where one or more directions may have infinite uncertainty. Examples of degenerate Gaussian used to approximate nonlinear likelihood functions are shown in Figure 1.1. The consistency of the MIGE is improved by ensuring the approximated density encloses the intersection of the prior and measurement density.
- **Accurate recursive TDOA-FDOA localization**: MIGE method is applied to improve TDOA and hybrid TDOA-FDOA localization accuracy. Test results show that the method outperforms existing methods even when the sensor-target geometry is poor.
- **Robust visual SLAM suitable for general motion**: An accurate visual odometry (SLAM front-end) method is proposed that utilizes our Bayes Dense Flow,

Mahalanobis eight-point algorithm, MIGE for scene triangulation, robust scale estimation and pose fusion. An existing robust linear pose graph SLAM is used to further reduce the pose drift in the SLAM back-end. The popular KITTI dataset and our own outdoor UAV dataset are used to demonstrate the performance of our visual odometry and scene reconstruction.

- **Efficient camera trajectory smoothing**: An efficient camera trajectory smoothing method is proposed, which is applied to a video stabilization task. Experimental results show our method achieves smooth motion trajectory with minimal overshoot.

1.4 Publications

All the results in this thesis have been published or are currently under review in refereed journal and conference papers. They are listed in reverse chronological order as follows.

- Y. Ng, J. Kim and C. Yu. A Degenerate Gaussian Representation for Efficient Bayesian Filtering: Part I - Theory, submitted to *IEEE Transactions on Aerospace and Electronic Systems*
- Y. Ng, J. Kim and H. Li. A Degenerate Gaussian Representation for Efficient Bayesian Filtering: Part II - Robust Visual SLAM, submitted to *IEEE Transactions on Aerospace and Electronic Systems*
- Y. Ng, J. Kim, J. Wei and C. Yu. A Degenerate Gaussian Representation for Efficient Bayesian Filtering: Part III - Hybrid TDOA-FDOA Localization, in preparation for submission to *IEEE Transactions on Aerospace and Electronic Systems*
- Y. Ng, J. Wei, C. Yu and J. Kim. Measurement-Wise Recursive TDoA-based Localization Using Local Straight Line Approximation, in *Australian and New Zealand Control Conference*, Gold Coast, Australia, December 2017
- Y. Ng, J. Kim and H. Li. Robust Dense Optical Flow with Uncertainty for Monocular Pose-Graph SLAM, in *Australasian Conference on Robotics and Automation*, Sydney, Australia, December 2017
- Y. Ng, B. Jiang, C. Yu and H. Li. Non-iterative, fast SE(3) path smoothing, in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Daejeon, South Korea, October 2016

1.5 Thesis Structure

The thesis is organized as illustrated in Figure 1.2.

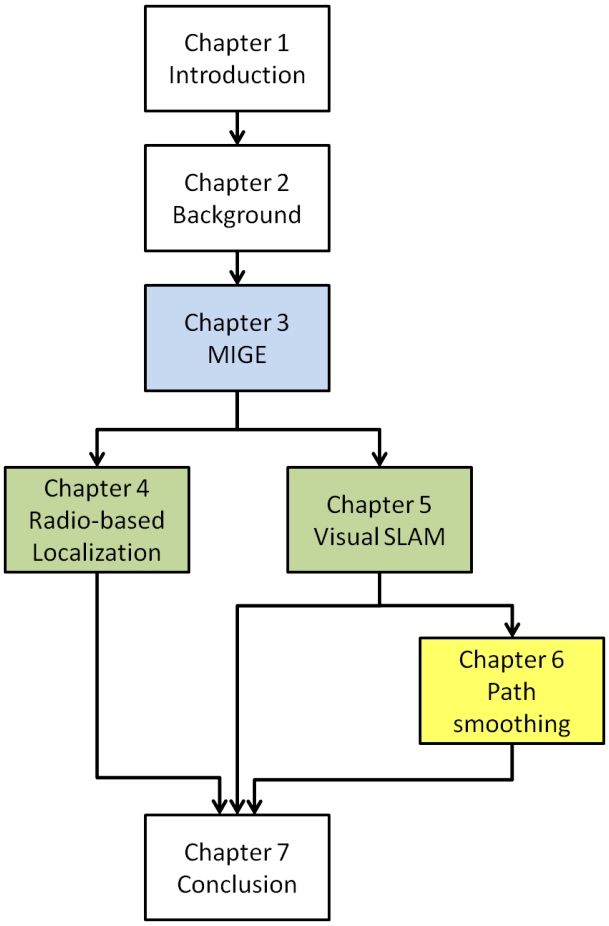

Figure 1.2: Structure of this thesis, where our minimal iterative Gaussian estimator (MIGE) is shown in blue, the two primary application areas of MIGE are highlighted in green, while yellows shows the path smoothing work that is related to the estimated camera motion from visual SLAM.

Chapter 2 reviews the background theories that are used in later chapters. This includes theories in Bayesian filtering, computer vision, radio-based localization and 3D pose representations. For Bayesian filtering, it covers Bayes' theorem, well known non-linear filters, and defines estimator's consistency. Next, theories regarding time-difference-of-arrival and frequency-difference-of-arrival to be used in Chapter 4 are presented. This chapter also covers important theories in computer vision needed in Chapter 5, such as the pinhole camera model, homography, camera projection, feature descriptor, optical flow, epipolar geometry, essential matrix and SLAM. This is then followed by a brief discussion of commonly used 3D pose representation relevant to Chapter 6.

Chapter 3 presents a novel minimal iterative Gaussian estimator (MIGE), showing the intuition and mathematical derivation of the estimator. Particularly, bearing-only and range-only localization will be discussed. Different parts of the MIGE method are presented, which includes the minimal parametrisation, estimator's design to

improve consistency, state prediction and update steps of the estimator. We demonstrate the accuracy and consistency of MIGE by using Monte Carlo simulation for bearing-only and range-only localization.

Chapter 4 discuses our work on passive radio-based localization using time-difference-of-arrival (TDOA) and frequency-difference-of-arrival (FDOA) measurements. The real data experiments have a significant number of outlier measurements, which can be identified and discarded using the simple bounding box method. Simulation and real data experiment shows the improvement in localization accuracy compared to existing methods.

Chapter 5 describes our work on improving the performance of monocular visual SLAM for generic camera motion using the MIGE. The modified DCFlow algorithm [Xu et al., 2017] called Bayes Dense Flow provides accurate estimation of dense optical flow along with 2D uncertainty. A robust visual odometry is presented, which utilises our Bayes Dense Flow, Mahalanobis eight-point algorithm, MIGE-based scene reconstruction, robust scale estimation, and pose fusion. The SLAM back-end relies on an existing robust linear pose graph method [Cheng et al., 2015]. The performance of the resulting monocular visual SLAM is verified using KITTI and outdoor UAV dataset.

Chapter 6 introduces our work on video stabilization application. Given the estimated camera trajectory (*e.g.* using our proposed visual SLAM), a smooth trajectory can be estimated. The difference between the smoothed trajectory and the original trajectory is used to compute a homography to reduce camera shake caused by unstable rotational motion.

Chapter 7 presents the conclusions of the thesis, and discusses future extensions of our work.

Chapter 2

Background

This chapter introduces some preliminaries on the various subjects that were studied. The background theories, concepts and algorithms that form the basis of these works are covered.

2.1 Bayesian filtering

Bayesian filtering techniques were first developed for state estimation in [Ho and Lee, 1964]. A Bayesian filter is a general probabilistic approach in estimating the conditional probability density function (PDF) of an unknown state of a system, using noisy measurements as input.

The state's PDF provides a complete description of the state, which contains the information about the state's mean and uncertainty (spread). This allows new measurements to be incorporated into the estimation in a recursive fashion, which has the benefits of less stringent computational and memory requirements. Thus, the Bayesian filter is often used for real-time state estimation, especially in a resource constraint system.

The state model of any general, nonlinear, discrete time estimation problem can be written as

$$\boldsymbol{x}_k = f_k(\boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}) + \boldsymbol{v}_{k-1}, \tag{2.1}$$

$$\boldsymbol{z}_k = h_k(\boldsymbol{x}_k, \boldsymbol{u}_k) + \boldsymbol{w}_k, \tag{2.2}$$

where $\boldsymbol{x}_k \in \mathbb{R}^{n_x}$ is the state vector at discrete time k; $\boldsymbol{u}_k \in \mathbb{R}^{n_u}$ is the input vector; $\boldsymbol{z}_k \in \mathbb{R}^{n_z}$ is the measurement vector; $f_k(\cdot, \cdot)$ is the known state propagation function; $h_k(\cdot, \cdot)$ is the known measurement function; $\boldsymbol{v}_{k-1}$ and $\boldsymbol{w}_k$ are independent process and measurement noise.

Figure 2.1 shows the graphical representation of the state-space model of the system.

The subsequent subsections will cover important backgrounds on Bayesian filtering.

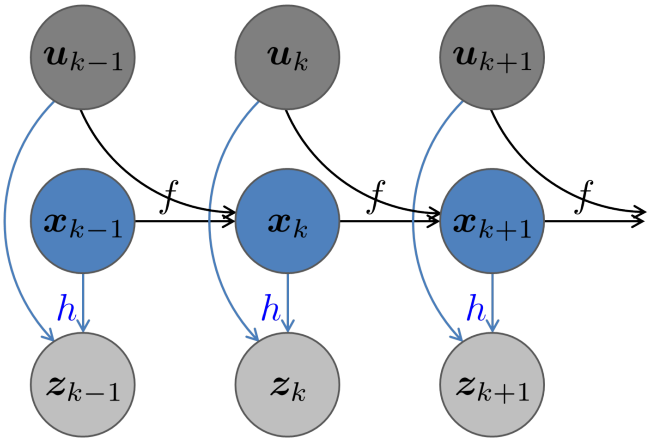

Figure 2.1: Graphical representation of the discrete time state-space model of a general, nonlinear estimation problem (see equation (2.1) and (2.2)). $\boldsymbol{u}_k$ is the input, $\boldsymbol{x}_k$ is the state, and $\boldsymbol{z}_k$ is the measurement at time k.

2.1.1 Bayes' theorem

Arguably the most important theorem in Bayesian filter is the Bayes' theorem, which was discovered by Thomas Bayes and described in a paper published posthumously in 1763 [Bayes and Price, 1763]. Bayesian theory describes the fundamental law governing logical inference. One of the first papers that studies Bayesian approach in an iterative state estimation framework was done by Ho and Lee [1964]. It has also been applied in a number of related research fields, such as Bayesian inference [Bernardo and Smith, 1994; Robert, 1994; Press, 2003], Bayesian learning [Spragins, 1965], optimisation of adaptive systems [Lin and Yau, 1967; Chin et al., 2002], and Monte Carlo methods [Chen, 2003].

Bayes' theorem computes the posterior PDF, $p(\boldsymbol{x}|\boldsymbol{z})$ when given the prior PDF, $p(\boldsymbol{x})$ and likelihood PDF, $p(\boldsymbol{z}|\boldsymbol{x})$. The Bayes' theorem is given as

$$p(\boldsymbol{x}|\boldsymbol{z}) = \frac{p(\boldsymbol{z}|\boldsymbol{x})p(\boldsymbol{x})}{p(\boldsymbol{z})} = \frac{p(\boldsymbol{z}|\boldsymbol{x})p(\boldsymbol{x})}{\int p(\boldsymbol{z}|\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x}}. \tag{2.3}$$

The recursive form of Bayes' theorem is the basis of all sequential Bayesian filtering. The assumptions used in recursive Bayesian filters are:

- The state satisfies a first order Markov process such that $p(\boldsymbol{x}_k|\boldsymbol{X}^{k-1}) = p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$, where $\boldsymbol{X}^k$ is the set of states $\{\boldsymbol{x}_0, \boldsymbol{x}_1, \cdots, \boldsymbol{x}_k\}$. This means that the current state $\boldsymbol{x}_k$ only depends on the previous state $\boldsymbol{x}_{k-1}$, and independent of all other states before it $\{\boldsymbol{x}_0, \boldsymbol{x}_1, \cdots, \boldsymbol{x}_{k-2}\}$.

- The observations are independent of one another such that $p(\boldsymbol{z}_i|\boldsymbol{z}_j) = p(\boldsymbol{z}_i)$, where $i \neq j$.

Theorem 2.1 ([Chen, 2003, pg. 9]). *Let $\boldsymbol{Z}^k$ be the set of measurements $\{\boldsymbol{z}_1, \boldsymbol{z}_2, ..., \boldsymbol{z}_k\}$. The*

recursive form of the Bayes' theorem can be written as

$$p(\boldsymbol{x}|\boldsymbol{Z}^k) = \frac{p(\boldsymbol{z}_k|\boldsymbol{x})p(\boldsymbol{x}|\boldsymbol{Z}^{k-1})}{p(\boldsymbol{z}_k|\boldsymbol{Z}^{k-1})}. \tag{2.4}$$

It can equivalently be written as

$$p(\boldsymbol{x}|\boldsymbol{z}_1, \boldsymbol{z}_2, \cdots, \boldsymbol{z}_k) = c \cdot \left[\prod_{i=1}^{k} p(\boldsymbol{z}_i|\boldsymbol{x})\right] p(\boldsymbol{x}), \tag{2.5}$$

where c is the normalisation constant that makes the sum of the probability density to be equals to one, while $p(\boldsymbol{x})$ is the initial prior for the state $\boldsymbol{x}$.

Bayesian filtering is optimal in the sense that it uses all the available information (expressed by probabilities) to compute the posterior distribution, which contains the complete description of the state being estimated. However, in general, the complete description of the state requires an infinite amount of memory and computational power. Thus, approximate (sub-optimal) methods are used. A common approximation is to assume the PDFs are Gaussian. Some properties of a Gaussian PDF are discussed in the following section.

2.1.2 Gaussian PDF

A Gaussian probability density function (PDF) is a special distribution that can be fully described by a mean, $\boldsymbol{\mu}$ and covariance matrix, $\boldsymbol{P}$, such that

$$\boldsymbol{\mu} = \mathbb{E}\left[\boldsymbol{x}\right], \tag{2.6}$$

$$\boldsymbol{P} = \mathbb{E}\left[(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^T\right]. \tag{2.7}$$

where $\mathbb{E}[\cdot]$ is the expected value function.

A general Gaussian PDF on state vector $\boldsymbol{x}$ can be written as

$$\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{P}) = \frac{1}{\sqrt{\det(2\pi\boldsymbol{P})}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu})^T\boldsymbol{P}^{-1}(\boldsymbol{x} - \boldsymbol{\mu})\right). \tag{2.8}$$

Two examples of Gaussian PDF are shown in Figure 2.2.
Some interesting properties of a Gaussian distribution are: [Gut, 2009]

1. Every covariance matrix $\boldsymbol{P}$ is a square matrix, such that $\boldsymbol{P} \in \mathbb{R}^{n\times n}$.

2. Every covariance matrix $\boldsymbol{P}$ is a symmetrical matrix, such that $\boldsymbol{P} = \boldsymbol{P}^T$.

3. For any symmetric covariance matrix $\boldsymbol{P}$, there exist orthogonal matrix $\boldsymbol{C}$ such that $\boldsymbol{C}^T\boldsymbol{P}\boldsymbol{C} = \boldsymbol{D}$, where $\boldsymbol{D}$ is the diagonal matrix with eigenvalues of $\boldsymbol{P}$.

4. Every covariance matrix $\boldsymbol{P}$ is non-negative definite, such that $\boldsymbol{P} \succeq 0$.

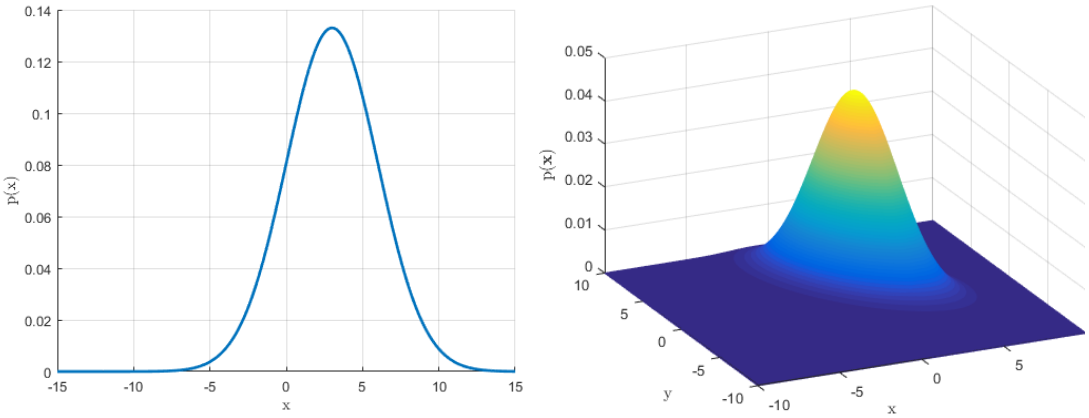

Figure 2.2: Examples of Gaussian PDF. From left to right: (a) 1D Gaussian; (b) 2D Gaussian

5. The exponential power of a Gaussian PDF (equation (2.8)) follows the Chi-square distribution.

   Let $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{P})$, where $\boldsymbol{P}$ is non-singular with $\det(\boldsymbol{P}) > 0$, then $(\boldsymbol{x} - \boldsymbol{\mu})^T \boldsymbol{P}^{-1} (\boldsymbol{x} - \boldsymbol{\mu}) \sim \chi^2(n)$

6. A random variable undergoing matrix transformation is computed as follows.

   Let $\boldsymbol{x} \in \mathbb{R}^n$ be a random variable with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{P}$. Further, let $\boldsymbol{B}$ be a $m \times n$ matrix, a constant vector $b \in \mathbb{R}^n$, and $\boldsymbol{q} = \boldsymbol{B}\boldsymbol{x} + \boldsymbol{b}$. Then, the mean and covariance matrix of $\boldsymbol{q}$ are given by

$$\mathbb{E}[\boldsymbol{q}] = \boldsymbol{B}\boldsymbol{\mu} + \boldsymbol{b}, \tag{2.9}$$

$$\mathbb{E}\left[(\boldsymbol{q} - \mathbb{E}[\boldsymbol{q}])(\boldsymbol{q} - \mathbb{E}[\boldsymbol{q}])^T\right] = \boldsymbol{B}^T \boldsymbol{P} \boldsymbol{B}. \tag{2.10}$$

7. The sum of two Gaussian random vectors is computed as follows.

   Let $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}_x, \boldsymbol{P}_x)$, and $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{\mu}_y, \boldsymbol{P}_y)$. Suppose $\boldsymbol{w} = \boldsymbol{x} + \boldsymbol{y}$. Then, $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{\mu}_x + \boldsymbol{\mu}_y, \boldsymbol{P}_x + \boldsymbol{P}_y)$.

Property 6 and 7 are useful for state prediction (or propagation), where the state can undergo some linear transformations (*e.g.* rotations) and addition of a random variable (*e.g.* translation with noise).

2.1.3 Kalman filter

A well known filter is the Kalman filter [Kalman, 1960, 1963], which is optimal when the estimation problem is linear with Gaussian noise. It is optimal in the sense that the solution is unbiased with minimum variance.

The Kalman filter has two steps, namely prediction and update steps. In the prediction step, the state PDF is propagated from the previous time step to the current time step, which corresponds to equation (2.1). The update step performs a correction on the state PDF based on the measurement taken at the current time step, which corresponds to equation (2.2). In the linear case, the function $f_k(\boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1})$ and $h_k(\boldsymbol{x}_k, \boldsymbol{u}_k)$ in equation (2.1) and (2.2) simplifies to $\boldsymbol{F}_k\boldsymbol{x}_{k-1} + \boldsymbol{B}_k\boldsymbol{u}_{k-1}$ and $\boldsymbol{H}_k\boldsymbol{x}_k + \boldsymbol{C}_k\boldsymbol{u}_k$ respectively.

In the stationary case, where $\boldsymbol{F}_k$, $\boldsymbol{H}_k$, $\boldsymbol{v}_{k-1}$ and $\boldsymbol{w}_k$ are constant, then the Kalman filter is equivalent to the least square Wiener filter [Wiener, 1949]. Originally, the Kalman filter was derived with the orthogonal projection method. Kailath [1970] reformulated the Kalman filter to the well known form using innovation method by Wold and Kolmogorov [Wold, 1938; Kolmogorov et al., 1962].

In practice, most estimation problems are nonlinear and non-Gaussian, which violates the assumptions used by the Kalman filter. Thus, numerous work was done to overcome the limitations of the Kalman filter. Some of the more well known nonlinear filters are discussed in the following sections.

2.1.4 Extended Kalman filter

The most commonly used nonlinear recursive filter is the extended Kalman filter (EKF) [Smith et al., 1962; Sorenson, 1985]. It is closely related to the classical Kalman filter, where the nonlinear problem is locally linearised using the Jacobian

$$\hat{\boldsymbol{F}}_k = \left.\frac{d\boldsymbol{f}_k(\boldsymbol{x}, \boldsymbol{u}_{k-1})}{d\boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}_{k-1|k-1}}, \tag{2.11}$$

$$\hat{\boldsymbol{H}}_k = \left.\frac{d\boldsymbol{h}_k(\boldsymbol{x}, \boldsymbol{u}_k)}{d\boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}_{k|k-1}}. \tag{2.12}$$

Then, the extended Kalman filter follows the classical Kalman filter closely, and is shown in Algorithm 1.

For highly nonlinear, non-Gaussian estimation problems, other filters were proposed. These are explored in subsequent sections.

2.1.5 Unscented Kalman filter

An alternative nonlinear recursive filter that is able to capture the posterior mean and covariance information is the unscented Kalman filter (UKF) proposed by [Julier and Uhlmann, 2004]. [Wan and Van Der Merwe, 2000] also showed the use of the UKF for nonlinear system identification, training of neural network, and dual estimation problem.

The UKF uses deterministic sampling approach, where a minimal set of carefully chosen sample points (sigma points) are used to capture the true mean and covariance of the Gaussian random variable.

Algorithm 1: Extended Kalman filter

Data: Given prior $\{\boldsymbol{x}_0, \boldsymbol{P}_0\}$, state propagation function f_k, measurement function h_k, process covariance matrix $\boldsymbol{Q}_k$, measurement covariance matrix $\boldsymbol{R}_k$

Result: Posterior $\{\boldsymbol{x}_{k|k}, \boldsymbol{P}_{k|k}\}$

initialization;
while *time k is increasing* **do**
  $k = k + 1$;
  Linearise f_k to get $\hat{\boldsymbol{F}}_k$ using (2.11) ;
  Predict state mean, $\boldsymbol{x}_{k|k-1} = f_k(\boldsymbol{x}_{k-1|k-1}, \boldsymbol{u}_k)$;
  Predict state covariance, $\boldsymbol{P}_{k|k-1} = \hat{\boldsymbol{F}}_k \boldsymbol{P}_{k-1|k-1} \hat{\boldsymbol{F}}_k^T + \boldsymbol{Q}_k$;
  if *new measurement arrives* **then**
    Linearise h_k to get $\hat{\boldsymbol{H}}_k$ using (2.12) ;
    Innovation mean, $\boldsymbol{y}_k = \boldsymbol{z}_k - h_k(\boldsymbol{x}_{k|k-1}, \boldsymbol{u}_k)$;
    Innovation covariance, $\boldsymbol{S}_k = \boldsymbol{R}_k + \hat{\boldsymbol{H}}_k \boldsymbol{P}_{k|k-1} \hat{\boldsymbol{H}}_k^T$;
    Kalman gain, $\boldsymbol{K}_k = \boldsymbol{P}_{k|k-1} \hat{\boldsymbol{H}}_k^T {\boldsymbol{S}_k}^{-1}$;
    Posterior mean, $\boldsymbol{x}_{k|k} = \boldsymbol{x}_{k|k-1} + \boldsymbol{K}_k \boldsymbol{y}_k$;
    Posterior covariance, $\boldsymbol{P}_{k|k} = (\boldsymbol{I} - \boldsymbol{K}_k \hat{\boldsymbol{H}}_k) \boldsymbol{P}_{k|k-1}$;
  end
end

Let $\mathbf{x}_{k-1|k-1}$ be the prior mean state, $\mathbf{P}_{k-1|k-1}$ be the covariance matrix, N be the dimension of the state space, and $\left(\sqrt{\mathbf{M}}\right)_i$ is the i^{th} column of the matrix square root of $\mathbf{M}$. Assuming the prior distribution is Gaussian, the sigma points are [Julier and Uhlmann, 2004]

$$
\begin{aligned}
\mathbf{x}_{k-1|k-1}^{(0)} &= \mathbf{x}_{k-1|k-1} \\
W_{k-1|k-1}^{(0)} &= \frac{1}{3} \\
\mathbf{x}_{k-1|k-1}^{(i)} &= \mathbf{x}_{k-1|k-1} + \left(\sqrt{\frac{N}{1 - W_{k-1|k-1}^{(0)}} \mathbf{P}_{k-1|k-1}} \right)_i \\
W_{k-1|k-1}^{(i)} &= \frac{1 - W_{k-1|k-1}^{(0)}}{2N} \\
\mathbf{x}_{k-1|k-1}^{(i+N)} &= \mathbf{x}_{k-1|k-1} - \left(\sqrt{\frac{N}{1 - W_{k-1|k-1}^{(0)}} \mathbf{P}_{k-1|k-1}} \right)_i \\
W_{k-1|k-1}^{(i+N)} &= \frac{1 - W_{k-1|k-1}^{(0)}}{2N}
\end{aligned}
\tag{2.13}
$$

These sample points are then propagated through the nonlinear system dynam-

ics, which can better reflect the true mean and covariance information compared to EKF. The Unscented Kalman filter (UKF) is shown in Algorithm 2.

Algorithm 2: Unscented Kalman filter

Data: Given prior $\{\boldsymbol{x}_0, \boldsymbol{P}_0\}$, state propagation function f_k, measurement function h_k, process covariance matrix $\boldsymbol{Q}_k$, measurement covariance matrix $\boldsymbol{R}_k$
Result: Posterior $\{\boldsymbol{x}_{k|k}, \boldsymbol{P}_{k|k}\}$

initialization;
Compute sigma points $\mathbf{x}_{k-1|k-1}^{(i)}$ and weights $W_{k-1|k-1}^{(i)}$;
while *time k is increasing* **do**
  $k = k + 1$;
  Propagate sigma points by $\mathbf{x}_{k|k-1}^{(i)} = f_k(\mathbf{x}_{k|k-1}^{(i)}, \mathbf{u}_k)$;
  Predict state mean, $\mathbf{x}_{k|k-1} = \sum_{i=0}^{2N} W_{k-1|k-1}^{(i)} \mathbf{x}_{k|k-1}^{(i)}$;
  Predict state covariance,
  $\mathbf{P}_{k|k-1} = \sum_{i=0}^{2N} W_{k-1|k-1}^{(i)} \left(\mathbf{x}_{k|k-1}^{(i)} - \mathbf{x}_{k|k-1}\right) \left(\mathbf{x}_{k|k-1}^{(i)} - \mathbf{x}_{k|k-1}\right)^T$;
  if *new measurement arrives* **then**
    Linearise ... ;
    Innovation mean, ... ;
    Innovation covariance, ... ;
    Posterior mean, ... ;
    Posterior covariance, ...;
  end
end

2.1.6 Grid-based method

If the state is discrete and finite, grid-based methods can produce optimal results. Suppose the discrete state $\boldsymbol{s} \in \mathbb{N}$ consists of a finite number of states $\{1, 2, \cdots, N_s\}$. Let $\boldsymbol{s}_k^i$ denotes the discrete state with index i at time k, and $w_{k|k}^i$ denotes the conditional probability for each $\boldsymbol{s}_k^i$ given measurements up to time k, such that

$$w_{k|k}^i = p(\boldsymbol{s}_k = \boldsymbol{s}^i | \boldsymbol{Z}^k).$$

Then the posterior PDF at time k is represented as

$$p(\boldsymbol{s}_k | \boldsymbol{Z}^k) = \sum_{i=1}^{N_s} w_{k|k}^i \delta(\boldsymbol{s}_k - \boldsymbol{s}_k^i). \tag{2.14}$$

The predicted PDF at time k is represented as

$$p(\boldsymbol{s}_k | \boldsymbol{Z}^{k-1}) = \sum_{i=1}^{N_s} w_{k|k-1}^i \delta(\boldsymbol{s}_k - \boldsymbol{s}_k^i). \tag{2.15}$$

The grid-based method can also be applied to continuous state space, where an approximate grid-based method can be used [Arulampalam et al., 2002]. The grids need to be sufficiently dense to closely approximate the state space. This in turn incurs high computational and memory cost.

For an unbounded state space, some truncation is necessary to make the method tractable. Partitioning of the state space also need to be regular, otherwise the subsequent update may be computationally complex.

Some work on adaptive grid-based method were also performed [Cai et al., 1995; Bao et al., 2014]. However, the computational complexity is still high. Another related work proposed the use of a piece-wise constant function to approximate the density function on linear system [Kramer and Sorenson, 1988].

2.1.7 Gaussian sum filter

Gaussian sum filter [Sorenson and Alspach, 1971; Alspach and Sorenson, 1972] was proposed that requires less memory compared to grid-based method, while allowing close approximation of non-Gaussian probability density. The non-Gaussian PDF is approximated by weighted sum of Gaussian densities, which is also known as Gaussian mixture model (GMM). Let $\boldsymbol{\mu}_i$, $\boldsymbol{P}_i$ and w_i be the mean, covariance matrix and multiplicative weighing for the i^{th} Gaussian density. Then, the PDF represented by the GMM is

$$p(\boldsymbol{x}) = \sum_{i=1}^{N_g} w_i \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_i, \boldsymbol{P}_i), \tag{2.16}$$

where the weighing w_i is strictly positive, and $\sum_{i=1}^{N_g} w_i = 1$

It was shown in [Sorenson and Alspach, 1971] that as N_g approaches infinity and $\boldsymbol{P}_i$ approaches the zero matrix, any general continuous PDF can be accurately approximated. As $\boldsymbol{P}_i$ approaches the zero matrix, $\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_i, \boldsymbol{P}_i)$ approaches the unit impulse function at $\boldsymbol{\mu}_i$.

There are a lot of ways to approximate a given PDF using a sum of Gaussian. Alspach and Sorenson [1972] suggested $\boldsymbol{u}_i$ be chosen in a grid-like fashion around the state space region that are significant, and $\boldsymbol{P}$ is chosen as $b\boldsymbol{I}$ with appropriate value of positive scalar b (depending on grid size). Assuming $\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_i, \boldsymbol{P}_i)$ are normalised, then w_i is chosen such that $\sum_{i=1}^{N_g} w_i = 1$.

Both the prior and measurement likelihood can be approximated using Gaussian sum. The state prediction and update step can then be performed using a bank of EKF. Suppose that the prior and measurement likelihood are approximated by N_p and N_m number of Gaussian densities respectively, then after the update step, there will be $N_p N_m$ Gaussian densities. This causes an exponential increase in the number of Gaussian densities with each measurement update.

Thus, some methods that can reduce the number of Gaussian densities without significantly changing the PDF is vital to ensure computational tractability. Two methods to prevent the number of Gaussian densities to increase exponentially are by

combining close-by densities, and discarding densities with low probability. [Sorenson and Alspach, 1971]

2.1.8 Particle filter

Another popular method that is useful for capturing non-Gaussian PDF with lower computational and memory requirement compared to the grid-based method is the Monte Carlo method. The most well known sequential Monte Carlo method is the particle filter. There are a number of good tutorial papers on particle filter, such as [Arulampalam et al., 2002; Gustafsson, 2010].

The theoretical justification for the Monte Carlo method is similar to Gaussian sum filter, where the true PDF is approximately captured by a large number of random samples (particles). Similar to Gaussian sum filter, the PDF can be captured more accurately as the number of samples N_p increases to infinity. However, large number of samples incurs large computational and memory burden. Thus, a method to efficiently sample (and resample) is very important for Monte Carlo methods.

A common way to obtain and maintain useful samples is sampling-importance resampling (SIR). The steps in SIR can be summarised as follows:

- Select N_p samples $\left\{\boldsymbol{x}^{(i)}\right\}_{i=1}^{N_p}$ from the proposed distribution
- Calculate importance weights w_i for each sample
- Normalise the importance weights such that $\sum_{i=1}^{N_p} w_i = 1$
- Resample from the discrete set $\left\{\boldsymbol{x}^{(i)}\right\}_{i=1}^{N_p}$, where the probability of being selected is proportional to their corresponding w_i

There are numerous ways to resample from the discrete set. For example, multinomial resampling [Rubin, 1987; Efron and Tibshirani, 1993], Residual resampling [Liu and RongChen, 1998; Whitley, 1994], stratified resampling [Kitagawa, 1996], and systematic or deterministic resampling [Kitagawa, 1996].

An example of the multinomial resampling is shown in Figure 2.3. The basic resampling steps can be summarised as follows:

- Given discrete samples and its corresponding importance weights $\{\boldsymbol{x}^{(i)}, w_i\}$. Normalise importance weights such that $\sum_{i=1}^{N_p} w_i = 1$, and compute the cumulative sum of importance weights $\{s_i\}$.
- Select N_p random samples $\{u_i\}$, where different methods of selection were proposed.
- For each u_i, select the sample $\boldsymbol{x}^{(i)}$ that corresponds to the s_i such that $s_{i-1} < u_i \leq s_i$.
- Reset the importance weights $w_i = 1/N_p$.

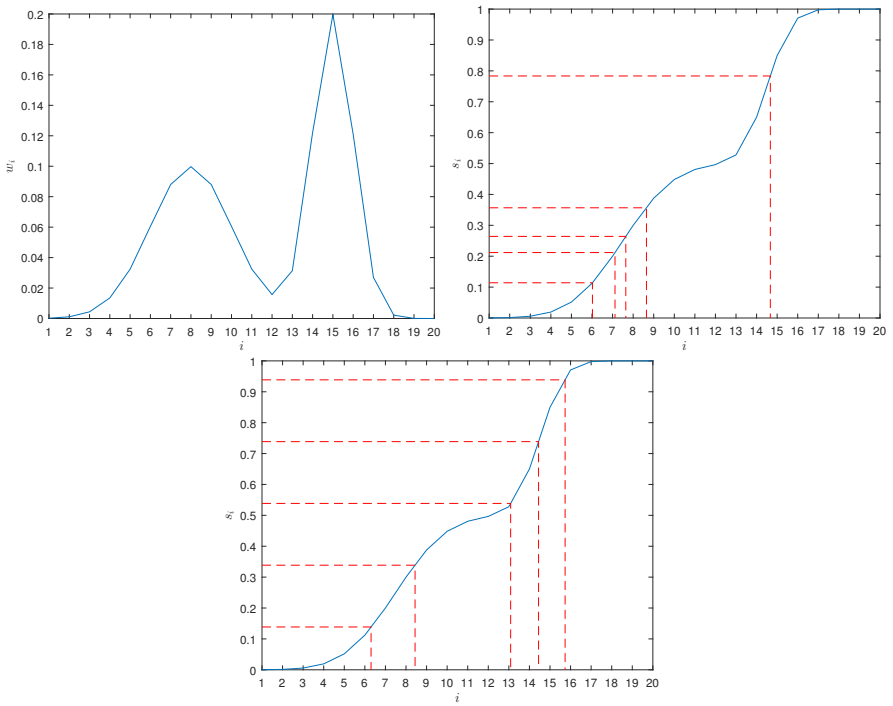

Figure 2.3: Examples of multinomial sampling. From top left to bottom: (a) an arbitrary importance weights for sample $\boldsymbol{x}^{(i)}$; (b) cumulative sum of importance weights showing multinomial resampling of 5 samples ($\{\boldsymbol{x}^{(6)}, \boldsymbol{x}^{(8)}, \boldsymbol{x}^{(8)}, \boldsymbol{x}^{(9)}, \boldsymbol{x}^{(15)}\}$); (c) cumulative sum of importance weights showing deterministic resampling of 5 samples ($\{\boldsymbol{x}^{(7)}, \boldsymbol{x}^{(9)}, \boldsymbol{x}^{(14)}, \boldsymbol{x}^{(15)}, \boldsymbol{x}^{(16)}\}$).

For multinomial resampling method, the $\{u_i\}$ is randomly selected from the uniform distribution between $(0, 1]$. In residual resampling, for each w_i, $n_i = \lfloor w_i N_p \rfloor$ samples of $\boldsymbol{x}^{(i)}$ were kept, the new weight $\tilde{w}_i = w_i - n_i / N_p$ is used in multinomial resampling step to select the remaining samples. In Stratified resampling, $\{u_i\}$ were selected from within a uniformly partitioned region between $(0, 1]$ such that each u_i is a randomly selected from the uniform distribution between $\left(\frac{i-1}{N_p}, \frac{i}{N_p}\right]$. Deterministic resampling method select $u_i = \frac{i-\alpha-1}{N_p}$, where α is randomly selected value from the uniform distribution between $(0, 1]$.

Due to the regularity of the selection of u_i, the deterministic resampling was shown to be able to replicate all samples, where the variance between different selected samples is the smallest. The computational complexity is also low, at $\mathcal{O}(N_p)$ [Chen, 2003]. Thus, this resampling method is selected for the SIR particle filter in our experimental comparison.

2.1.9 Estimator's Consistency

The consistency of an estimator is defined as the convergence of the estimation towards the true value as the number of measurements approaches infinity. The fol-

lowing theorem provides a more formal definition.

Theorem 2.2 ([Ibragimov and Has'Minskii, 2013, pg. 30])**.** *An estimator is said to be consistent if for some value of ϵ, the state estimate up to measurement t, $\tilde{\boldsymbol{x}}_t$ converges to the true state $\boldsymbol{x}$ in probability, such that*

$$\lim_{t\to\infty} p(|\tilde{\boldsymbol{x}}_t - \boldsymbol{x}| > \epsilon) = 0. \tag{2.17}$$

The consistency described in Theorem 2.2 is an asymptotic property of an estimator. However, this only applies to estimation of a constant, stationary parameter. The finite sample consistency for a filter in dynamical system is evaluated as follows.

Theorem 2.3 ([Bar-Shalom et al., 2004, pg. 233])**.** *Under the Gaussian assumption, the state error should be acceptable as zero mean and have magnitude corresponding to the estimated state covariance.*

Suppose the state estimation up to measurement t is $\tilde{\boldsymbol{x}}_t$, true state is $\boldsymbol{x}_t$, and the estimated covariance matrix is $\boldsymbol{P}$. Then, the average normalised estimation error squared (NEES) is

$$\mathbb{E}\left[(\tilde{\boldsymbol{x}}_t - \boldsymbol{x}_t)^T \boldsymbol{P}^{-1} (\tilde{\boldsymbol{x}}_t - \boldsymbol{x}_t)\right] = n_x, \tag{2.18}$$

where n_x is the dimension of the state vector. Equation (2.18) is a property of a chi-square distribution (see chapter 2.1.2, property 5).

Note that a biased estimator may still be consistent. For a static estimator, if (2.17) holds, a biased estimator with the bias magnitude less than or equals to ϵ is consistent. For dynamic estimator, if $\boldsymbol{P} = \bar{\boldsymbol{P}} + \text{diag}(\boldsymbol{b}^2)$, where $\bar{\boldsymbol{P}}$ is the covariance matrix of the unbiased estimator and $\boldsymbol{b}$ is the bias vector. Then, the biased estimator with estimated covariance matrix $\boldsymbol{P}$ is still consistent.

2.2 Radio-based localization

A number of strategically placed radio sensors can be used to localize an emitter location. If the exact time of signal emission is known, the time-of-arrival (TOA) or time of flight can be measured. This provides a distance measurement from the sensor, which constrains the possible location of the emitter onto any point on the surface of a sphere. However, this usually requires the emitter to send the transmission time to the sensor, where the clock between emitter and sensors need to be synchronised. These limit the usefulness of such technique.

In some application areas, this cooperative behaviour between the sensor and target is not possible or undesirable. Thus, other passive measurements are explored. The most common passive measurements for radio-based localization are time-difference-of-arrival (TDOA) and frequency-difference-of-arrival (FDOA).

Suppose $k = 1, 2, ..., N$ denotes the time index when the measurements are made. The radio emitter is located at the unknown location $\boldsymbol{p_k} = [x_k, y_k, z_k]^T$. The known coordinates of the sensor i and sensor j are $\boldsymbol{s}_{i,k} = [x_{i,k}, y_{i,k}, z_{i,k}]^T$ and $\boldsymbol{s}_{j,k} = [x_{j,k}, y_{j,k}, z_{j,k}]^T$

respectively. The relative velocity between the emitter and sensors i and j are represented as $\boldsymbol{v}_{i,k}$ and $\boldsymbol{v}_{j,k}$ respectively. The displacement vector between the emitter and the sensors are thus denoted as $\boldsymbol{d}_{i,k} = \boldsymbol{p}_k - \boldsymbol{s}_{i,k}$ and $\boldsymbol{d}_{j,k} = \boldsymbol{p}_k - \boldsymbol{s}_{j,k}$. The distance between the emitter and sensors are then denoted as $r_{i,k} = \sqrt{(\boldsymbol{d}_{i,k})^T \boldsymbol{d}_{i,k}}$ and $r_{j,k} = \sqrt{(\boldsymbol{d}_{j,k})^T \boldsymbol{d}_{j,k}}$. The theory behind TDOA and FDOA are presented in the following sections.

2.2.1 TDOA

Time-difference-of-arrival as the name suggests, measures the difference in time the same signal arrives at different sensors.

The time-difference-of-arrival (TDOA) equation is represented as

$$\tau_{ij,k} = \frac{1}{c}(r_{i,k} - r_{j,k}), \tag{2.19}$$

where c is the signal propagation speed (speed of light for radio signal).

Each TDOA measurement provides a hyperboloid constraint on the possible emitter location, and the minimum number of sensors required to obtain a unique emitter location estimate in 3D is 5.

2.2.2 FDOA

Frequency-difference-of-arrival (FDOA) measurement is caused by a phenomenon called Doppler shift. It is the stretching or compressing of a waveform, due to the non-zero relative velocity between the emitter and sensor. Similar to TDOA, without knowledge of the exact transmission frequency, the differential Doppler or FDOA can be measured.

The frequency-difference-of-arrival (FDOA) equation is represented as

$$\nu_{ij,k} = \frac{f_c}{c}((\boldsymbol{v}_{i,k})^T \boldsymbol{u}_{i,k} - (\boldsymbol{v}_{j,k})^T \boldsymbol{u}_{j,k}), \tag{2.20}$$

where c is the signal propagation speed (speed of light for radio signal), f_c is the carrier frequency and

$$\boldsymbol{u}_{i,k} = \frac{\boldsymbol{d}_{i,k}}{r_{i,k}}, \quad \boldsymbol{u}_{j,k} = \frac{\boldsymbol{d}_{j,k}}{r_{j,k}}. \tag{2.21}$$

2.3 Monocular visual SLAM

In the area of visual SLAM, the following section covers some basic background knowledge of the research. "SLAM" refers to simultaneous localization and mapping, where both the sensor pose and the map points are jointly estimated. "Visual SLAM" refers to a branch in SLAM research that focuses on the use of visual sensor (*i.e.* camera) for the SLAM task, while "monocular" refers to the use of a single RGB

colour camera in contrast to stereo (two camera), multi-camera (more than two cameras), RGBD (colour and depth camera) and other visual systems. Some important theories in visual SLAM are covered in the following sections.

2.3.1 Pinhole camera model

The simplest and most commonly used camera model is the pinhole camera model, which assumes linear projection of 3D scene onto the 2D image plane. This is illustrated in Figure 2.4.

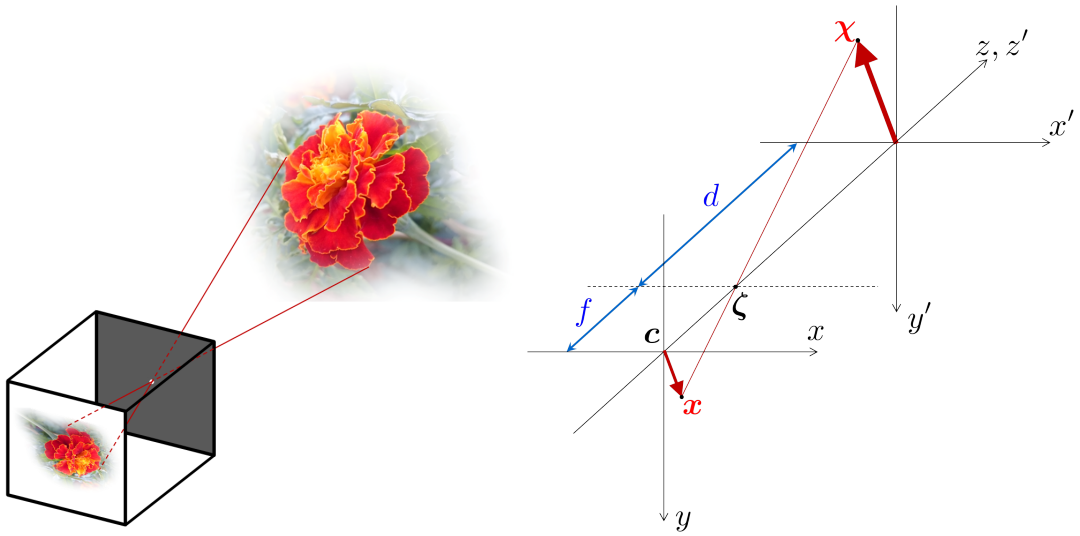

Figure 2.4: Illustration of pinhole camera model. From left to right: (a) An illustrative example of 3D scene projection onto 2D image plane; (b) The geometrical relationship showing the projection of a 3D scene point $\boldsymbol{\chi}$ (in x'–y' plane) to image point $\boldsymbol{x}$ (in x–y or image plane), where $\boldsymbol{\zeta}$ is the optical centre or the camera centre, d is the depth of the scene point from $\boldsymbol{\zeta}$, f is the focal length, z is the principal axis of the camera, $\boldsymbol{c}$ is the image centre.

From Figure 2.4(b), it is clear that the values of $\boldsymbol{x}$ can be computed as follows. Let $\boldsymbol{x} = [x, y]^T$ and $\boldsymbol{\chi} = [X, Y, Z]^T$, then, $Z = d$, and

$$\begin{bmatrix} x \\ y \end{bmatrix} = -\frac{f}{Z} \begin{bmatrix} X \\ Y \end{bmatrix}. \tag{2.22}$$

The computation can be simplified by using a virtual image plane, which is placed between the pinhole centre $\boldsymbol{O}$ and the 3D scene point. The virtual image plane is shifted by a distance equals to f in the z direction from $\boldsymbol{O}$. This makes the projected virtual image to be in the same orientation (not rotated) as the scene point. Then, the relationship between the virtual image point $\boldsymbol{p}_v = [x_v, y_v]^T$ and the scene point $\boldsymbol{\chi}$ is

$$\begin{bmatrix} x_v \\ y_v \end{bmatrix} = \frac{f}{Z} \begin{bmatrix} X \\ Y \end{bmatrix}. \tag{2.23}$$

2.3.2 Homogeneous coordinate

A useful coordinate representation is the homogeneous coordinates, where any scalar multiple of the homogeneous coordinates represents the same vector. Let v be a vector of appropriate dimension, and s be a non-zero scalar, then the homogeneous coordinate $\bar{v}$ can be written as

$$\bar{v} = \begin{bmatrix} v \\ 1 \end{bmatrix} \equiv \begin{bmatrix} sv \\ s \end{bmatrix} \tag{2.24}$$

Theorem 2.4 ([Hartley and Zisserman, 2003, pg. 27])**.** *Suppose the homogeneous vector represents the 2D point $\bar{\boldsymbol{x}} = [x, y, 1]^T$ and 2D line $ax + by + c = 0$ is parametrised as $\boldsymbol{l} = [a, b, c]^T$. Then, if the dot product of the two homogeneous vectors is equals to zero, such that*

$$\bar{\boldsymbol{x}}.\boldsymbol{l} = \bar{\boldsymbol{x}}^T\boldsymbol{l} = \boldsymbol{l}^T\bar{\boldsymbol{x}} = 0.$$

Then, point $\bar{\boldsymbol{x}}$ lies on the line $\boldsymbol{l}$.

Theorem 2.5 ([Hartley and Zisserman, 2003, pg. 27])**.** *Given two 2D lines $\boldsymbol{l} = [a, b, c]^T$ and $\boldsymbol{l}' = [a', b', c']^T$, the intersection point of the lines $\bar{\boldsymbol{x}}$ can be calculated using vector cross-product, such that $\bar{\boldsymbol{x}} = \boldsymbol{l} \times \boldsymbol{l}'$.*

Theorem 2.6 ([Hartley and Zisserman, 2003, pg. 28])**.** *Given two 2D homogeneous points $\bar{\boldsymbol{x}}$ and $\bar{\boldsymbol{x}}'$, the 2D line passing through both points is $\boldsymbol{l} = \bar{\boldsymbol{x}} \times \bar{\boldsymbol{x}}'$.*

2.3.3 2D Homography

2D projective geometry is the study of projective plane's properties that are invariant under a group of transformation known as *projectivities*. The following definition and theorem are provided in [Hartley and Zisserman, 2003].

Definition 2.1. *Projectivity is an invertible mapping h that maps points in plane $\mathbb{P}^2$ to itself, where collinearity of points is preserved.*

Theorem 2.7 ([Hartley and Zisserman, 2003, pg. 33])**.** *A plane-to-plane mapping $h : \mathbb{P}^2 \to \mathbb{P}^2$ is a projectivity if and only if it is a linear mapping by a non-singular matrix $\boldsymbol{H}$ such that for any homogeneous point $\boldsymbol{x}$ in $\mathbb{P}^2$, $h(\boldsymbol{x}) = \boldsymbol{H}\boldsymbol{x}$.*

The projective transformation is also known as *homography*, where the non-singular matrix $\boldsymbol{H}$ is known as the homography matrix. Due to the use of homogeneous coordinate, non-zero multiplicative scaling does not change the location of the 2D point, such that $\boldsymbol{H}\boldsymbol{x}_i \equiv s\boldsymbol{H}\boldsymbol{x}_i$. Thus, the 3×3 homography matrix has eight degrees of freedom, and has rank 3.

Let $\boldsymbol{x} = [x, y, 1]^T$ be the original homogeneous coordinate of a 2D point, $\boldsymbol{x}' = [x', y', 1]^T$ be the transformed point, the projective transformation or homography on a 2D point can be written as follows

$$\lambda \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}. \tag{2.25}$$

The homography matrix can be computed from point correspondences between two images. From (2.25), we can expand and normalise the homogeneous coordinates as

$$x' = \frac{\lambda x'}{\lambda} = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}},$$
$$y' = \frac{\lambda y'}{\lambda} = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}}.$$

It can be observed that each point correspondences provide two equations for the elements of $\boldsymbol{H}$, such that

$$x'(h_{31}x + h_{32}y + h_{33}) = h_{11}x + h_{12}y + h_{13},$$
$$y'(h_{31}x + h_{32}y + h_{33}) = h_{21}x + h_{22}y + h_{23}.$$

It can be rewritten as

$$\begin{bmatrix} x & y & 1 & 0 & 0 & 0 & -xx' & -yx' & -x' \\ 0 & 0 & 0 & x & y & 1 & -xy' & -yy' & -y' \end{bmatrix} \boldsymbol{h} = \boldsymbol{0}, \tag{2.26}$$

where $\boldsymbol{h} = \begin{bmatrix} h_{11} & h_{12} & h_{13} & h_{21} & h_{22} & h_{23} & h_{31} & h_{32} & h_{33} \end{bmatrix}^T$.

These are linear equations of the elements of $\boldsymbol{H}$, and four point correspondences is sufficient to solve the homography matrix up to a multiplicative scaling factor. However, this minimum of four points correspondences assume that no three points are collinear. More sophisticated method of computing the homography matrix can be found in chapter 4 of [Hartley and Zisserman, 2003]. Homography can be used for projective distortion removal, where an example is shown in Figure 2.5.

A less general transformation of the homography is the affine transformation, where $[h_{31}, h_{32}, h_{33}] = [0, 0, 1]$, which has 6 degrees of freedom. If $\begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{bmatrix} = s\boldsymbol{R}$, where $\boldsymbol{R}$ is the 2D rotation matrix, it is known as similarity transformation, which has 4 degrees of freedom. If $s = 1$ in similarity transformation, it becomes Euclidean transformation. The invariant properties under different transformation are detailed in [Hartley and Zisserman, 2003].

2.3.4 3D to 2D camera projection

For a camera at arbitrary pose, the homogeneous 3D scene point $\bar{\boldsymbol{\chi}}_{(4\times1)}$ and the homogeneous 2D image point $\bar{\boldsymbol{x}}_{(3\times1)}$ is related by a projective transformation

$$\lambda\bar{\boldsymbol{x}}_{(3\times1)} = \boldsymbol{P}_{(3\times4)}\bar{\boldsymbol{\chi}}_{(4\times1)}. \tag{2.27}$$

where the subscripts in parenthesis denote the dimension of the corresponding matrix and vectors, and λ is a scalar that makes the third element of $\bar{\boldsymbol{x}}_{(3\times1)}$ to be equals to one.

The matrix $\boldsymbol{P}_{(3\times4)}$ in general has rank 3, and 11 degrees of freedom. The intrinsic

Figure 2.5: An example of removing projective distortion (of blue building) from a perspective image of a plane. From top to bottom: (a) original image; (b) homography transformed image. Notice that the windows of the blue building has orthogonal edges after the homography transformation, but other objects not on the same plane may look distorted. There are also black border on the left of the transformed image due to missing information (outside of original image boundary).

properties of the camera (5 degrees of freedom), namely the focal length, camera skew and image centre may be extracted into a 3×3 matrix by a simple decomposition. The remaining 6 degrees of freedom are the extrinsic parameter of the camera, which contains the 3D rotation and translation. The general matrix $\boldsymbol{P}_{(3\times4)}$ for a pinhole camera can then be written as

$$\begin{aligned} \boldsymbol{P}_{(3\times4)} &= \boldsymbol{K}\left[\boldsymbol{I}_{(3\times3)} \;\; \boldsymbol{0}_{(3\times1)}\right]\boldsymbol{T}_{ex} \\ &= \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} \boldsymbol{R}_{(3\times3)} & \boldsymbol{t}_{(3\times1)} \\ \boldsymbol{0}_{(1\times3)} & 1 \end{bmatrix}, \end{aligned} \tag{2.28}$$

where $\boldsymbol{K}$ is known as the intrinsic camera parameter matrix, $\boldsymbol{T}_{ex}$ is the extrinsic camera parameter matrix, f_x and f_y are the focal lengths, s is the camera skew parameter,

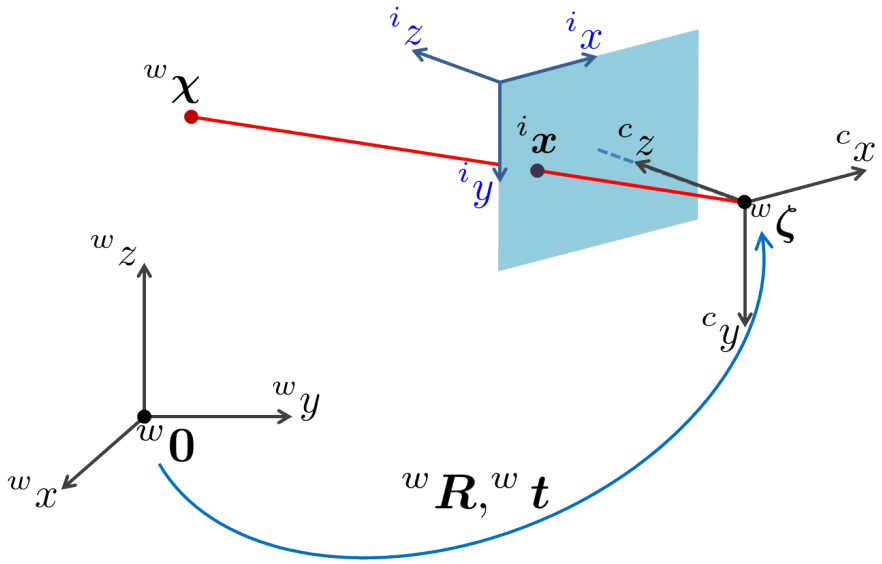

Figure 2.6: Illustrative figure showing the projection of a 3D point ${}^{w}\boldsymbol{\chi}$ to image point ${}^{i}\boldsymbol{x}$ on the image plane (blue shaded region). The superscript before each variables represents the coordinate frame they are defined, where w represents the world coordinate frame, c is the camera frame, and i is the image frame.

$[c_x, c_y]$ are the camera centre coordinate in the image plane, $\boldsymbol{R}_{(3\times3)}$ is the 3D rotation matrix, and $\boldsymbol{t}_{(3\times1)}$ is the 3D translation vector.

Figure 2.6 shows the projection of a 3D scene point onto the image plane. The ${}^{w}\boldsymbol{R}$ and ${}^{w}\boldsymbol{t}$ describes the pose of the camera with respect to the world coordinate frame. We know that fixing the 3D scene points while moving the camera, and fixing the camera pose while moving the 3D scene points (in an inverse rigid body transformation) will produce the same image point location. This means ${}^{w}\boldsymbol{R}$ and ${}^{w}\boldsymbol{t}$ is the inverse transformation of $\boldsymbol{T}_{ex}$, such that

$$\begin{aligned}\begin{bmatrix} {}^{w}\boldsymbol{R} & {}^{w}\boldsymbol{t} \\ \mathbf{0}_{(1\times3)} & 1 \end{bmatrix} &= (\boldsymbol{T}_{ex})^{-1} \\ &= \begin{bmatrix} \boldsymbol{R}_{(3\times3)}{}^{T} & -\boldsymbol{R}_{(3\times3)}{}^{T}\boldsymbol{t} \\ \mathbf{0}_{(1\times3)} & 1 \end{bmatrix}.\end{aligned} \tag{2.29}$$

Note that the example shown in Figure 2.4(b) assumes that both the camera and world coordinate (for 3D scene points) frames are aligned. The intrinsic parameters of the camera are also assumed to have equal focal length in both x and y direction ($f_x = f_y = f$), with zero skew and translation. In such a case, the matrix $\boldsymbol{P}_{(3\times4)}$ can be expressed as

$$\boldsymbol{P}_{(3\times4)} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{2.30}$$

The focal lengths f_x and f_y may not be equal when the CCD sensor representing

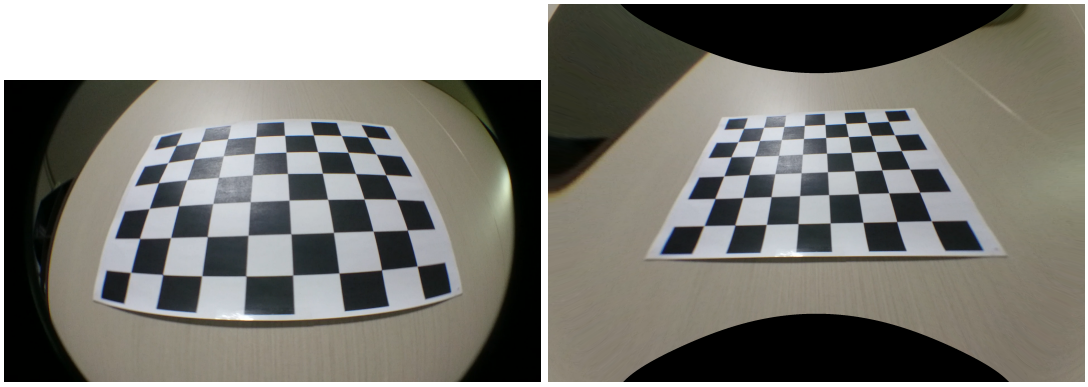

Figure 2.7: An example of radial distortion. From left to right: (a) Input image with fish-eye lens where the radial distortion is visible, (b) radial distortion corrected image where straight lines appear straight. The black region at the top and bottom of the radial distortion corrected image are areas with missing information (outside of original image boundary).

the image pixels are rectangular instead of square-shaped. The skew parameter s may be non-zero when the x and y axis of the CCD array are not perpendicular, or when a "picture of a picture" process occurs. [Hartley and Zisserman, 2003]

Two classes of camera matrix are finite camera, and camera with centre at infinity. An example of the second type is an affine camera, which represents parallel projection. In my work, I focus on finite camera, where the focal length is finite.

Theorem 2.8 ([Hartley and Zisserman, 2003, pg. 158])**.** *Suppose $\boldsymbol{P}_{(3\times4)}$ is the camera projection matrix that maps homogeneous 3D scene point to homogeneous 2D image point. Then, the right null space of $\boldsymbol{P}_{(3\times4)}$ is the homogeneous coordinate of the camera centre $\boldsymbol{\zeta}$.*

There is another phenomenon that affects images taken by a camera, which is not captured by the linear projective transformation in (2.27). It is due to the use of optical lens to focus light rays in a camera. Compared to pinhole camera, the use of optical lens allows more light to be captured, but introduces radial distortion in the image.

The effect of radial distortion is greater the farther away from the distortion centre, and it is more obvious as the focal length decreases. For example, radial distortion is very evident in wide-angle photography. An example of radial distortion is shown in Figure 2.7. It can also be observed that the projective distortion is not removed by the radial distortion correction, where an object further away from the camera will still appear smaller than close by object.

Radial distortion can be modelled as

$$\begin{bmatrix} x_d \\ y_d \end{bmatrix} = \mathcal{L}(r) \begin{bmatrix} x \\ y \end{bmatrix}, \tag{2.31}$$

where $[x_d, y_d]^T$ is the measured image coordinates (after radial distortion), $[x, y]^T$ is the ideal image coordinates, r is the radius from the distortion centre, and $\mathcal{L}(r)$ is the

distortion factor, which is a function of r.

The distortion factor is modelled by approximation of an arbitrary function by Taylor series, such that $\mathcal{L}(r) = 1 + \kappa_1 r + \kappa_2 r^2 + \kappa_3 r^3 + \cdots$. The coefficients and distortion centre $\{\kappa_1, \kappa_2, ..., x_c, y_c\}$ are considered part of the camera calibration parameters, and are often estimated together with $\boldsymbol{K}$ in (2.28).

It is often not necessary to warp the image to correct for radial distortion. This is because image warping will distort noise model and may introduce aliasing effect to the image. Instead, the feature position may be corrected by applying appropriate transformation according to (2.31). In our work, we assume that the camera intrinsic and radial distortion parameters are calibrated accurately.

2.3.5 Feature descriptor and matching

In most computer vision tasks, we need to find the location of features and the corresponding matching features in two or more images. For example, a minimum of four feature correspondences is required to compute the homography matrix. Features of interest are first detected independently in each image. Then, each feature is matched to features in other images using proximity and local visual similarity. There may be more than one visually similar features in other images for a particular feature. The best matching feature is selected based on the similarity metric chosen.

There are numerous ways to detect and describe an image feature. The simplest one is the Harris corner detector and descriptor proposed by Harris and Stephens [1988]. Harris corner detector can localize corners in the image at subpixel accuracy, typically with an error less than 1 pixel [Schmid and Zisserman, 1998].

Other commonly used feature descriptors are Scale-Invariant Feature Transform (SIFT) [Lowe, 2004], Speeded Up Robust Feature (SURF) [Bay et al., 2006] and oriented rotated BRIEF (ORB) [Xu et al., 2012]. These features are more robust against scale changes compared to the simple Harris corner. It can be observed that the same object appears larger in a perspective image. Thus, the scale-invariant property is important when matching features where the translational magnitude is large.

More recently, with the increase in popularity of artificial intelligence and neural network research, the use of convolutional neural network (CNN) trained features are becoming the norm [Zhou et al., 2014; Jia et al., 2014; Girshick et al., 2014]. These features can be seen as a generalisation of the hand-crafted feature descriptors previously mentioned, where the weights for different visual cues are learned from a large dataset.

The matching features can be determined by using a predefined feature similarity metric. Some example of feature similarity metric are normalised cross-correlation (CC), squared sum of intensity difference (SSD), and the dot product between feature vector. CC is robust against affine mapping of the intensity value (*i.e.* $I_1 = \alpha I_2 + \beta$, scaling and offset), while SSD is more sensitive, and dot product is computationally efficient.

2.3.6 Optical flow

Unlike the sparse nature of the feature descriptor and matching discussed in previous section, optical flow is the dense motion field of image pixels. This means the optical flow at each pixels describes the direction and magnitude of the corresponding pixel's motion between two images. An example of computed optical flow is shown in Figure 2.8.

Figure 2.8: An example of optical flow computed using *MATLAB*'s Farneback optical flow function. The input images are taken from KITTI odometry dataset [Geiger et al., 2012], where the camera is moving forward. The blue arrows show the direction and magnitude of the pixels' motion between two consecutive frames. Note that the areas with no texture (*e.g.* walls of building) has no optical flow due to the difficulty in computing optical flow within those regions.

Since the well-known work [Horn and Schunck, 1981; Lucas et al., 1981], recent developments of optical flow are focused on handling large displacement, occlusion, illumination changes and reducing computational complexity. Another common challenge of optical flow is the aperture problem, where the pixel's motion perpendicular to the intensity gradient cannot be estimated accurately.

Classical optical flow algorithm optimises a cost function of the form

$$C(\boldsymbol{f}) = C_{data}(\boldsymbol{f}) + \lambda C_{reg}(\boldsymbol{f}), \tag{2.32}$$

where $\boldsymbol{f}$ is the computed optical flow, C_{data} is the data term that penalises visually dissimilar pixel, C_{reg} is the regularisation term that encourages spatially smooth variation of optical flow field, while λ controls the trade-off between the two terms.

There are a number of possible data terms C_{data} that can be used. For example, similarity in terms of brightness [Horn and Schunck, 1981], gradient [Brox et al., 2004], affine intensity and blur [Seitz and Baker, 2009], photometrically invariant features [Liu et al., 2011].

The regularisation term C_{reg} was first proposed by [Horn and Schunck, 1981], where a homogeneous regularisation was applied. This does not respects flow discontinuity where different objects may have different optical flow direction and magnitude. Since then, image edge driven [Lefebure et al., 1999; Nagel and Enkelmann,

1986], flow driven [Black and Anandan, 1991], median filtering [Sun et al., 2010] among others had been proposed to smooth the computed optical flow.

2.3.7 RANSAC

The most widely used assumption in parameter estimation is that the error follows a Gaussian distribution. However, in practical scenario, this is not valid due to the presence of outliers. For example, incorrect feature matches may produce an error distribution with a long tail, where the probability of deviation from the mean value does not decreases to zero exponentially fast.

The percentage of outliers may also be high such that using large number of measurement does not guarantee convergence to the correct solution. Thus, outliers will significantly degrade the performance of an estimator if they are not handled properly. One method commonly used in computer vision to reduce the effect of outliers is Random Sample Consensus (RANSAC) by [Fischler and Bolles, 1981].

The main idea of RANSAC is to identify inliers (non-outliers), so that the estimator only uses the inliers for parameter estimation. Outliers that violate the Gaussian assumption are discarded and not used. First, a minimum number of measurements are randomly selected to compute an estimate. The number of support (inliers) is computed, where the inliers have an error less than a threshold distance from the estimate. The same steps are repeated N times and the estimate with the highest number of support is used.

This is a robust estimation technique such that it is robust against a modest amount of outliers with potentially unmodelled error distribution. The general steps of RANSAC are shown in Algorithm 3.

Algorithm 3: General Random Sample Consensus (RANSAC)

Data: Measurements with outliers (*e.g.* image point correspondences)
Result: Robust parameter estimate
1 initialization;
2 **while** $k < N$ **do**
3 $k = k + 1$;
4 Randomly select a minimal number of measurements from the input measurement set;
5 Estimate parameter using the selected measurements;
6 Compute the distance (error) of all measurements to the estimate;
7 Identify consensus set (inliers) where the error is less than threshold t;
8 **if** *number of inliers* > *previous largest number of inliers* **then**
9 Store the inlier set S_i;
10 **end**
11 **end**
12 Using the inlier set S_i, recompute the parameter;

The distance used to identify inliers (line 5 of Algorithm 3) follows the assumed

measurement error model. Under the common Gaussian error assumption, the squared Mahalanobis distance follows $\chi^2(n)$, the chi-square distribution with n degrees of freedom. The threshold on Mahalanobis distance can be selected based on the desired confidence interval with the a-priori covariance matrix. For example, in a 2D case, a threshold of 3χ ensures 98.89% of inliers are maintained, while there is a 1.11% chance inliers are being discarded. In the literature, the measurement error is commonly assumed to be isotropic and homogeneous Gaussian, which simplifies the Mahalanobis distance to scaled Euclidean distance. Other distance metric like reprojection error may also be used.

There is also an inherent trade-off between maintaining a large percentage of inliers and the ability to effectively remove outliers. In the extreme case, a Mahalanobis distance threshold of infinity can maintain all inlier measurements, but all outlier measurements are also preserved.

The number of iterations N should be high, while still being computationally tractable. The value of N can be chosen such that with a probability of p (usually 0.99), one of the selected measurement set is free of outliers. Suppose the percentage of inliers among all the measurement is ϵ (equivalent to probability of selecting an inlier), and n be the minimal required number of measurements to compute a solution, then

$$N = \frac{\log(1-p)}{\log(1-\epsilon^n)}. \tag{2.33}$$

In the case where the percentage of inliers is known a-priori, the RANSAC iteration can terminate early (before N) as the currently determined inliers proportion approaches the known value. However, the percentage of inliers is usually not known beforehand, thus, the value of ϵ can first be set to a worst case value (say 0.1), which is then updated as larger inlier set is found. When a larger inlier set is identified, the percentage of inliers is known to be at least equals to the current number of inliers divided by the total number of measurements. Then, the value of N also decreases according to (2.33).

After the inlier-outlier sets are determined, the inliers are used to refine the parameter estimate to obtain the final solution. The final refinement can be performed either by linear weighted least square method, or an iterative non-linear optimisation method. In my work, I choose the weighted least square method due to the lower computational cost. After the refinement step, the inlier-outlier classification may change slightly. Thus, the measurements can be reclassified using the distance threshold, and refinement step repeated until the classification converges.

The method to classify inliers-outliers in Algorithm 3 uses a simple threshold. Maximizing the number of inliers can be interpreted as an optimisation of the following cost function.

$$\mathcal{C} = \sum_i \gamma(d_i) \quad \text{where } \gamma(e) = \begin{cases} 0 & \text{for } e < t \text{ (inliers)} \\ 1 & \text{for } e \geq t \text{ (outliers)} \end{cases} \tag{2.34}$$

Robust cost function may also be used during inlier-outlier classification. A trun-

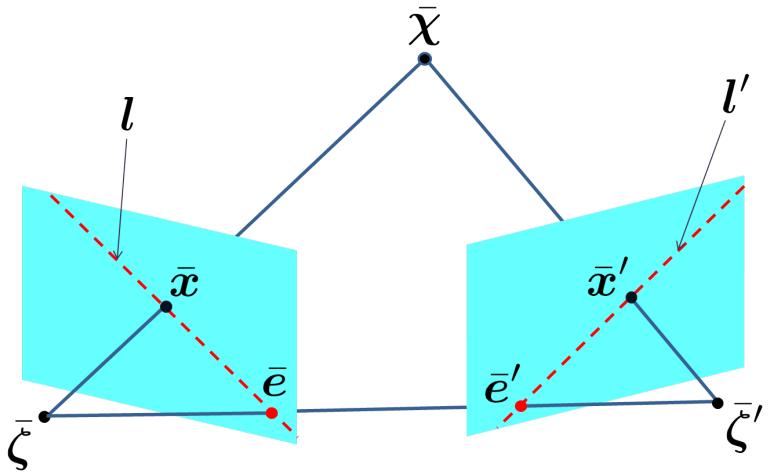

Figure 2.9: Illustrative figure showing the epipolar geometry. Blue regions represent the two image planes, $\bar{x}$ and $\bar{x}'$ are the matching image features of the same 3D scene point $\bar{\chi}$, $\bar{e}$ is the image point of the camera centre $\bar{\zeta}'$ (similarly for $\bar{e}'$ and $\bar{\zeta}$). $\bar{e}$ and $\bar{e}'$ are called the epipoles.

cated squared distance cost function is shown as follows. [Hartley and Zisserman, 2003].

$$\mathcal{C} = \sum_i \gamma(d_i) \quad \text{where} \quad \gamma(e) = \begin{cases} e^2 & \text{for } e^2 < t^2 \text{ (inliers)} \\ t^2 & \text{for } e^2 \geq t^2 \text{ (outliers)} \end{cases} \tag{2.35}$$

2.3.8 Epipolar geometry and fundamental matrix

Given two views (images) of the same scene, epipolar geometry is an intrinsic projective geometry that depends only on the intrinsic camera parameter and the relative pose between the cameras. Fundamental matrix captures this geometrical relationship in a 3×3 matrix with 7 degrees of freedom, and rank 2. Given the homography coordinates of matching feature points in two images as $\bar{x}$ and $\bar{x}'$, the following equation holds.

$$\bar{x}'^T \boldsymbol{F} \bar{x} = 0 \tag{2.36}$$

The fundamental matrix can be computed without the knowledge of the intrinsic camera parameter $\boldsymbol{K}$. Equation (2.36) can be rewritten as $\bar{x}'^T \boldsymbol{l}'$, and from Theorem 2.4, we can see that $\bar{x}'$ lies on the line $\boldsymbol{l}' = \boldsymbol{F}\bar{x}$. Similarly, by applying a transpose operation to (2.36), we get $\bar{x}^T \boldsymbol{F}^T \bar{x}' = 0$ such that $\bar{x}$ lies on the line $\boldsymbol{l} = \boldsymbol{F}^T \bar{x}'$. This geometrical relationship is illustrated in Figure 2.9.

It is noted that the points $\bar{\zeta}$, $\bar{\zeta}'$ and $\bar{\chi}$ lie on the same plane called the epipolar plane. For different 3D scene point $\bar{\chi}$, the epipolar plane may rotate around the baseline (line joining $\bar{\zeta}$ and $\bar{\zeta}'$). The intersection of the epipolar plane with the image plane then gives rise to the epipolar lines $\boldsymbol{l}$ and $\boldsymbol{l}'$ respectively.

Any points on the line passing through the points $\bar{\zeta}'$ and $\bar{\chi}$ will get projected onto the point $\bar{x}'$. This means that given an image point $\bar{x}'$, position of the corresponding

scene point $\bar{\chi}$ is constrained to the line through $\bar{\zeta}'$ and $\bar{\chi}$, without any constraint on the distance. Thus, the epipolar line $\boldsymbol{l}$ may also be seen as the projection of the line through $\bar{\zeta}'$ and χ onto the first image plane (left blue region in Figure 2.9). This argument applies similarly to line $\boldsymbol{l}'$.

The epipole $\bar{e}$ is the projection of the second camera centre $\bar{\zeta}'$ onto the first image plane. Similarly, the epipole $\bar{e}'$ is the projection of the first camera centre $\bar{\zeta}$ onto the second image plane. The epipolar lines will always pass through the respective epipoles regardless of the position of the 3D scene point.

The epipolar line is helpful when searching for matching image feature, where the search will be constrained to a one-dimensional search along the line, rather than the full two-dimensional image space.

The Fundamental matrix $\boldsymbol{F}$ is related to the two camera projective transformations $\boldsymbol{P}_{(3\times4)}$ and $\boldsymbol{P}'_{(3\times4)}$ in (2.27). This was first derived by Xu and Zhang [1996] and is given as follows.

Theorem 2.9 ([Hartley and Zisserman, 2003, pg. 243])**.** *Given the projective transformations matrices of camera* 1 *and camera* 2 *be* $\boldsymbol{P}$ *and* $\boldsymbol{P}'$ *respectively, the fundamental matrix* $\boldsymbol{F}$ *is equals to*

$$\boldsymbol{F} = [\boldsymbol{P}'\bar{\zeta}]_{\times}\boldsymbol{P}'\boldsymbol{P}^{+}, \tag{2.37}$$

where $[\boldsymbol{u}]_{\times}$ *is the skew symmetric matrix of vector* $\boldsymbol{u}$ *for cross product such that* $\boldsymbol{u} \times \boldsymbol{v} = [\boldsymbol{u}]_{\times}\boldsymbol{v}$, $\bar{\zeta}$ *is the homogeneous coordinate of the camera* 1*'s centre such that* $\boldsymbol{P}\bar{\zeta} = \boldsymbol{0}$*, and* $\boldsymbol{P}^{+}$ *is the pseudo-inverse of* $\boldsymbol{P}$ *such that* $\boldsymbol{P}\boldsymbol{P}^{+} = \boldsymbol{I}$*.*

Given a set of matching image points there are different methods to compute the fundamental matrix. For example, five-point algorithm [Nister, 2004; Li and Hartley, 2006], six-point algorithm [Schaffalitzky et al., 2000], and seven-point algorithm [Hartley and Zisserman, 2003] has been proposed.

However, we will focus on eight-point algorithm as the computation is the most straight forward. The normalised eight-point algorithm has also been shown to produce good result. [Hartley and Zisserman, 2003]

Suppose a pair of matching image features are $\boldsymbol{x} = [x, y, 1]^T$ and $\boldsymbol{x}' = [x', y', 1]^T$, the equation that satisfies the epipolar geometry (2.36) is

$$x'xf_{11} + x'yf_{12} + x'f_{13} + y'xf_{21} + y'yf_{22} + y'f_{23} + xf_{31} + yf_{32} + f_{33} = 0, \tag{2.38}$$

where f_{ij} are the elements of the fundamental matrix at i^{th} row and j^{th} column.

The same equation can also be written in the following form

$$[x'x, x'y, x', y'x, y'y, y', x, y, 1]\boldsymbol{f} = 0. \tag{2.39}$$

Given n pairs of matching image features, the equations can be concatenated into

a linear equation as

$$\boldsymbol{A}\boldsymbol{f} = \begin{bmatrix} x'_1x_1 & x'_1y_1 & x'_1 & y'_1x_1 & y'_1y_1 & y'_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x'_nx_n & x'_ny_n & x'_n & y'_nx_n & y'_ny_n & y'_n & x_n & y_n & 1 \end{bmatrix} \boldsymbol{f} = \boldsymbol{0}. \tag{2.40}$$

If the matrix $\boldsymbol{A}$ has rank 8, then the solution of fundamental matrix is then equals to the right-null space of $\boldsymbol{A}$. In the presence of noise, the rank of $\boldsymbol{A}$ may be 9. Then, using singular value decomposition (SVD), the solution that minimises the Frobenius norm $||\boldsymbol{A}\boldsymbol{f}||$ is the right singular vector that corresponds to the smallest singular value.

The computed fundamental matrix using this linear method may not satisfy the singular property of the fundamental matrix. We can enforce the singularity constraint by computing the closest singular matrix $\tilde{\boldsymbol{F}}$ to the computed fundamental matrix $\boldsymbol{F}$, which minimises the Frobenius norm $||\boldsymbol{F} - \tilde{\boldsymbol{F}}||$. This is easily accomplished by SVD. Suppose $\boldsymbol{F} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^T$, where $\boldsymbol{D} = \text{diag}(r, s, t)$ and $r \geq s \geq t$. Then, $\tilde{\boldsymbol{F}} = \boldsymbol{U}\text{diag}(r, s, 0)\boldsymbol{V}^T$.

In [Hartley, 1997], the importance of normalisation of the image features location during the fundamental matrix computation was shown. A translation and scaling is applied to image features such that for each sets of image features, the mean of the coordinate is at the origin, and the root mean squared distance of the points to the origin is $\sqrt{2}$. This provides a significant improvement to the conditioning of matrix $\boldsymbol{A}$, which improves the stability of the solution.

The method described thus far computes the fundamental matrix by minimising the algebraic error $||\bar{\boldsymbol{x}}'^T\boldsymbol{F}\bar{\boldsymbol{x}}||$. However, minimising this error does not guarantees the minimisation of a meaningful geometrical distance. Thus, other methods that minimises a different cost function was proposed. The Gold Standard method minimises the reprojection error, where the cost function is

$$\sum_i d(\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i)^2 + d(\boldsymbol{x}'_i - \hat{\boldsymbol{x}}'_i)^2,$$

where $d(\boldsymbol{v})$ is the Euclidean norm of vector $\boldsymbol{v}$, $\{\boldsymbol{x}_i, \boldsymbol{x}'_i\}$ are the measured position of the i^{th} matching image coordinate, while $\{\hat{\boldsymbol{x}}_i, \hat{\boldsymbol{x}}'_i\}$ are the estimated "true" location of the corresponding image point that satisfies $\hat{\boldsymbol{x}}'^T_i\boldsymbol{F}\hat{\boldsymbol{x}}_i = 0$ exactly. The Gold Standard method is given in Algorithm 11.3 from [Hartley and Zisserman, 2003].

The Gold Standard method is accurate but it is computationally complex. A first-order approximation of the geometric error called Sampson distance was used [Torr and Zisserman, 1998; Zhang, 1998]. It was inspired by iterative weighted least square method, which was first used to fit a conic to scattered data points in [Sampson, 1982].

Theorem 2.10 ([Hartley and Zisserman, 2003, pg. 287])**.** *Given the i^{th} matching pixel $\boldsymbol{x}_i$ and $\boldsymbol{x}'_i$, the cost function to compute the fundamental matrix that minimises the first order*

approximation of geometric distance (Sampson distance) is given by

$$\sum_i \frac{(\boldsymbol{x}_i'^T \boldsymbol{F} \boldsymbol{x}_i)^2}{(\boldsymbol{F}\boldsymbol{x}_i)_1^2 + (\boldsymbol{F}\boldsymbol{x}_i)_2^2 + (\boldsymbol{F}^T\boldsymbol{x}_i')_1^2 + (\boldsymbol{F}^T\boldsymbol{x}_i')_2^2}. \tag{2.41}$$

Thus, when estimating the fundamental matrix $\boldsymbol{F}$ from point correspondences, we can scale each of the linear equations (rows of (2.40)) by

$$\phi_i = \frac{1}{\sqrt{(\boldsymbol{F}\boldsymbol{x}_i)_1^2 + (\boldsymbol{F}\boldsymbol{x}_i)_2^2 + (\boldsymbol{F}^T\boldsymbol{x}_i')_1^2 + (\boldsymbol{F}^T\boldsymbol{x}_i')_2^2}}. \tag{2.42}$$

The fundamental matrix is then computed using SVD as before. This can be done iteratively, where an initial $\boldsymbol{F}$ is first computed without the reweighing, and subsequent computations use the previously computed $\boldsymbol{F}$ to compute the weighs for each equations.

There are other distance measures like the symmetric epipolar distance [Zhang, 1998], and Katakani distance [Fathy et al., 2011; Kanatani et al., 2008]. However, [Fathy et al., 2011] shows that the Sampson distance is still superior in terms of the accuracy and computational complexity after outliers has been removed. It is also noted that most of the existing distance measures assumes an isotropic, homogeneous Gaussian error for the image features location.

2.3.9 Essential matrix and inter-frame pose

In most cases, the camera intrinsic parameters can be estimated beforehand through camera calibration. If the intrinsic camera parameter matrix is known, the image points can be expressed in normalized coordinates. Suppose $\boldsymbol{K}$ is the intrinsic camera parameter matrix, $\bar{\boldsymbol{x}}$ is the homogeneous coordinate of an image point, the homogeneous image point in normalized coordinates is

$$\hat{\boldsymbol{x}} = \boldsymbol{K}^{-1}\bar{\boldsymbol{x}}. \tag{2.43}$$

We can rewrite the epipolar geometry equation (2.36) as

$$\hat{\boldsymbol{x}}'^T \boldsymbol{E} \hat{\boldsymbol{x}} = 0. \tag{2.44}$$

The matrix $\boldsymbol{E}$ is called essential matrix, which is a 3×3 matrix with 5 degrees of freedom. It is related to the fundamental matrix as

$$\boldsymbol{E} = \boldsymbol{K}'^T \boldsymbol{F} \boldsymbol{K}. \tag{2.45}$$

Theorem 2.11 ([Hartley and Zisserman, 2003, pg. 257])**.** *Suppose the normalised camera projective matrices are* $\hat{\boldsymbol{P}} = [\boldsymbol{I} \;\; \boldsymbol{0}]$ *and* $\hat{\boldsymbol{P}}' = [\boldsymbol{R} \;\; \boldsymbol{t}]$*. The essential matrix is equals to*

$$\boldsymbol{E} = [\boldsymbol{t}]_\times \boldsymbol{R}. \tag{2.46}$$

From equation (2.44) and (2.46), we can see that the essential matrix has 5 degrees of freedom. This arises from 3 degrees of freedom for 3D rotation, while 2 degrees of freedom for 3D translation with scale ambiguity (any scalar multiple of $\boldsymbol{E}$ will still satisfy (2.44)). We can extract the rotation and translation from the essential matrix as follows.

Theorem 2.12 ([Hartley and Zisserman, 2003, pg. 259])**.** *Given an essential matrix $\boldsymbol{E}$, assuming the first camera projection matrix $\boldsymbol{P} = [\boldsymbol{I} \ \boldsymbol{0}]$. The singular value decomposition of the essential matrix can be written as*

$$\boldsymbol{E} = \boldsymbol{U} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \boldsymbol{V}^T.$$

We define a matrix $\boldsymbol{W}$ such that

$$\boldsymbol{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.47}$$

There are four possible solutions for the second camera projection matrix

$$\boldsymbol{P}' = \begin{bmatrix} \boldsymbol{U}\boldsymbol{W}\boldsymbol{V}^T & +\boldsymbol{u}_3 \end{bmatrix}, \tag{2.48}$$

or

$$\boldsymbol{P}' = \begin{bmatrix} \boldsymbol{U}\boldsymbol{W}\boldsymbol{V}^T & -\boldsymbol{u}_3 \end{bmatrix}, \tag{2.49}$$

or

$$\boldsymbol{P}' = \begin{bmatrix} \boldsymbol{U}\boldsymbol{W}^T\boldsymbol{V}^T & +\boldsymbol{u}_3 \end{bmatrix}, \tag{2.50}$$

or

$$\boldsymbol{P}' = \begin{bmatrix} \boldsymbol{U}\boldsymbol{W}^T\boldsymbol{V}^T & -\boldsymbol{u}_3 \end{bmatrix}, \tag{2.51}$$

where $\boldsymbol{u}_3$ is the last column of $\boldsymbol{U}$.

Only one of the solutions of $\boldsymbol{P}'$ represents the true relative camera pose. This can be identified using the chirality constraint, where the triangulated 3D scene point must lie in front of both cameras.

2.3.10 SLAM

Simultaneous localization and mapping (SLAM) refers to method that computes both the current sensor/robot pose (self-localization) and location of landmarks (environment mapping). In visual SLAM, by using a calibrated camera, image feature matches in normalised coordinate allows the computation of essential matrix. The camera pose can then be recovered from the computed essential matrix.

The next step is to estimate the location of landmarks, which can be calculated by triangulation. In computer vision literature, the triangulation of landmarks (3D

scene points) is known as reconstruction. Reconstruction is typically performed as follows:

1. Identify image features matches from two (or more) images
2. Compute the fundamental matrix with image features matches
3. Estimate the camera matrices from the fundamental matrix
4. Triangulate the 3D scene points that corresponds to the matching image features

Without knowledge of the camera intrinsic parameter, reconstruction has a projective ambiguity, where the angles between rays of light also has additional scale ambiguity. Longuet-Higgins [1981] showed that if the camera intrinsic parameters are calibrated, then the reconstruction is determined up to a similarity transformation (scale, rotation and translation).

The similarity transformation ambiguity is one of the limitation of visual SLAM, where it is not possible to recover the exact location or pose of the camera and 3D scene points. For example, the absolute latitude, longitude or orientation cannot be recovered purely from images. They can only be determined up to a Euclidean transformation (3D rotation and translation) with respect to the world frame. Thus, the commonly used convention is to assume the first camera at the first time instant is aligned to the world coordinate frame. The rest of the poses and reconstruction will then be expressed with respect to this frame of reference.

Another limitation of visual SLAM is the scale ambiguity of the reconstruction and translation. This is due to the fact that any scalar multiple of translation $\boldsymbol{t}$ will still satisfy the equation (2.46). With a translation of $\lambda\boldsymbol{t}$, the triangulated 3D scene points will also be equally scaled by λ. An observable example of the scale ambiguity is the difficulty in distinguishing images of a perfect miniature replica from images of an actual building.

The reconstruction scale can be fixed, where a metric reconstruction can be obtained with additional knowledge or assumptions. For example, in images taken from a camera mounted on a vehicle, the known height of the camera can be used to fix the scale. Other knowledge of the scene is also possible, such as known object with fixed dimension can be used to recover the scale.

2.4 Path smoothing

Estimation of robot's pose is an important task in robotics system. One method to obtain the pose estimate is by using an onboard camera to obtain noisy bearing measurements of mostly stationary environment (see Chapter 2.3). However, due to the presence of noise, the estimated robot pose trajectory may not be smooth even if the robot actually undergoes a smooth motion. Another research problem considers the estimated pose to be accurate, where computing the smooth pose trajectory is

useful in applications such as video stabilization. Thus, an effective path smoothing method is important for both tasks.

The commonly used representations of a three-dimensional pose is briefly discussed as follows.

2.4.1 Translation representation

3D translation is typically represented using a vector of length 3, such that

$$t \in \mathbb{R}^3. \tag{2.52}$$

2.4.2 Rotation representation

3D rotation can be represented in different forms. Some well-known examples include the rotation matrix, Euler angles, angle-axis representation and quaternion. However, it was known that the Euler angles representation is not unique, as the same rotation can be obtained by a sequence of different rotations. Angle-axis representation is also not unique, where applying a negative rotational angle along the negative axis is equivalent to the non-negative case. Similarly, quaternion representation is also not unique as $\boldsymbol{q} = -\boldsymbol{q}$.

Thus, we focus on the rotation matrix representation, which uniquely defines a particular 3D orientation. The 3D rotation matrix $\boldsymbol{R}$ is a 3×3 matrix, and has the following properties.

$$\boldsymbol{R}^T = \boldsymbol{R}^{-1}, \tag{2.53}$$

$$\det(\boldsymbol{R}) = +1. \tag{2.54}$$

Orthogonal group $O(n)$ satisfies the property (2.53), while rotation from the special orthogonal group $SO(n)$ also satisfies the property (2.54).

Chapter 3

Minimal Iterative Gaussian Estimator

This chapter introduces a new approximate Bayesian estimator using degenerate Gaussian. The most closely related work are the inverse-depth parameterization in monocular simultaneous localization and mapping (SLAM) to represent the image features in infinite distance [Civera et al., 2008; Montiel et al., 2006] and extended information filter (EIF) commonly used in SLAM [Anderson and Moore, 1979; Thrun et al., 2005; Huang and Dissanayake, 2007], which however relies on the linearized measurement Jacobians. This leads to inconsistency in estimation when the prior uncertainty is large. We propose a new filtering method called Minimal Iterative Gaussian Estimator (MIGE). The measurement likelihood is expressed in the state space, which allows computation of the measurement likelihood covariance in the state space to ensure estimator consistency. Due to the mapping from lower dimensional measurement space to higher dimensional state space, we utilize a degenerate Gaussian to better approximate the measurement likelihood. We also introduce a new re-parametrization to handle the degeneracy while decreasing the memory and computational requirement of the algorithm. The key contributions of this chapter are three-fold:

- The explicit use of degenerate Gaussian to approximate the non-Gaussian measurement likelihood function, while ensuring estimation consistency by utilizing the prior uncertainty and local nonlinearity of the measurement function. To the best of the author's knowledge, this has not been addressed elsewhere.
- A new parametrized form to handle the degenerate Gaussian is further developed, resulting in the proposing of the Minimal Iterative Gaussian Estimator.
- The performance of the methods, in terms of accuracy, consistency and computational complexity, is verified through extensive simulation analysis.

The rest of the chapter is organized as follows. Section 3.1 covers some related work. In Section 3.2, the nonlinear state and measurement models are briefly described and Section 3.3 introduces the degenerate Gaussian as an approximated likelihood function. A re-parametrization method is discussed in Section 3.4 followed

by the Minimal Iterative Gaussian Estimator framework in Section 3.5. Section 3.6 presents the simulation results for the bearing-only and range-only cases in 2D and 3D scenarios. Finally, the summary is presented in Section 3.7.

3.1 Related Work

The Bayesian filtering framework has been widely investigated and applied to many engineering fields such as probabilistic robot navigation, sensor network, target tracking [Chen, 2003; Stano et al., 2013] and so on. Its benefits lie in the full probabilistic descriptions of the nonlinear state and measurement models, utilizing the conditional independence between the variables of interest.

The most commonly used efficient non-linear estimator suitable for resource constrained system are the Extended Kalman Filter (EKF) [Sorenson, 1985]. However, it was known that the estimators are often inconsistent, where the uncertainty of the estimate is often underestimated. A number of past works [Huang et al., 2010; Li and Mourikis, 2013] focus on finding a consistent EKF suitable for simultaneous localization and mapping tasks. The works concluded that the inconsistency arises due to the sensor's (or robot's) orientation being seemingly observable when the actual measurement does not provide such information. More recently, Zhang et al. [2017] claims that the inconsistency of EKF is due to the filter not being invariant to stochastic rigid body transformation. However, even when this factor was removed by using accurate position and orientation of the robot (purely localization task), the EKF (or equivalently EIF) can still produce inconsistent result (Figure 3.7(b)). We postulate that the inconsistency arises due to the mismatch between the true conditional PDF of the measurement and the approximated PDF.

There are also other variants of Kalman filter such as extended information filter (EIF) [Anderson and Moore, 1979], unscented Kalman filter (UKF) [Uhlmann, 1997] or cubature Kalman filter (CKF) [Arasaratnam and Haykin, 2009]. The EIF is mathematically equivalent to EKF, where the conditional PDF is represented using the information matrix and information vector instead of the covariance matrix and mean. The UKF uses a set of sample points (called sigma points), where the mean and covariance matrix are subsequently computed from the propagated sample points through the nonlinear state prediction and measurement functions. Similar to UKF, CKF uses a set of cubature points to capture the underlying conditional PDF, where cubature rule is applied during the cubature points selection.

Another approach has been to utilize a large number of parameters to represent nonlinear, non-Gaussian conditional PDF. One example of such approach is the Gaussian sum filter [Sorenson and Alspach, 1971; Alspach and Sorenson, 1972]. It was proven that any arbitrary PDF can be represented as a weighted sum of a sufficiently large number of Gaussian PDFs [Ito and Xiong, 2000; Maz'ya and Schmidt, 1996]. A bank of EKF was used to propagate and update the PDFs. One primary drawback of the Gaussian sum filter is the difficulty in ensuring accurate approximation of the true PDF, while ensuring computational tractability. This is due to the

tendency for the number of Gaussian PDFs to grow exponentially with increasing nonlinearity of the PDF to be approximated. A suitable method to trim and merge the Gaussian PDFs is vital for this filter [Sorenson and Alspach, 1971]. Even with the trimming and merging of Gaussian PDF, the computational complexity of Gaussian sum filter remains high.

An alternative class of nonlinear estimator suitable for a highly nonlinear, non-Gaussian PDF is based on the Monte Carlo method. The Monte Carlo methods can be traced back to the attempt by Buffon to estimate the value of π in 1777 [Solomon, 1978]. The modern formulation of the Monte Carlo method was developed in physics [Metropolis and Ulam, 1949; Metropolis et al., 2004], statistics [Chen, 2003; Robinson, 2010] and engineering [Doucet et al., 2001] among others. The Monte Carlo method is a stochastic sampling method designed to solve an analytically intractable problem of a complex system. A sequential Monte Carlo method like the particle filter [Gordon et al., 2002] combines the powerful Monte Carlo sampling with Bayesian inference, which allows real-time state estimation with a reasonable computational cost. However, sequential Monte Carlo methods are not suitable for solving high dimensional problems due to the need for a sizeable amount of samples (or particles) to capture the underlying PDF adequately.

In comparison, our proposed method exploits the geometrical aspect of most non-Gaussian measurement likelihood by explicitly fitting a degenerate Gaussian to the likelihood function within a local region. The estimator's consistency is improved by computing the covariance of the non-degenerate directions using prior uncertainty and local nonlinearity of the measurement function. A minimal parametrized representation is also introduced, which reduces the computational and memory requirement.

3.2 Nonlinear System Model

The stochastic state and measurement models can be written as

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k) + \boldsymbol{v}_k \tag{3.1}$$

$$\boldsymbol{z}_k = \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{w}_k, \tag{3.2}$$

where $\boldsymbol{x}_k$ is the state vector at discrete time k; $\boldsymbol{u}_k$ is the input vector; $\boldsymbol{z}_k$ is the measurement vector; $\boldsymbol{f}$ is the state transition function; $\boldsymbol{h}$ is the known measurement function; $\boldsymbol{v}_k$ and $\boldsymbol{w}_k$ are independent process and measurement noise, which is a zero-mean Gaussian process with a covariance of $\boldsymbol{Q}_k$ and $\boldsymbol{S}_k$ respectively.

The probability density function of the state can be written as follows

$$p(\boldsymbol{x}_k) = C \cdot \exp\left\{-\frac{1}{2}(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k)^T \boldsymbol{P}_k^{-1}(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k)\right\}, \tag{3.3}$$

where $\hat{\boldsymbol{x}}_k$ is the state mean, and the covariance matrix $\boldsymbol{P}_k$ is a positive definite matrix.

The probabilistic likelihood function for a given measurement ($z = z_k$) becomes

$$p(z_k|x_k) = \exp\left\{-\frac{1}{2}(z_k - h(\hat{x}))^T S_k^{-1}(z_k - h(\hat{x}))\right\}. \tag{3.4}$$

Note that the P_k and S_k can be further decomposed into a rotation matrix R_k and a diagonal matrix Σ_k such that $S_k = R_k \Sigma_k R_k^T \succ 0$. In addition, the normalization constant in the likelihood is dropped since it is not a probability density in general. Although the measurement model is in a Gaussian form, the likelihood $p(z_k|x_k)$ in the state-space typically has nonlinear, non-Gaussian shapes. Figure 3.1 illustrates two examples of likelihood functions for the bearing-only and range-only measurement, having a conic and a shell shape, respectively.

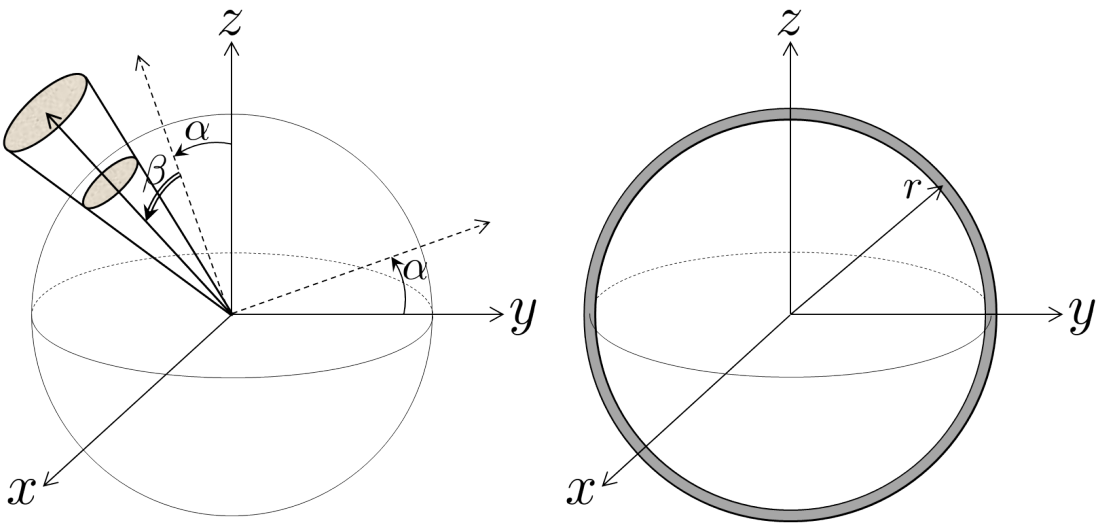

Figure 3.1: Nonlinear likelihood distribution: (a) the bearing-only measurement with α and β angles, and (b) the range-only measurement with r and uncertainty.

3.3 Degenerate Gaussian

A degenerate Gaussian is defined as one or more of its eigenvalues in Σ being infinite. It is thus not a proper probability distribution due to infinite area or volume. In particular, two cases are of interest: cylindrical and planar function,

$$\text{Cylindrical Function: } \Sigma = \left[\begin{array}{cc|c} {\sigma_1}^2 & 0 & 0 \\ 0 & {\sigma_2}^2 & 0 \\ \hline 0 & 0 & \infty \end{array}\right] \tag{3.5}$$

$$\text{Planar Function: } \Sigma = \left[\begin{array}{cc|c} \infty & 0 & 0 \\ 0 & \infty & 0 \\ \hline 0 & 0 & {\sigma_1}^2 \end{array}\right]. \tag{3.6}$$

The likelihood function of a bearing measurement can be approximated as a cylindrical function with an infinite eigenvalue in one axis (for example, z-axis) as shown in Figure 3.2, and that of a range measurement can be approximated as a planar function with two infinite eigenvalues as illustrated in Figure 3.3. These likelihood functions can be geometrically transformed with rotation and translation to approximate the three-dimesional (3D) bearing and range sensing.

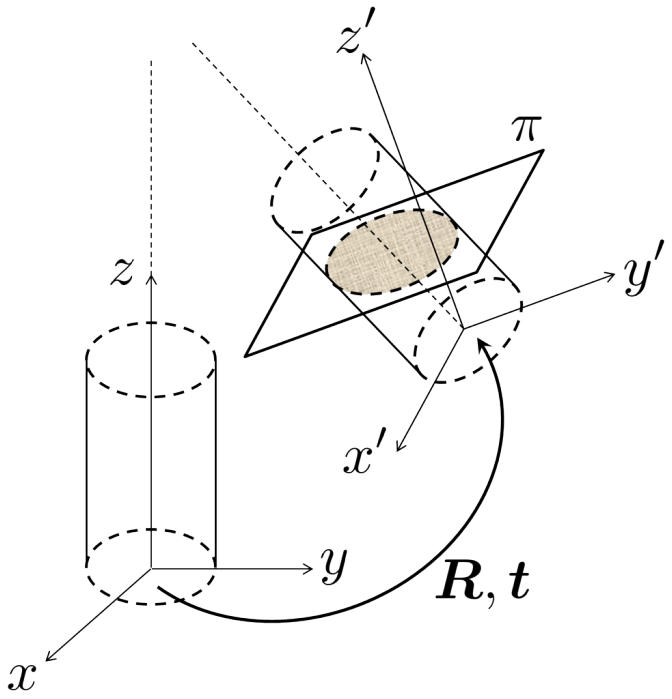

Figure 3.2: A degenerate Gaussian distribution to approximate the bearing-only measurement likelihood which has an infinite uncertainty in one of the principle axes. From left cylinder to right cylinder: (a) degenerate Gaussian with infinite uncertainty along the z direction; (b) rotated and translated degenerate Gaussian with uncertainty defined on the plane π (parallel to x'–y' plane) as shown in the shaded region.

3.3.1 Bearing-only case

Figure 3.2 shows the transformation of the degenerate Gaussian to approximate the bearing-only measurement from a camera. First, the vertical cylinder function is tilted along the direction pointed by the elevation (α) and yaw (β) angles, yielding,

$$p(\boldsymbol{z}_k|\boldsymbol{x}) = \mathcal{G}(\boldsymbol{0}, \boldsymbol{R}_1 \boldsymbol{\Sigma} {\boldsymbol{R}_1}^T), \tag{3.7}$$

where $\mathcal{G}$ represents a Gaussian likelihood function (without the normalization constant) and

$$\boldsymbol{R}_1(\alpha, \beta) = \boldsymbol{R}_{y'}(\beta)\boldsymbol{R}_{x'}(\alpha) \tag{3.8}$$

$$\boldsymbol{R}_{x'}(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) \\ 0 & \sin(\alpha) & \cos(\alpha) \end{bmatrix} \tag{3.9}$$

$$\boldsymbol{R}_{y'}(\beta) = \begin{bmatrix} \cos(\beta) & 0 & \sin(\beta) \\ 0 & 1 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) \end{bmatrix} \tag{3.10}$$

$$\boldsymbol{\Sigma} = \left[\begin{array}{cc|c} {\sigma_1}^2 & 0 & 0 \\ 0 & {\sigma_2}^2 & 0 \\ \hline 0 & 0 & \infty \end{array}\right]. \tag{3.11}$$

Then the tilted cylindrical function undergoes the rigid-body transformation $(\boldsymbol{R}, \boldsymbol{t})$ of the camera frame, giving

$$p(\boldsymbol{z}_k|\boldsymbol{x}) = \mathcal{G}(\boldsymbol{t}, \boldsymbol{R}\boldsymbol{R}_1\boldsymbol{\Sigma}{\boldsymbol{R}_1}^T\boldsymbol{R}^T). \tag{3.12}$$

3.3.2 Range-only case

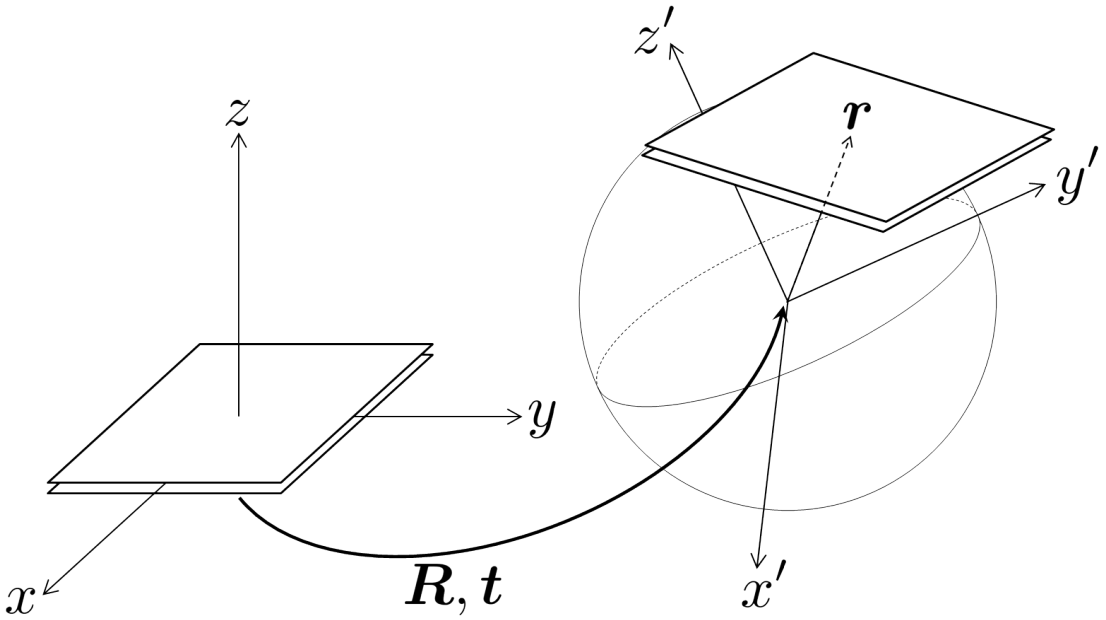

Figure 3.3: A degenerate Gaussian Distribution to approximate the range-only measurement likelihood which has an infinite uncertainty in two of the principle axes. From left to right: (a) degenerate Gaussian with infinite uncertainty in both x and y direction; (b) rotated and translated degenerate Gaussian towards the range vector $\boldsymbol{r}$.

For the range-only case, Figure 3.3 illustrates the transformation of a planar degenerate Gaussian to approximate the measurement. Similar to the bearing-only case, the planar function is tilted towards the prior direction of a ranging radio tar-

get. A planar degenerate Gaussian has infinite uncertainty along the x and y axes which can be written as

$$p(\boldsymbol{z}_k|\boldsymbol{x}) = \mathcal{G}(\boldsymbol{0}, \boldsymbol{\Sigma}) \tag{3.13}$$

$$\boldsymbol{\Sigma} = \left[\begin{array}{cc|c} \infty & 0 & 0 \\ 0 & \infty & 0 \\ \hline 0 & 0 & {\sigma_1}^2 \end{array}\right]. \tag{3.14}$$

For a prior target vector $\boldsymbol{r} = [r_x,\ r_y,\ r_z]^T$ expressed in the sensor frame ($x'\ y'\ z'$), the planar function is tilted along the direction, yielding

$$p(\boldsymbol{z}_k|\boldsymbol{x}) = \mathcal{G}(\boldsymbol{r}, \boldsymbol{R}_1 \boldsymbol{\Sigma} {\boldsymbol{R}_1}^T) \tag{3.15}$$

where

$$\begin{aligned} \boldsymbol{R}_1 &= \boldsymbol{R}_{y'}(\beta)\boldsymbol{R}_{x'}(\alpha) && (3.16)\\ \alpha &= \arctan(-r_y/\sqrt{{r_x}^2 + {r_z}^2}) && (3.17)\\ \beta &= \arctan(r_x/r_z). && (3.18) \end{aligned}$$

Then the tilted planar function undergoes the rigid-body transformation $(\boldsymbol{R}, \boldsymbol{t})$ of the sensor frame, yielding the final planar function as

$$p(\boldsymbol{z}_k|\boldsymbol{x}) = \mathcal{G}(\boldsymbol{R}\boldsymbol{r} + \boldsymbol{t}, \boldsymbol{R}\boldsymbol{R}_1 \boldsymbol{\Sigma} {\boldsymbol{R}_1}^T \boldsymbol{R}^T). \tag{3.19}$$

Note that the direction of $\boldsymbol{r}$ is determined from the prior estimate and iterative estimation process, and the range measurement determines only the magnitude of $\boldsymbol{r}$.

3.4 Re-parametrization

The degenerate Gaussian likelihood contains a singular information matrix (an inverse-covariance matrix) and thus the mean is not uniquely defined as well as the covariance parameters not being recoverable. For example, the covariance matrix of the degenerate Gaussian cannot be recovered from the singular information matrix as

$$\text{Cylindrical Function: } \boldsymbol{\Sigma}^{-1} = \left[\begin{array}{cc|c} {\sigma_1}^{-2} & 0 & 0 \\ 0 & {\sigma_2}^{-2} & 0 \\ \hline 0 & 0 & 0 \end{array}\right] \tag{3.20}$$

$$\text{Planar Function: } \boldsymbol{\Sigma}^{-1} = \left[\begin{array}{cc|c} 0 & 0 & 0 \\ 0 & 0 & 0 \\ \hline 0 & 0 & {\sigma_1}^{-2} \end{array}\right]. \tag{3.21}$$

This issue can be resolved in the information filtering framework as it can naturally handle the singular information. However, in 2D/3D sensing problems, we can further improved the computational complexity by re-parametrizing the degen-

erate Gaussian by expanding its quadratic equation of the exponential term. Let the information matrix for a 3D measurement be

$$\boldsymbol{P}^{-1} = \begin{bmatrix} Y_{xx} & Y_{xy} & Y_{xz} \\ Y_{xy} & Y_{yy} & Y_{yz} \\ Y_{xz} & Y_{yz} & Y_{zz} \end{bmatrix}. \tag{3.22}$$

The quadratic equation of the Gaussian exponential term (dropping the sign and half) in equation (3.3) expands as

$$\begin{aligned} &(\boldsymbol{x} - \hat{\boldsymbol{x}})^T \boldsymbol{P}^{-1} (\boldsymbol{x} - \hat{\boldsymbol{x}}) \\ &\quad = \begin{bmatrix} x - \hat{x} \\ y - \hat{y} \\ z - \hat{z} \end{bmatrix}^T \begin{bmatrix} Y_{xx} & Y_{xy} & Y_{xz} \\ Y_{xy} & Y_{yy} & Y_{yz} \\ Y_{xz} & Y_{yz} & Y_{zz} \end{bmatrix} \begin{bmatrix} x - \hat{x} \\ y - \hat{y} \\ z - \hat{z} \end{bmatrix} \\ &\quad = (Y_{xx})x^2 + (Y_{yy})y^2 + (Y_{zz})z^2 \\ &\qquad + (2Y_{xy})xy + (2Y_{xz})xz + (2Y_{yz})yz \\ &\qquad + (-2Y_{xx}\hat{x} - 2Y_{xy}\hat{y} - 2Y_{xz}\hat{z})x \\ &\qquad + (-2Y_{xy}\hat{x} - 2Y_{yy}\hat{y} - 2Y_{yz}\hat{z})y \\ &\qquad + (-2Y_{xz}\hat{x} - 2Y_{yz}\hat{y} - 2Y_{zz}\hat{z})z \\ &\qquad + const. \end{aligned} \tag{3.23}$$

Please note that the constant term can be dropped as it contribute to the scaling of the exponential function. The minimal representation of the 3D Gaussian function is thus obtained using a vector of length 9 to store the coefficients of the quadratic equation

$$\begin{aligned} \mathcal{P} &\triangleq [p_1,\ p_2,\ p_3,\ p_4,\ p_5,\ p_6,\ p_7,\ p_8,\ p_9]^T && (3.24) \\ &= [Y_{xx},\ Y_{yy},\ Y_{zz},\ 2Y_{xy},\ 2Y_{xz},\ 2Y_{yz}, \\ &\qquad -2(Y_{xx}\hat{x} + Y_{xy}\hat{y} + Y_{xz}\hat{z}), \\ &\qquad -2(Y_{xy}\hat{x} + Y_{yy}\hat{y} + Y_{yz}\hat{z}), \\ &\qquad -2(Y_{xz}\hat{x} + Y_{yz}\hat{y} + Y_{zz}\hat{z})]^T && (3.25) \\ \boldsymbol{\chi} &\triangleq [\ x^2 \quad y^2 \quad z^2 \quad xy \quad xz \quad yz \quad x \quad y \quad z\ ]^T. && (3.26) \end{aligned}$$

The degenerate Gaussian function is then represented using this new parameter ($\mathcal{P}$),

$$p(\boldsymbol{z}_k|\boldsymbol{x}) = \exp\left\{-\frac{1}{2}\mathcal{P}^T\boldsymbol{\chi}\right\}. \tag{3.27}$$

Note that the new parameter $\mathcal{P}$ actually is a concatenation of an information

matrix Y and an information state estimate $\hat{y}$, which can be converted

$$\boldsymbol{Y} = \mathrm{Mat}(\mathcal{P}^{(1:6)}) \triangleq \begin{bmatrix} p_1 & p_4/2 & p_5/2 \\ p_4/2 & p_2 & p_6/2 \\ p_5/2 & p_6/2 & p_3 \end{bmatrix} \tag{3.28}$$

$$\hat{\boldsymbol{y}} = \boldsymbol{Y}\hat{\boldsymbol{x}} \triangleq -\frac{1}{2}\mathcal{P}^{(7:9)}. \tag{3.29}$$

where $\mathrm{Mat}(\cdot)$ defined as a symmetric matricization operator converting the parameters into an information matrix.

3.5 Minimal Iterative Gaussian Estimator

By utilizing the degenerate Gaussians and the new coefficient parameters, a new estimation framework is proposed, termed Minimal Iterative Gaussian Estimator (MIGE), which has a state propagation and an iterative measurement update cycles.

3.5.1 State Propagation

Note that the measurement likelihood expressed in the state space is modelled using degenerate Gaussian, but prior and posterior likelihood are in general not degenerate. Thus, the inverse of the information matrix is well defined. Therefore, we can apply the EIF state propagation method. However, any other nonlinear state propagation methods can be applied here.

First, the prior information matrix $\boldsymbol{Y}_{k-1}$ and the information state $\hat{\boldsymbol{y}}_{k-1}$ are recovered from the prior parametrized representation $\mathcal{P}_{k-1|k-1}$ as in Eqs. 3.28 and 3.29.

Then the prediction is performed as in Anderson and Moore [1979],

$$[\boldsymbol{Y}_{k-1}, \hat{\boldsymbol{y}}_{k-1}] = \left[\mathrm{Mat}(\mathcal{P}^{(1:6)}_{k-1|k-1}), -\frac{1}{2}\mathcal{P}^{(7:9)}_{k-1|k-1}\right] \tag{3.30}$$

$$\boldsymbol{Y}_k = \left[\boldsymbol{F}_k \boldsymbol{Y}_{k-1}^{-1} \boldsymbol{F}_k^T + \boldsymbol{Q}_k\right]^{-1} \tag{3.31}$$

$$\hat{\boldsymbol{y}}_k = \boldsymbol{Y}_k \boldsymbol{f}(\boldsymbol{Y}_{k-1}{}^{-1}\hat{\boldsymbol{y}}_{k-1}, \boldsymbol{u}_k) \tag{3.32}$$

$$\mathcal{P}_{k|k-1} = \left[\mathrm{Mat}^{-1}(\boldsymbol{Y}_k)^T, -2\hat{\boldsymbol{y}}_k^T\right]^T, \tag{3.33}$$

where the subscript $i|j$ shows the variable at time i with measurements up to time j, and $\boldsymbol{F}_k$ and $\boldsymbol{B}_k$ are the Jacobian matrix $\partial \boldsymbol{f}/\partial \boldsymbol{x}$ and $\partial \boldsymbol{f}/\partial \boldsymbol{u}$, respectively.

3.5.2 Measurement Update

With new measurements, the state distribution can be updated by fusing the predicted density and the approximated measurement likelihood. The posterior parameter $\mathcal{P}_{k|k}$ is computed through element-wise addition of the predicted parameter

$\mathcal{P}_{k|k-1}$ and the measurement parameters $\{\mathcal{P}_i\}$ as

$$p(\boldsymbol{z}_i|\boldsymbol{x}) = \exp\left\{-\frac{1}{2}\mathcal{P}_i^T\boldsymbol{\chi}\right\} \tag{3.34}$$

$$\mathcal{P}_{k|k} = \mathcal{P}_{k|k-1} + \sum_{i=1}^{K}\mathcal{P}_i. \tag{3.35}$$

These propagate and update processes complete the estimation cycle of the *Minimal Iterative Gaussian Estimator*. Please note that the fusion process can be iterated to improve the approximated likelihood. It is known that the state uncertainty shrinks after measurement update. Thus, the measurement likelihood can be re-approximated by using the fused uncertainty instead of the prior uncertainty. This leads to an uncertainty reduction in the approximated measurement and a subsequent improvement of the estimation result as in the iteration of Gauss-Newton method Bell and Cathey [1993].

3.5.3 Computing Measurement Uncertainty for Estimator Consistency

The measurement likelihood covariance in the state space Σ is chosen such that the approximated likelihood encloses the true underlying likelihood within the current state prior $(\hat{\boldsymbol{y}}_{k|k-1}, \boldsymbol{Y}_{k|k-1})$. This is done to improve the consistency of the estimator. An illustrative example is shown in Figure 3.4.

The original likelihood may extend to infinity, and thus we only approximate the likelihood around a region of interest. This region is defined to be within χ Mahalanobis distance from the prior, where the χ value is selected depending on the desired level of confidence based on the chi-square distribution. We choose χ to be 3 in our experiment.

For a bearing measurement with isotropic noise (the covariance matrix has equal non-zero eigenvalues), the σ is chosen as $\sigma_\angle d_{max}$, where $\sigma_\angle$ is the measurement's uncertainty, $d_{max} = d_{\hat{x}} + 3d_P$ is the mean distance from the sensor plus 3 times the standard deviation in distance (from the prior). The second term in d_{max} can be seen as the deviation of the approximated likelihood from the actual measurement likelihood.

For a range measurement, the σ is chosen as $\sigma_r + (r - \sqrt{r^2 - (3w)^2})$, where σ_r is the measurement's uncertainty, r is the measured range, w is the largest width of the prior along the tangent plane. Thus, if the measured range is significantly larger than the uncertainty of the prior along the tangent plane, the $\sigma \approx \sigma_r$ as expected.

3.6 Simulation Results

Simulation study is conducted to analyse the performance of MIGE for a 2D and 3D bearing-only target localization.

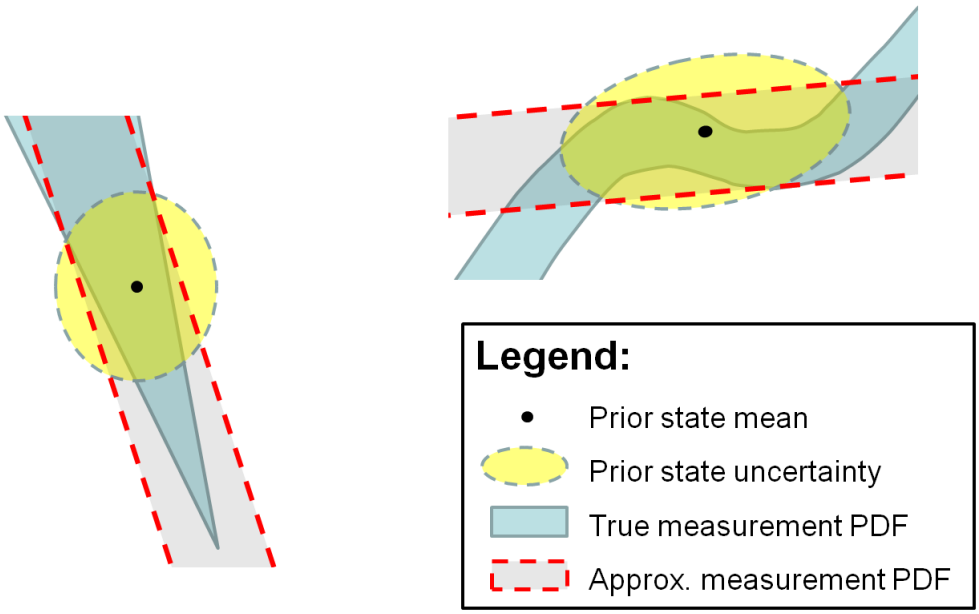

Figure 3.4: Illustrative figure showing the approximated uncertainty σ to avoid estimator inconsistency. The "width" of the distribution is chosen to ensure the intersection between the confidence contour of the prior (yellow ellipse) and the measurement likelihood (blue region) are enclosed within the confidence contour of the approximated likelihood. From left to top right: (a) bearing measurement; (b) arbitrary nonlinear measurement.

3.6.1 2D Bearing-only Localization

Monte Carlo simulation for a 2D and a 3D bearing-only localization are performed using the proposed MIGE. The true target location is at $[20.25 - 1.5t, -19.75 + 1.5t]$, with t being a time variable. The sensor is located at $[-20, -20]$. The simulated process (motion) noise has a zero mean and a standard deviation of σ_p for both x and y direction, while the angular error has a zero mean with a standard deviation of σ_a radians. In the Monte Carlo simulation, the noise strengths (σ_p and σ_a) vary from $10^{-3.5}$ to 10^{-1} which are selected based on the typical noise level from cameras. For example, the focal length of camera used in the popular KITTI dataset Geiger et al. [2012] is 718 pixels, and most of the optical flow error is less than 3 pixels Menze et al. [2018], which corresponds to an angular error of $10^{-2.38}$ radians.

The Monte Carlo simulation are performed 1000 times for each noise level and MIGE is compared to the probability grid (PG) method, Gaussian mixture model (GMM) filter, particle filter (PF) and extended information filter (EIF). For the probability grid, the grid is chosen with x-range between -30m and 60m, and y-range between -60m and 30m, with a resolution of 0.5m in both directions. For the GMM method, We use 4 Gaussians to approximate each measurement likelihood. 1000 particles are used for the particle filter.

For a visual comparison between the approximated probability density of grid-

based method and our method, we computed the likelihood at discretized state space as shown in Figure 3.5. From the figure, we can see that our approximated density is slightly wider compared to the grid-based one. This is caused by the design of our estimator to ensure the true target is located within the $\chi \leq 3$ bound of the uncertainty, and thus ensuring the consistentency.

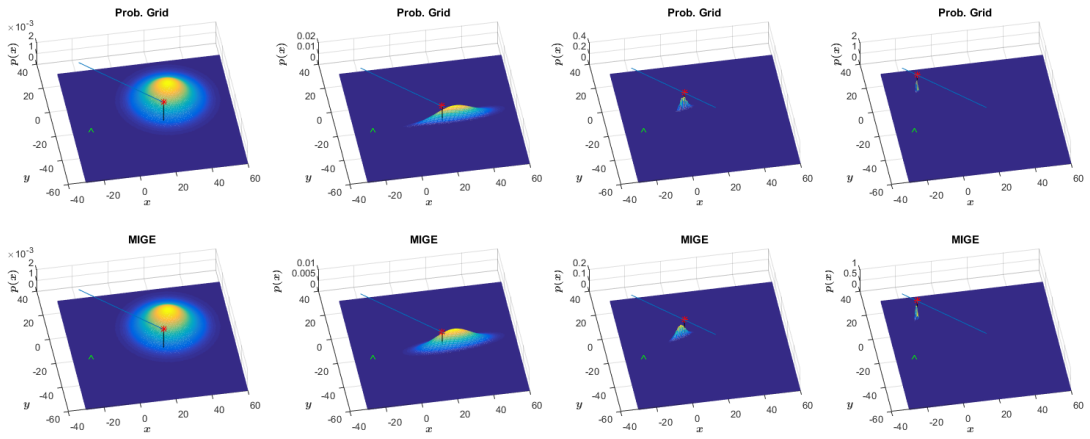

Figure 3.5: The evolution of the uncertainty for 2D bearing-only localization of a moving target. The simulated bearing has a zero-mean Gaussian noise with a standard deviation of 0.0316 radians. Top row is for the PG method, and the bottom row is for MIGE. From left to right: Probability density at discrete time 0 (prior), 1, 10 and 22.

The error metric chosen to evaluate the accuracy is the root-mean-squared error (RMSE). We evaluate the estimator consistency by using the normalized estimation-error-squared (NEES) as proposed by Bar-Shalom et al. [2004]. The expected value of the NEES for a consistent estimator should be equal to the dimension of the state vector (*i.e.* 2 for 2D case). We also evaluated the average computational time per measurement, and the required number of parameters to store the state density for each method. The results are presented in Figure 3.6.

From Figure 3.6(a), it can be observed that, at low to medium measurement and process noise, the MIGE outperforms all other methods in terms of accuracy, although MIGE achieves a similar result to particle filter at high noise level, while being slightly worst than probabilistic grid method. It should be mentioned that the accuracy of the probability grid method is limited by the resolution of the grid being used. Particle filter has a weakness of the so-called sample impoverishment problem Arulampalam et al. [2002]. Extended information filter sometimes diverges with low sensor noise, which is due to the inconsistency of the filter.

Gaussian mixture model (GMM) showed similar accuracy to MIGE at low noise level which can be observed in Figure 3.6(a). This can be explained by the resemblance between the two methods where the true underlying likelihood is explicitly approximated.

The primary difference is that MIGE uses a single degenerate Gaussian compared

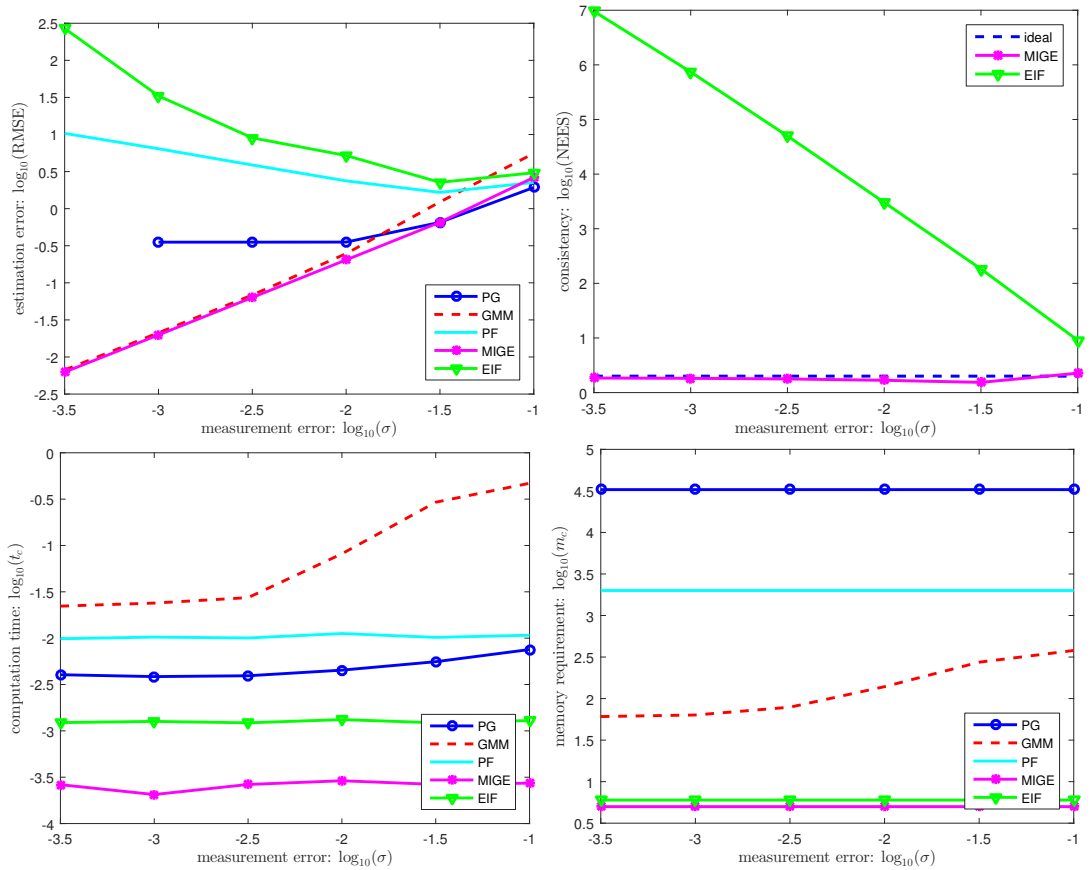

Figure 3.6: Monte Carlo simulation results for 2D tracking using bearing-only measurements. From top left to bottom right: (a) Root-mean-square error (RMSE); (b) Consistency evaluation ("ideal" is the consistent value); (c) Average computational time per measurement (t_c in seconds); (d) Average memory requirement to represent state likelihood function (m_c). "PG" is the probability grid method, "GMM" is Gaussian sum, "PF" is particle filter, "MIGE" is our Minimal Iterative Gaussian Estimator, and "EIF" is extended information filter.

to a set of Gaussians, leading to significant saving in terms of the memory requirement and the computational complexity. Figure 3.6(c)(d) shows the enhanced accuracy with small computational and memory requirement at high level noise, whilst the GMM method shows a noticeable increase in the computational and memory requirements. This is due to the difficulty in fusing multiple Gaussians at high level of noise which causes the increase in the number of Gaussians.

3.6.2 3D Bearing-only Localization

We also evaluated the performance of 3D bearing-only localization using Monte Carlo simulations. The sensors are assumed to be in stereo configuration, where the separation between them is 1 unit distance. The 3D points are randomly generated at 20 unit distance (depth) in front of the sensors, with the maximum x and y

upper bounded by the field of view of the camera used in KITTI datasetGeiger et al. [2012]. The sensor noise is defined to be on the image plane, with the covariance matrix

$$\boldsymbol{P}_{\pi} = \sigma^2 \boldsymbol{R} \begin{bmatrix} \beta & 0 \\ 0 & 1-\beta \end{bmatrix} \boldsymbol{R}^T, \tag{3.36}$$

where σ varies from $10^{-3.5}$ to 10^{-1}, $\beta \sim \mathcal{U}(0,1)$, and $\boldsymbol{R}$ is 2D rotation matrix at angle $\theta \sim \mathcal{U}(0,\pi)$.

Similar to the 2D case, we perform 10000 runs for each value of σ. The accuracy is evaluated using the RMSE, while estimator consistency is evaluated using NEES (ideal value is 3). The Monte Carlo simulation results are shown in Figure 3.7.

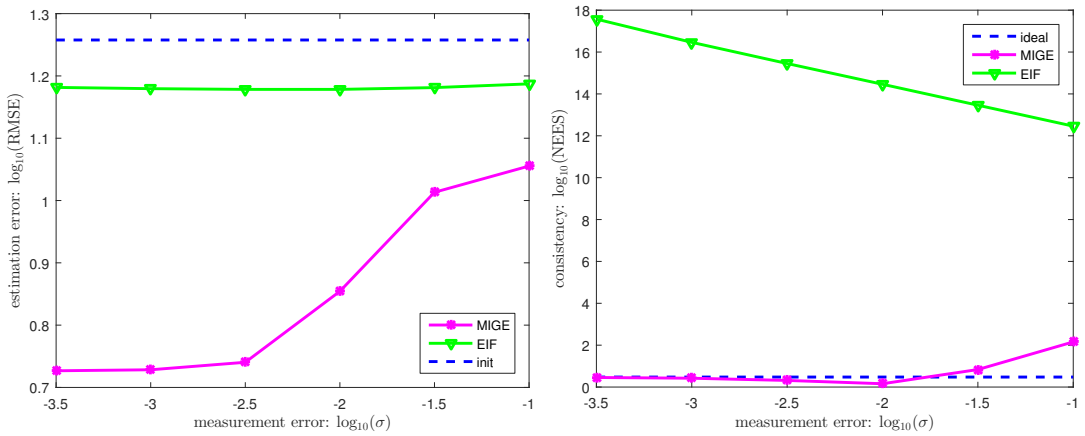

Figure 3.7: Monte Carlo simulation results for 3D triangulation using bearing-only measurements. From left to right: (a) Root-mean-square error (RMSE); (b) Consistency evaluation ("ideal" is the consistent value). "MIGE" is our Minimal Iterative Gaussian Estimator, and "EIF" is extended information filter.

From Figure 3.7, we can see that our method outperforms EIF method in both the accuracy and the uncertainty estimation. The EIF underestimates the uncertainty of the estimation, causing most estimates to have a large normalized estimation error. This will affect subsequent computation and fusion of the 3D point estimate. The average computational time of MIGE is $1.17 \times 10^{-4} s$ per measurement, while EIF is $8.26 \times 10^{-4} s$ per measurement.

3.6.3 Range-only Localization

Existing wireless radio used to obtain range measurements typically has an error between $1m$ to $3m$ Lanzisera et al. [2011]Kotaru et al. [2015]Rea et al. [2017]. Thus, we evaluate the performance of our estimator in 2D range-only target tracking using Monte Carlo simulation with measurement error from 10^0 to $10^{0.5} m$. The same sensor and target configuration as the 2D bearing-only localization is used. The results are presented in Figure 3.8.

From Figure 3.8, it can be observed that MIGE is more accurate than EIF. For MIGE, the decrease in measurement noise results in decreasing estimation error.

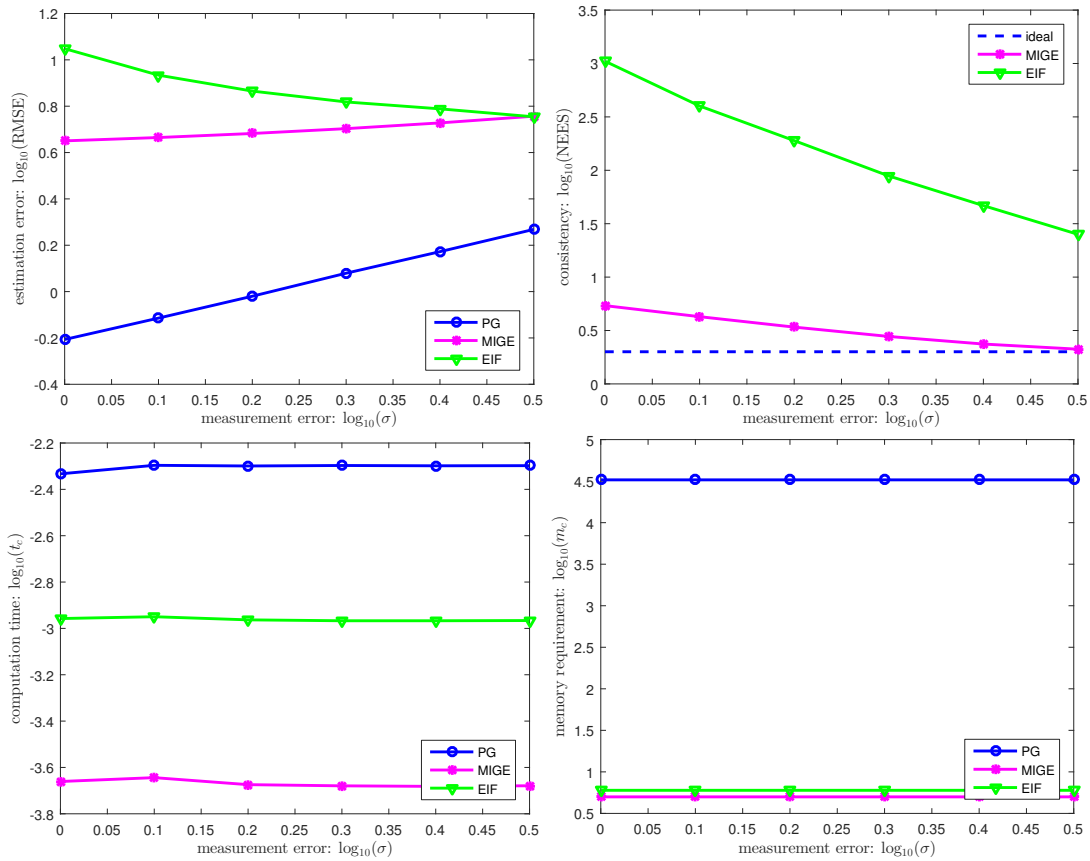

Figure 3.8: Monte Carlo simulation results for 2D tracking using range-only measurements. From top left to bottom right: (a) Root-mean-square error (RMSE); (b) Consistency evaluation ("ideal" is the consistent value); (c) Average computational time per measurement (t_c in seconds); (d) Average memory requirement to represent state likelihood function (m_c). "PG" is the probability grid method, "MIGE" is our Minimal Iterative Gaussian Estimator, and "EIF" is extended information filter.

However, the estimation error of EIF increases with a decrease in measurement noise due to the estimator becoming inconsistent. Despite the improvement in accuracy compared to EIF, MIGE accuracy is still not as accurate as the probabilistic grid method. This is due to the large initial error used and the poor geometry of the problem, where the tangential line to the circular constraint is almost parallel with the motion of the target. This causes the uncertainty ellipse to be unable to shrink significantly enough, where the measurement likelihood function within the prior cannot be sufficiently captured by a single degenerate Gaussian. Existing literatures avoid the poor geometry by using more than one sensors Rea et al. [2017] or combination of different measurementsKotaru et al. [2015]. This will alleviate the problem we observed, and allows more accurate target tracking performance using MIGE.

3.6.4 Computational Complexity and Consistency

The number of parameters required to represent a Gaussian distribution with N degrees of freedom is $(N^2 + 3N)/2$. In comparison, traditional Kalman filter and information filter require $N^2 + N$ parameters to describe the same distribution. The computation of the Mahalanobis distance is also more computationally demanding when expressed in the representation used by Kalman or information filter, which will be useful for goodness of fit test. If the fused mean is too far away (in terms of Mahalanobis distance) from the prior, then the measurement is treated as an outlier. The identified outlier measurements are discarded to avoid the state estimation from being corrupted.

From the χ^2 distribution, we can determine the probability of the estimated mean to agree with a given density. For example, a random variable with 2 degrees of freedom (2D case) has a 99.97% of chance to have a Mahalanobis distance less than 4, while a random variable with 3 degrees of freedom (3D case) has a 99.89% chance to have a Mahalanobis distance less than 4. Thus, we can conclude that if the computed Mahalanobis distance is greater than 4, the chances of the two densities describing the same variable is less than 0.03% for 2D case, or less than 0.11% for 3D case. Depending on the desired level of confidence, a different threshold on the Mahalanobis distance can be selected to discard outlier measurements.

MIGE also shows the improved consistency which can be observed from Figure 3.6(b), Figure 3.7(b) and Figure 3.8(b). As discussed by Bar-Shalom et al. [2004], assuming the likelihood function follows a Gaussian distribution, the average normalized estimation error squared (NEES) of a consistent estimator should be equals to the dimension of the state vector. This is achieved by ensuring that the approximated density encloses the true measurement likelihood within the current state prior.

It is in contrast to the EKF method, where linearisation of the measurement model is purely performed at the estimated mean, disregarding the current state uncertainty.

3.7 Summary

In this chapter, a degenerate Gaussian is proposed to approximate the nonlinear likelihood functions arising from the bearing-only and range-only localization problems. An efficient *Minimal Iterative Gaussian Estimator* utilising the approximated likelihood function is fomulated with a new parametrization method. Monte Carlo simulations showed enhanced performance in terms of accuracy, consistency and computational complexity when compared to existing techniques. This is a consequence of the efficient approximation of the likelihood functions.

Chapter 4

Bayesian Radio-Based Localization

This chapter presents a novel, accurate, *measurement-wise* recursive method of stationary, LOS target (emitter) 2D localization, using time-difference-of-arrival (TDOA) and frequency-difference-of-arrival (FDOA) measurements from multiple, localized sensor pairs. The method is based on our minimal iterative Gaussian estimator (MIGE) presented in Chapter 3. Due to the erroneous measurement, we also utilize the estimated uncertainty to make the method more robust against outliers.

The contributions of this work are:

- Based on MIGE, an efficient approximation of the nonlinear constraints are proposed for TDOA and FDOA localization. The hyperbolic constraint of TDOA and the pseudo-bearing constraint of the FDOA are approximated using a degenerate Gaussian.
- A closed-form analysis of the sensing geometry and uncertainty are also conducted. This helps in designing a consistent estimator, which ensures the correct convergence of the localization result as the number of measurements increases.

The chapter is organised as follows. Section 4.1 discusses some related work. Section 4.2 presents the degenerate Gaussian likelihood used to approximate the measurement probability density. Section 4.3 and 4.4 describe the parametrisation for time-difference-of-arrival (TDOA) and frequency-difference-of-arrival (FDOA) localization. Then, the experimental results are discussed in Section 4.6, followed by a simple overview of the propose method in Section 4.5. The chapter finishes with a summary in Section 4.7.

4.1 Related Work

TDOA localization is an active research area with many published papers. Some of the earliest work [Carter, 1981; Schau and Robinson, 1987] study the passive source localization using acoustic and radar sensors. The seminar paper by Carter [1987], presented a maximal likelihood estimator for the time delay between the signal received at two sensors. Some more recent works include: TDOA-based tracking of

a mobile emitter [Han et al., 2010; Miao et al., 2014], hybrid TDOA localization systems [Wei and Yu, 2016; Yin et al., 2016], TDOA-based localization under non-line-of-sight (NLOS) conditions [Wang et al., 2016] [Li et al., 2015], and so on. These research papers were built on past works related to stationary, line-of-sight (LOS) emitter localization using TDOA measurements. [Chan and Ho, 1994] proposed a closed-form, batch processing method to localize an emitter (signal source) using TDOA measurements. Their proposed method approximates the maximum-likelihood estimator and was shown to attain the Cramer-Rao lower bound in the small error region. A MATLAB implementation of Chan and Ho's closed-form method is available on University of Missouri website [Sun and Ho, 2010]. However, most robotics systems require real-time localization, which favours recursive method as opposed to methods that rely on batch processing.

In a recent paper, Choi *et al.* [Choi et al., 2013] proposed a TDOA localization estimator under the Robust Least Square (RoLS) estimator framework, which is recursive in time. This means that a set of measurements taken at a different time can be recursively combined to produce a more accurate localization result. However, the method proposed by Choi *et al.* is still restrictive in the sense that their method requires each set of measurements (taken at one time instance) to contain at least the minimum number of measurements from different sensors pair to obtain a unique localization solution (*i.e.* 3 for unique 2D TDOA localization).

A more desirable property of an estimator is the ability to update the location estimate using individual measurements. This is what we term a *measurement-wise recursive estimator*. *Measurement-wise recursive estimator* is beneficial because the target can still be tracked even if all but two sensors broke down during operation. It is also possible to save cost by using only a pair of mobile sensors instead of requiring multiple (at least 4 separate) sensors for 2D target localization. Kalman Filter [Kalman, 1960] and its variants (*e.g.* Extended Kalman Filter (EKF) and Unscented Kalman Filter (UKF) [Julier and Uhlmann, 2004]) are some well-known *measurement-wise recursive estimators*. The performance of TDOA localization using EKF and UKF are compared in [Fletcher et al., 2007].

There are also numerous works on joint TDOA-FDOA localization [Fowler and Hu, 2008; Musicki et al., 2010; Yeredor and Angel, 2011]. Recently, Wei and Yu [2016] proposed the use of a combination of a stationary and a mobile sensor to localize a stationary target. The localization is done by measuring the TDOA and FDOA from the pair of sensors. Following these assumptions, we apply our MIGE method to obtain an improved location estimator when compared to the existing methods.

There are also works based on motion pattern recognition in combination with Kalman Filter to track a moving emitter that does not satisfy the constant acceleration assumption [Han et al., 2010]. A more recent work is on localizing a source moving on a plane with Doppler-effect elimination and source plane scanning [Miao et al., 2014]. More recently, Zhong et al. [2016] uses a sequence of TDOA-FDOA measurements to detect and track multiple targets.

Time-difference-of-arrival (TDOA) and frequency-difference-of-arrival (FDOA) measurements are obtained by generalised cross-correlation of the received signal, or

from the cross-spectral density function in the frequency domain. Multi-path effects may cause the cross-correlation to have multiple peaks. [Rappaport et al., 1996] suggested the selection of the largest or the first peak as the measurement. Wei and Yu [2016] proposed a way to remove some outliers from the measurements. However, even with the proposed methods, outliers may still be present in the measurement. Thus, estimators need to be robust against outliers, which may otherwise corrupt the location estimate.

A lot of existing TDOA localization [Choi et al., 2013; Xu et al., 2015; Wei and Yu, 2016] and hybrid TDOA-FDOA localization [Musicki et al., 2010; Yeredor and Angel, 2011; Wei and Yu, 2016] works only consider two-dimensional (2D) localization, where the height of the target emitter is assumed known, or at the same height as the sensors. In our work, we also consider the 2D localization case, where the height of the sensor and emitter are assumed known, but can have different value.

4.2 Degenerate Gaussian Likelihood

Similar to our minimal iterative Gaussian estimator (MIGE), the main idea behind our new method is the approximation of the underlying probability distribution with a degenerate Gaussian likelihood function. This is supported by the fact that TDOA and FDOA constraints can be approximated by straight line constraints within a small region. An example with real data is shown in Figure 4.1.

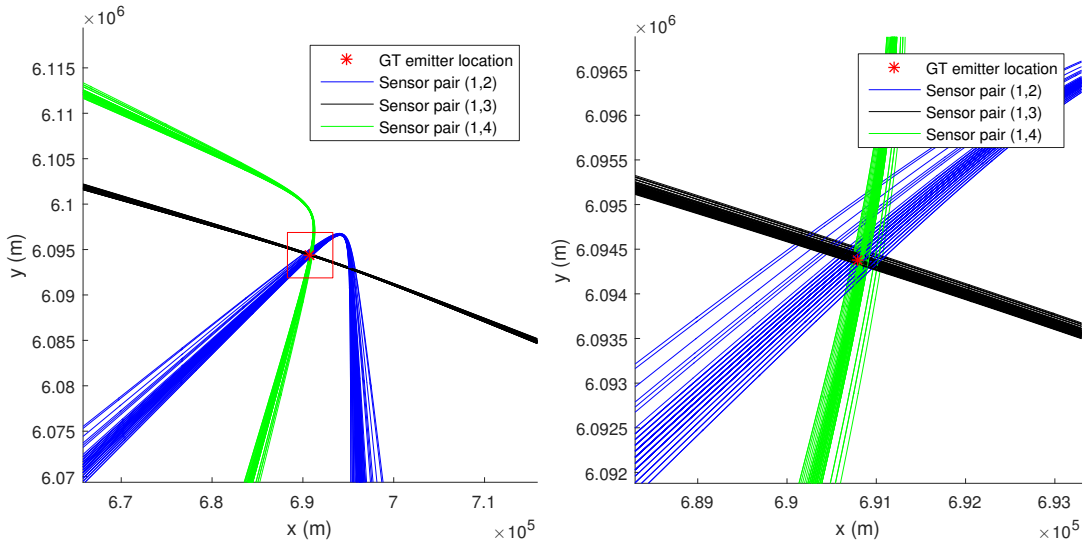

Figure 4.1: Plotted hyperbolic curves (blue, green and black lines) of real TDOA data. From left to right: (a) within a large local region (50σ bound), (b) within a small local region (5σ bound of prior used in our experiment), which looks very close to being straight lines. Red star corresponds to ground truth emitter location.

The equation of a 2D degenerate Gaussian likelihood is expressed as

$$\mathcal{G} = \exp\left(-\frac{d^2}{2\sigma^2}\right), \tag{4.1}$$

where exp is the natural exponential function, d is the distance from the mean (with highest probability), σ is the standard deviation of the Gaussian likelihood.

We compute the Gaussian function with degeneracy along a straight line as follows. We know that the minimum distance of an interest point (possible emitter location) at coordinates (x, y) from a straight line with parameters (a, b, c) can be calculated as

$$d_{min} = \frac{|ax + by + c|}{\sqrt{a^2 + b^2}}. \tag{4.2}$$

By substituting (4.2) into (4.1), the degenerate Gaussian likelihood centred around points on the straight line is then

$$\mathcal{G}(x, y|a, b, c, \sigma^2) = \exp\left(-\frac{(ax + by + c)^2}{(a^2 + b^2)(2\sigma^2)}\right). \tag{4.3}$$

Note that the volume under a degenerate Gaussian is not well defined, and so we use a likelihood function without the normalisation constant.

Similar to MIGE, we propose to parametrise a 2D Gaussian likelihood by storing only the polynomial coefficients of the exponential power (ignoring the constant term), with a vector $\mathcal{P}$ of length 5, such that

$$\mathcal{P}^T\boldsymbol{\chi} = [p_1, p_2, p_3, p_4, p_5] \begin{bmatrix} x^2 \\ x \\ y^2 \\ y \\ xy \end{bmatrix}. \tag{4.4}$$

Consequently, expanding the exponential power in (4.3) and following (4.4),

$$\mathcal{P} = -\begin{bmatrix} a^2 \\ 2ac \\ b^2 \\ 2bc \\ 2ab \end{bmatrix} / ((a^2 + b^2)(2\sigma^2)). \tag{4.5}$$

By assuming that all measurements are independent of each other, the likelihood is updated based on the Bayes' theorem. We know that the multiplication of two Gaussian functions is another Gaussian function, and that the multiplication of exponentials with the same base is simply the sum of their corresponding power (e.g. $\exp(x)\exp(y) = \exp(x + y)$). Thus, the likelihood update is done by simple addition in the parametric form. The uncertainty σ can also be scaled by a scalar product.

Since the likelihood is a Gaussian function, we can recover the mean from the

parametric vector $\mathcal{P}$ by computing the single critical point of the function (4.4), such that

$$\begin{bmatrix} \hat{x} \\ \hat{y} \end{bmatrix} = \begin{bmatrix} 2p_2p_3 - p_4p_5 \\ 2p_1p_4 - p_2p_5 \end{bmatrix} / ({p_5}^2 - 4p_1p_3) \tag{4.6}$$

Note that this is only well defined for non-degenerate Gaussian likelihood, where the denominator of a degenerate Gaussian is zero.

From the parametric $\mathcal{P}$ vector, we can also recover the uncertainty and bounds where the emitter may be located. This bound allows us to identify the best section of the nonlinear measurement curve to approximate with a tangent straight line, based on the current estimated uncertainty. It also allows the rejection of outlier measurements (where the measurement's curve lies outside of the bound). This is illustrated in Figure 4.2.

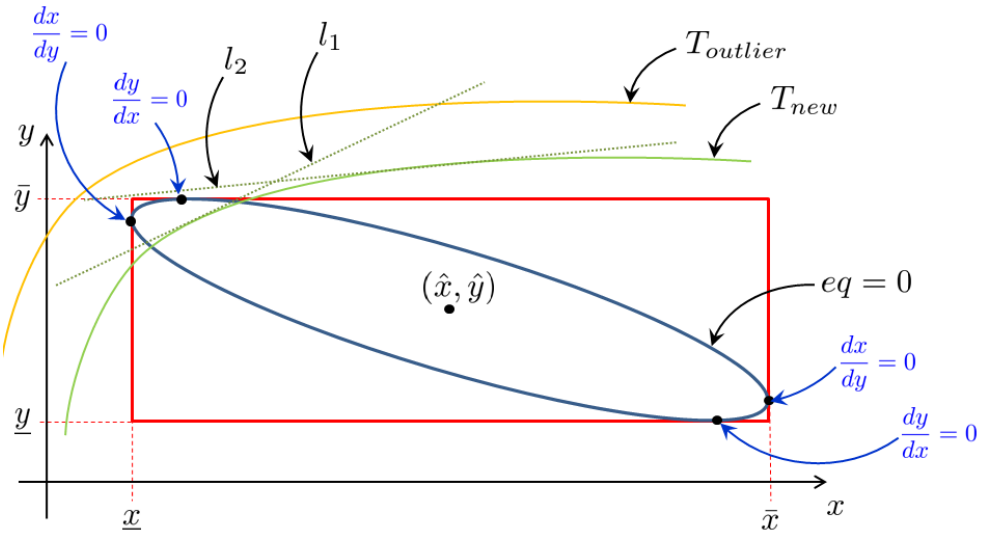

Figure 4.2: Illustrative figure showing the maximum-likelihood estimate (denoted by $(\hat{x}, \hat{y})$), the uncertainty ellipse (defined by $eq = 0$) of the previous estimate or prior (2D Gaussian distribution), the rectangular bounding box (defined by upper and lower bound on x and y), and a nonlinear measurement constraint (denoted by T_{new}). The bounding box can help in locating the correct section of the curved constraint that most satisfies the uncertainty of the prior, which is then approximated by a tangential straight line (i.e. l_1 instead of l_2). The bounding box can also be used to ignore outlier measurements (e.g. $T_{outlier}$), when the curve lies outside of the bounding box. The points on the elliptical bound that intersect the rectangular bound satisfies either $\frac{dy}{dx} = 0$ or $\frac{dx}{dy} = 0$ as denoted in the figure.

In order to find the 5 standard deviation (elliptical) bound (from χ^2 distribution, 99.9996% of the time, true emitter location is within this bound), we substitute $d = 5\sigma$ into (4.1), and solve for $eq = 0$, where

$$eq \triangleq \mathcal{P}^T \boldsymbol{\chi} + \frac{25}{2}. \tag{4.7}$$

We further simplify the computation by using a rectangular bounding box that encloses the elliptical bound, which is fully defined by the upper and lower bounds on x and y (Figure 4.2). The upper and lower bound of y are calculated as follows.

$$\overline{y} = \frac{-b_1 + \sqrt{{b_1}^2 - 4a_1c_1}}{2a_1}, \quad \underline{y} = \frac{-b_1 - \sqrt{{b_1}^2 - 4a_1c_1}}{2a_1} \tag{4.8}$$

where

$$a_1 = -\frac{{p_5}^2}{4p_1} + p_3, \quad b_1 = -\frac{p_2p_5}{2p_1} + p_4, \quad c_1 = -\frac{{p_2}^2}{4p_1} + p_6 + \frac{25}{2}$$

Similarly, the upper and lower bound of x are calculated as follows.

$$\overline{x} = \frac{-b_2 + \sqrt{{b_2}^2 - 4a_2c_2}}{2a_2}, \quad \underline{x} = \frac{-b_2 - \sqrt{{b_2}^2 - 4a_2c_2}}{2a_2} \tag{4.9}$$

where

$$a_2 = -\frac{{p_5}^2}{4p_3} + p_1, \quad b_2 = -\frac{p_4p_5}{2p_3} + p_2, \quad c_2 = -\frac{{p_4}^2}{4p_3} + p_6 + \frac{25}{2}$$

The first step of our method then becomes a tangent straight line fitting of the nonlinear measurement constraint within the uncertainty bound. This defines the values of $[a, b, c]$, where $ax + by + c = 0$ is the equation of the tangent line. The following sections explain the choice of standard deviation σ for TDOA and FDOA localization.

4.3 TDOA parametrisation

The time-difference-of-arrival (TDOA) equation is represented as

$$\tau_{ij,k} = \frac{1}{c}(r_{i,k} - r_{j,k}), \tag{4.10}$$

where c is the signal propagation speed (speed of light for radio signal, $r_{i,k}$ and $r_{j,k}$ are the distance of sensor i and sensor j from the target emitter respectively.

As previously mentioned, the TDOA hyperbolic curves are approximated with tangential straight lines. We also analyse the different "spread" of the uncertainty due to the relative placement of the sensors and the emitter, represented using the *sensor-target geometry* factor. This is explained as follows.

The two graphs in Figure 4.3 shows different TDOA hyperbolic curves plotted with the same step-size for τ. Two interesting features of the sensor target geometry factor emerged upon analysing the two graphs. First, it was clear that the "spread" (distance between lines) is larger in sensor pair $(2, 1)$ compared to sensor pair $(3, 1)$, even though the step-size used for τ is the same for both plots.

Second, it was also noted that the "spread" is larger when the emitter is farther away from the sensors, or when the emitter is not between the two sensors and almost co-linear with the line joining the two sensors.

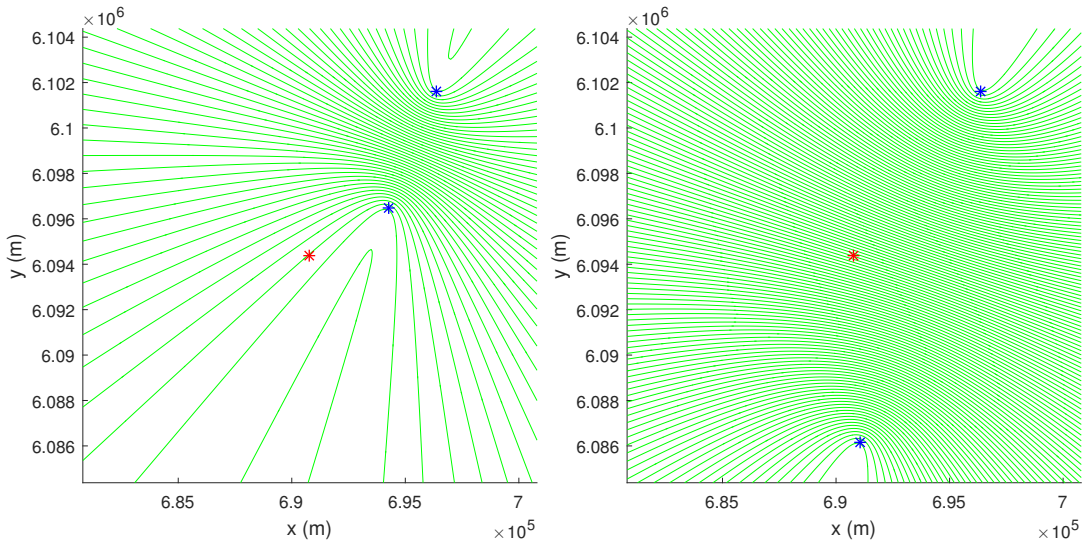

Figure 4.3: TDOA hyperbolic curves plotted with the same step-size, showing different "spread" of uncertainty due to relative placement of sensors (blue stars) and emitter (red star). From left to right: (a) Sensor pair $(2,1)$; (b) Sensor pair $(3,1)$. Note that some of the vertex of the hyperbolas are not within the line joining the focal points (sensors position). This is due to the position of the actual focal points (sensor position) to be on a different 2D plane (height).

The *sensor-target geometry* factor is computed as follows. Let $[x_i, y_i, z_i]$ be the sensor i coordinates, $[x_j, y_j, z_j]$ be the sensor j coordinates, and $[x, y, z]$ be emitter coordinates.

We calculate the small change in x with respect to small change in $\tau_{ij,k}$ by first assuming y and z are constant, then differentiate (4.10) with respect to $\tau_{ij,k}$, we get

$$\frac{dx}{d\tau_{ij,k}} = \frac{r_{i,k}\, r_{j,k}}{r_{j,k}(x - x_i) - r_{i,k}(x - x_j)}. \tag{4.11}$$

Similarly, we calculate the small change in y with respect to small change in $\tau_{ij,k}$ by first assuming x and z are constant, then differentiate (4.10) with respect to $\tau_{ij,k}$, we get

$$\frac{dy}{d\tau_{ij,k}} = \frac{r_{i,k}\, r_{j,k}}{r_{j,k}(y - y_i) - r_{i,k}(y - y_j)}. \tag{4.12}$$

With these and assuming that the hyperbolic lines are parallel (valid for small change in τ), we can find the small shift in distance between the hyperbolas with a small change in $\tau_{ij,k}$ using simple trigonometry, such that

$$\frac{dD_{ij,k}}{d\tau_{ij,k}} = \left(\frac{dx}{d\tau_{ij,k}} \frac{dy}{d\tau_{ij,k}}\right) / \sqrt{\left(\frac{dx}{d\tau_{ij,k}}\right)^2 + \left(\frac{dy}{d\tau_{ij,k}}\right)^2}. \tag{4.13}$$

The magnitude $|\frac{dD_{ij,k}}{d\tau_{ij,k}}|$ is known as the *sensor-target geometry factor*. An example of the *sensor-target geometry factor* at different sensor coordinates is shown in Figure (4.4).

Each sensor pair has a bounded error characterised by the measurement error of the two sensors. This measurement error (in meters) can be calculated during the calibration stage. This is obtained from the standard deviation of each set of TDOA measurements (for each sensor pair), $\sigma_{m,ij}$, multiplied by the speed of light, c, where

$$\sigma_{d,ij} = c\,\sigma_{m,ij}. \tag{4.14}$$

Another factor that contributes to the inaccuracy of the approximation using degenerate Gaussian along a straight line comes from hyperbolic shape of the actual measurement likelihood. Using (4.2), the distance of each points on the hyperbola (within the prior uncertainty bound) can be computed. We improve the estimator's consistency by increasing the uncertainty of the degenerate Gaussian using the root mean squared error (RMSE).

Finally, combining the sensor uncertainty, *sensor-target geometry factor* and the deviation from straight line, the resulting magnitude of uncertainty σ in (4.5) for each sensor pair is computed as

$$\sigma = \left|\frac{dD_{ij}}{d\tau_{ij,k}}\right| \sigma_{d,ij} + \epsilon_h, \tag{4.15}$$

where ϵ_h is the RMSE of points on the hyperbola from the approximated straight line.

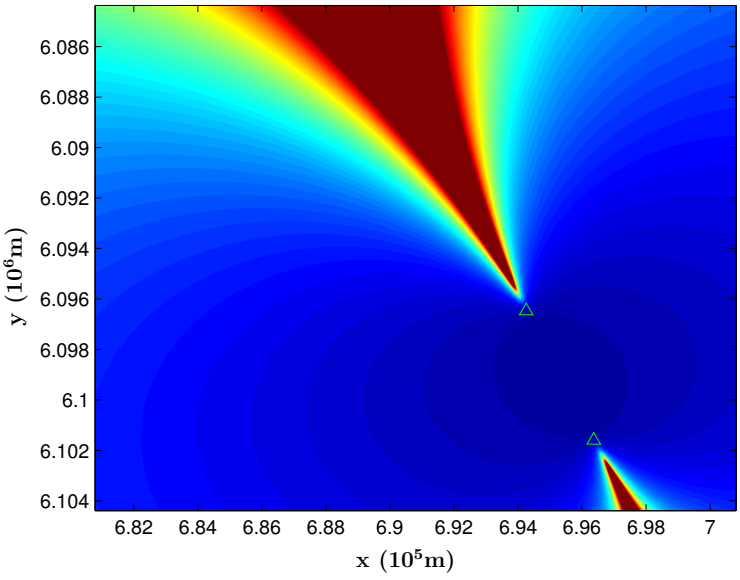

Figure 4.4: Sensor-target geometry factor plot around the TDOA hyperbolic curve at different locations (sensor pair $(2,1)$), where dark blue corresponds to small values, while dark red corresponds to high values.

4.4 FDOA parametrisation

The frequency-difference-of-arrival (FDOA) equation is represented as

$$\nu_{ij,k} = \frac{f_c}{c}((\boldsymbol{v}_{i,k})^T \boldsymbol{u}_{i,k} - (\boldsymbol{v}_{j,k})^T \boldsymbol{u}_{j,k}), \tag{4.16}$$

where c is the signal propagation speed (speed of light for radio signal), f_c is the carrier frequency, $\boldsymbol{v}_{i,k}$ and $\boldsymbol{v}_{j,k}$ are relative velocity between the emitter and sensors i and j respectively, $\boldsymbol{u}_{i,k}$ and $\boldsymbol{u}_{j,k}$ are unit vector pointing from sensor i and j to the emitter.

Wei and Yu [2016] proposed the use of a stationary and a mobile sensor to localize a stationary target. We analyse the same problem and found that the FDOA constraints simplifies to frequency-of-arrival (FOA).

Assuming sensor j is stationary, the second term of (4.16) is equals to zero, while the first term can be seen as the dot product between $\boldsymbol{v}_{i,k} = [v_x, v_y, v_z]$ and $\boldsymbol{u}_{i,k}$. The velocity of sensor i is obtained from GPS sensor, which does not provide velocity in the z axis. Thus, we assume the sensor is travelling at a constant height, where v_z is zero. In this work, we also assume the height for the emitter is known, but can be different from the height of the sensors.

We also know that the dot product between two vectors is equals to the product of their magnitude and cosine of the angles between them. This provides two bearing constraints, mirrored around the motion vector of sensor i. The two possible degenerate Gaussian likelihood will then be centred along one of the two lines

$$l_1 \;:\; (\tan(\angle_1))x + (-1)y + (y_i - \tan(\angle_1)x_i) = 0, \tag{4.17}$$
$$l_2 \;:\; (\tan(\angle_2))x + (-1)y + (y_i - \tan(\angle_2)x_i) = 0, \tag{4.18}$$

where the angles

$$\angle_1 = \arctan\left(\frac{v_y}{v_x}\right) + \theta, \tag{4.19}$$
$$\angle_2 = \arctan\left(\frac{v_y}{v_x}\right) - \theta, \tag{4.20}$$
$$\theta = \arccos\left(\frac{c\nu_{ij,k}}{f_c||\boldsymbol{v}_{i,k}||\cos(\arctan(\frac{z-z_i}{(x-x_i)^2+(y-y_i)^2}))}\right). \tag{4.21}$$

Similar to TDOA, the FDOA constraint is approximated using a degenerate Gaussian along a straight line, where the spread of the likelihood function is computed as follows.

First, we differentiate (4.21) to compute the small change in the angle θ with a small change in FDOA measurement ν. This is the *sensor-target geometry* for FDOA,

which is

$$\frac{d\theta}{dv_{ij,k}} = \frac{c}{\sqrt{\left(f_c||\boldsymbol{v}_{i,k}||\cos(\arctan(\frac{z-z_i}{(x-x_i)^2+(y-y_i)^2}))\right)^2 - (cv_{ij,k})^2}}. \tag{4.22}$$

Then, we check if the sensor i location is within the current uncertainty prior. If it is within the current uncertainty prior, the standard deviation of the angle will be higher. This is due to the fact that both bearing angles are possible as a constraint for the emitter location. On the contrary, if the sensor i is located outside of the prior uncertainty and only one angle is within the bound, then the standard deviation of the angle σ_θ will only be affected by the *sensor-target geometry* and the standard deviation of the FDOA measurement. This is illustrated in Figure 4.5.

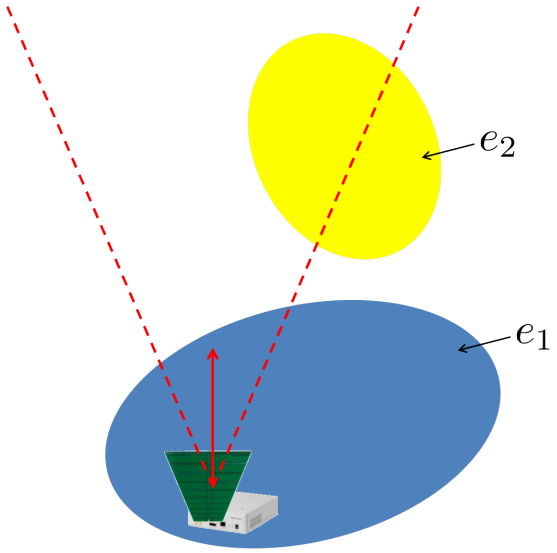

Figure 4.5: Two cases of interest for frequency-difference-of-arrival measurements. If the prior uncertainty ellipse is e_1, then the measurement likelihood is less certain because both bearing angles are possible. If the prior uncertainty ellipse is e_2, then there is only one bearing angle possible (within a specified confidence).

Finally, the standard deviation of the degenerate Gaussian σ is then

$$\sigma = \left(\left|\frac{d\theta}{dv_{ij,k}}\right|\sigma_v + \Delta_\theta\right) d_{max}, \tag{4.23}$$

where d_{max} is the maximum distance between the sensor i to the emitter (according to the current prior), and

$$\Delta_\theta = \begin{cases} 2\theta & \text{if sensor } i \text{ is inside the prior } (e_1 \text{ case}) \\ 0 & \text{otherwise } (e_2 \text{ case}) \end{cases} \tag{4.24}$$

4.5 Algorithm overview

The key steps of the new method may be summarised as follows.

Algorithm 4: TDOA-FDOA Localization

Data: Given initial estimate $\boldsymbol{x}_0$, covariance matrix $\boldsymbol{P}_0$, TDOA measurements $\{\tau_{ij,k}\}$, FDOA measurements $\{\nu_{ij,k}\}$
Result: Emitter location estimate in parametric form $\mathcal{P}_k$

```
Convert prior x_0 and P_0 into parametric form;
while time k is increasing do
    Find bounding box following (4.8) and (4.9);
    Find contour of nonlinear constraint;
    if contour outside bounding box then
        pass;
    else
        Compute tangent line;
        Compute sensor-target geometry factor;
        Compute parametric form of measurement likelihood;
    end
    Update by element-wise addition;
end
```

The proposed method is also simple and general enough to combine different types of measurements under the same framework (i.e. straight line approximation and parameter updating). For example, angle-of-arrival (AOA) measurement, receive signal strength (RSS), can be combined with TDOA measurements for improved localization accuracy.

4.6 Experimental results

We verify the performance of our proposed method by performing Monte Carlo simulation and real data experiments for TDOA and TDOA-FDOA localization.

4.6.1 TDOA localization

Monte Carlo simulations are performed to compare the performance of different TDOA Localization methods using the minimum number of measurements (i.e. 3 measurements from 4 different sensor pairs), at increasing measurement noise (Gaussian with zero mean, variance from $-50dB$ to $30dB$). For recursive methods, the initial estimate is initialised with an additive Gaussian Noise with zero mean and a variance of $43dB$ ($40dB$ in both x and y).

RoLS method from Choi et al. [2013] is a time-recursive method, where an initial estimate is given as a prior. However, in this experiment, only measurements from a

single time instance are used. Thus, we may consider it to be a batch process method with a prior.

RoLS is also modified to *measurement-wise recursive* method for a fair comparison with other methods like EKF, UKF and our newly proposed method.

In this section, Monte Carlo simulation results are presented to compare the performance of batch processing methods from Sun and Ho [2010], NoLS and RoLS from Choi et al. [2013]; and *measurement-wise recursive* methods like EKF, UKF, modified version of RoLS and our proposed method.

A sequence of real TDOA measurements is also collected and used in the later part of our experiment, to test the performance of our newly proposed TDOA localization method.

4.6.1.1 Good Geometry Monte Carlo Simulation

The first experiment involves good sensor-target geometry, where there is large separation between the sensors ($7 - 13km$), and the target (emitter) does not lie close to the curved part of the TDOA hyperbolas. Figure 4.6 shows the hyperbolas in a small local region, along with the location of sensors and target.

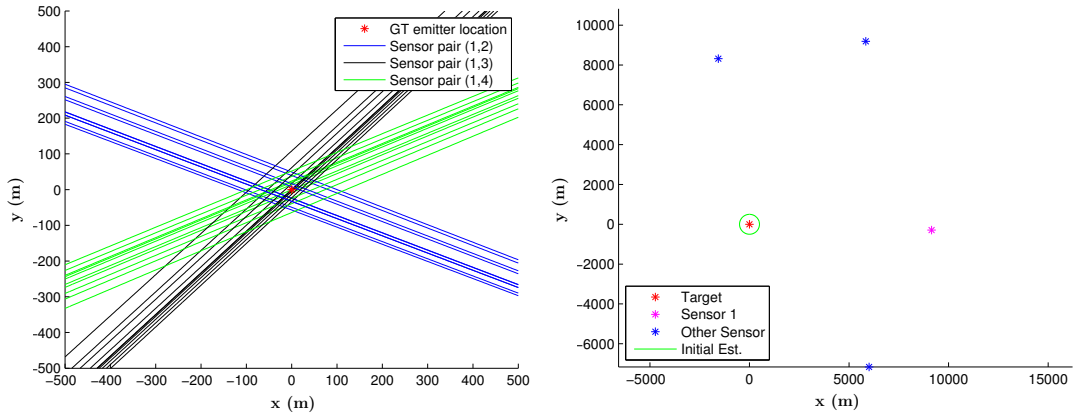

Figure 4.6: Plots of good geometry case. From top to bottom: (a) shows the 10 sets of TDOA hyperbolas in a small local region (5σ bound of prior uncertainty), which is observed to have a clear line intersect and the curves looks very close to being straight lines. (b) shows the location of the sensors, target and the initial uncertainty (5σ bound) used.

Figure 4.7 shows the Monte Carlo simulation results. From Figure 4.7, it can be observed that our newly proposed method outperforms EKF, UKF and *measurement-wise* (MW) *recursive* RoLS method in small measurement noise case, and stays close to the Cramer Rao Lower Bound (CRLB) line. It is also noted that at very low noise level ($10\log(c\sigma) < -30$), the batch processing method from Sun and Ho [2010] produces slightly better result than our proposed method.

It can also be observed that the MW recursive RoLS solution deviates from the initial $43dB$ location significantly when the measurement noise is low. This could be

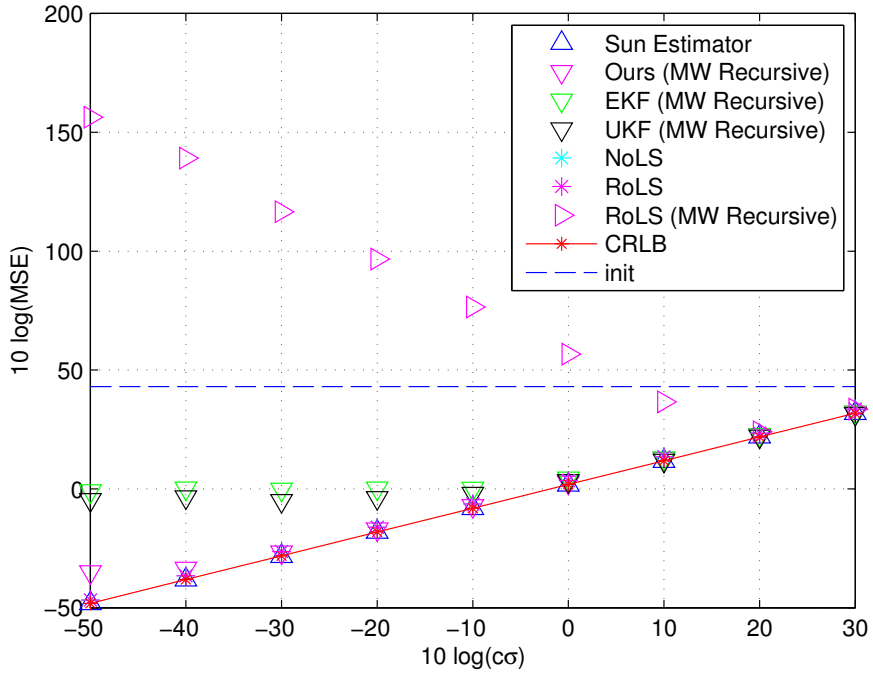

Figure 4.7: Monte Carlo simulation results for good geometry case. "Ours" is the new method we proposed. "MW Recursive" refers to *measurement-wise recursive* method. "init" refers to the initial uncertainty used as a prior. "MSE" is the mean squared error, while "$c\sigma$" is the TDOA measurement noise multiplied by the speed of light.

due to the poor conditioning of the matrix $\tilde{\boldsymbol{H}}_k$ and vector $\boldsymbol{z}_k$ (see Choi et al. [2013]) when only one measurement is used at each estimation step. When the noise level is low, instead of improving the estimation, the poorly conditioned equation drags the initial estimate farther away from the correct value.

Small measurement noise also corresponds to having a large amount of measurement data, which greatly favours our new method compared to other MW recursive methods.

On the other hand, other batch processing methods like Sun and Ho [2010], NoLS and RoLS Choi et al. [2013] perform similarly to each other. RoLS is considered batch processing here because only a single time instance is used in this experiment.

Interestingly, Sun and Ho Sun and Ho [2010] method performed slightly better when all measurements taken from different sensors are batch processed, compared to the more recent NoLS or RoLS. This could be due to Choi et al. Choi et al. [2013], similar to our method, uses the simplifying assumption that the noise from different sensor pairs is independent of each other.

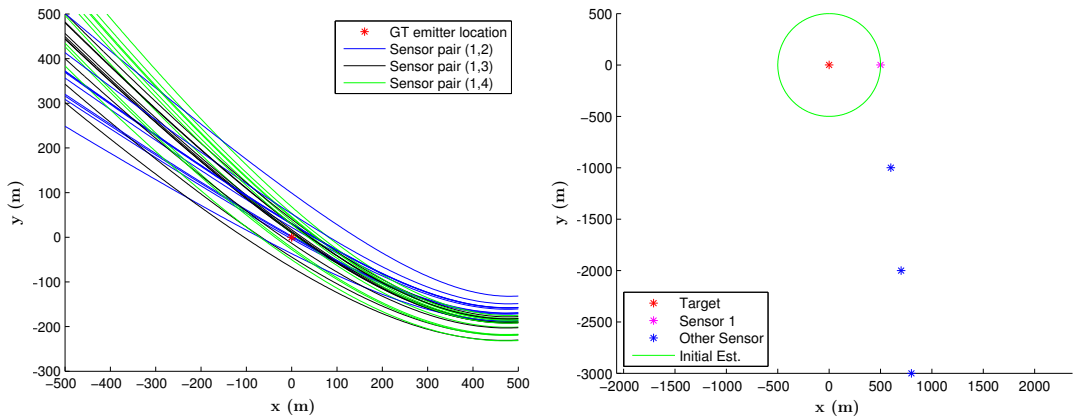

Figure 4.8: Plots of poor geometry case. From top to bottom: (a) shows the 10 sets of TDOA hyperbolas in a small local region (5σ bound of prior uncertainty), which is observed to have a poor line intersect and the curves does not look close to being straight lines. (b) shows the location of the sensors, target and the initial uncertainty (5σ bound) used.

4.6.1.2 Poor Geometry Monte Carlo Simulation

"Poor geometry" refers to sensors being close to each other and to the target (emitter) such that the hyperbolic lines are almost parallel to each other, and the hyperbolas cannot be well estimated by a simple straight line. Figure 4.8 shows the hyperbolas in a small local region, and location of sensors and target.

Figure 4.9 shows the Monte Carlo simulation results. From Figure 4.9, it can be observed that our newly proposed method is robust to poor sensor-target geometry and outperforms all other methods in most of the range of measurement noise tested. It is also the closest to the Cramer Rao Lower Bound (CRLB).

On the other hand, EKF and MW RoLS result diverge from the initial estimate (with $43dB$) at low noise level, while UKF and RoLS only achieved a small improvement from the initial prior provided. It is also noted that the MATLAB implementation of Chan and Ho's work Sun and Ho [2010] and NoLS cannot provide any useful localization result.

4.6.1.3 Real Data

Using four synchronised software define radios (SDRs) placed at known locations (GPS localized), we have collected 61 TDOA measurements per sensor pair. Figure 4.10(a) shows the location of the sensors with respect to the target (radio tower) location. Figure 4.1 shows the hyperbolas within the 5σ bound of the initial estimate (prior).

The measurement standard deviations are $56.5m$, $80.7m$, $55.2m$ for sensor pairs $(2,1)$, $(3,1)$ and $(4,1)$ respectively.

Figure 4.10(b) shows a few examples of the localization error trajectories over 61 sets of TDOA measurements, where each measurement set consists of one TDOA

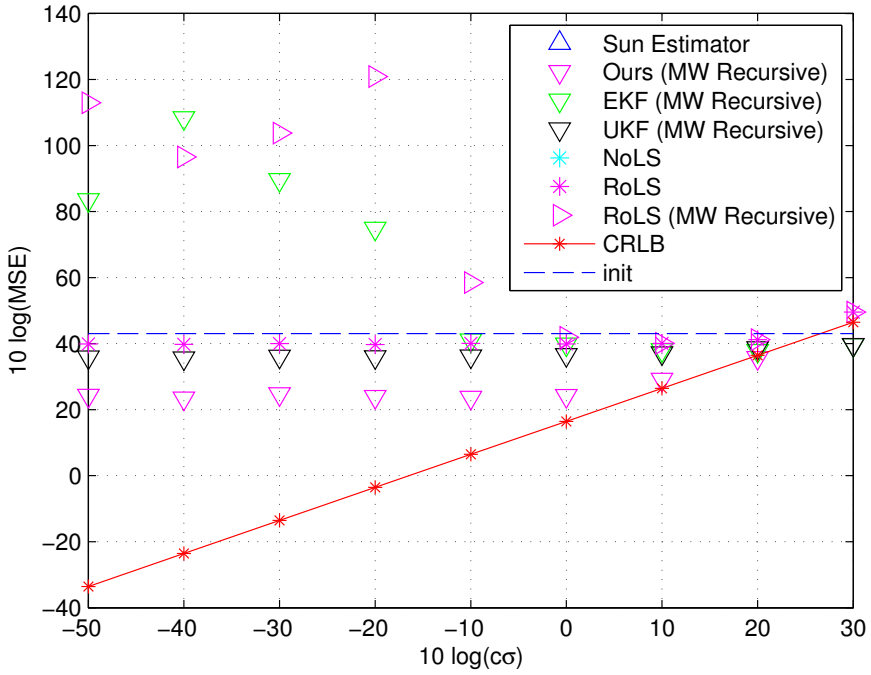

Figure 4.9: Monte Carlo simulation results for poor geometry case. "Ours" is the new method we proposed. "MW Recursive" refers to *measurement-wise recursive* method. "init" refers to the initial uncertainty used as a prior. "MSE" is the mean squared error, while "$c\sigma$" is the TDOA measurement noise multiplied by the speed of light.

measurement from each sensor pair $((2,1), (3,1)$ and $(4,1))$. The plot shows that the newly proposed method can successfully reduce the initial localization error of more than $1300m$ to a final localization error of less than $50m$.

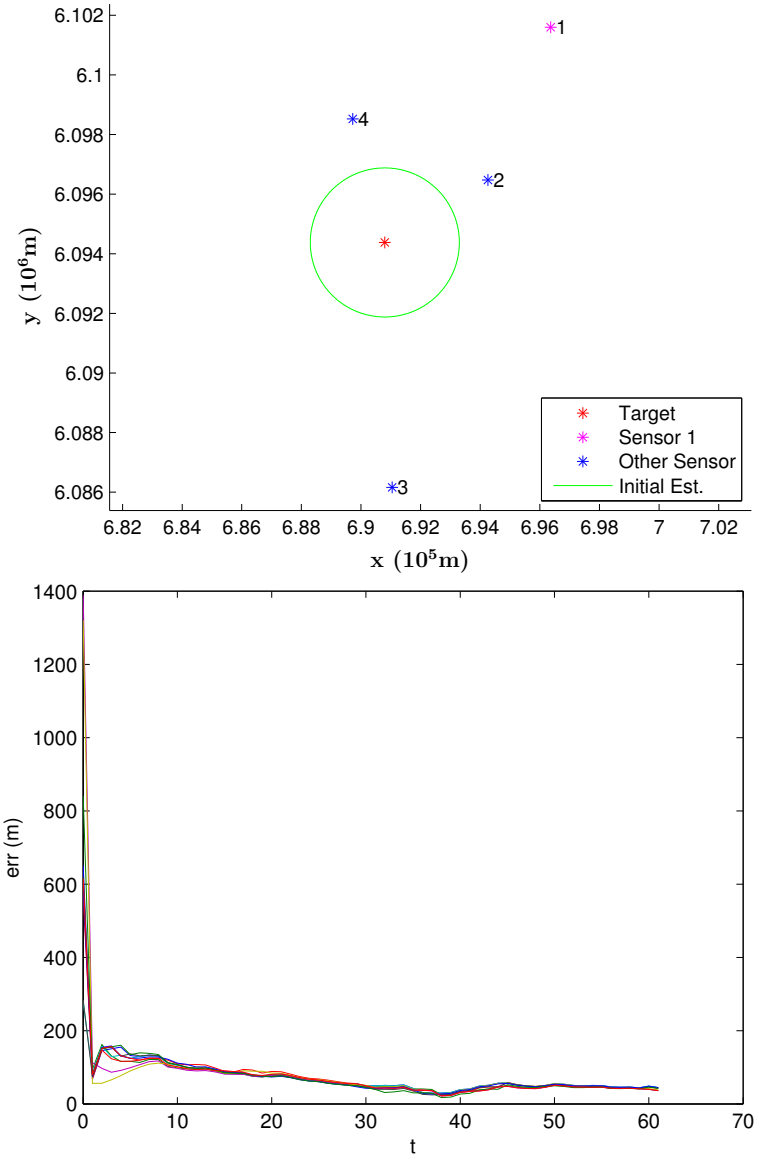

Figure 4.10: Plots of real data experiment. From top to bottom: (a) shows the location of sensors, target and initial uncertainty (5σ bound) for the real data experiment, (b) shows 10 estimation error trajectories (in meters) versus time. Each run is initialised with a prior (zero mean Gaussian around the true emitter location with standard deviation of $500m$). Note that time refers to different instances of measurements set (61 sets of measurements in total), and not the absolute time.

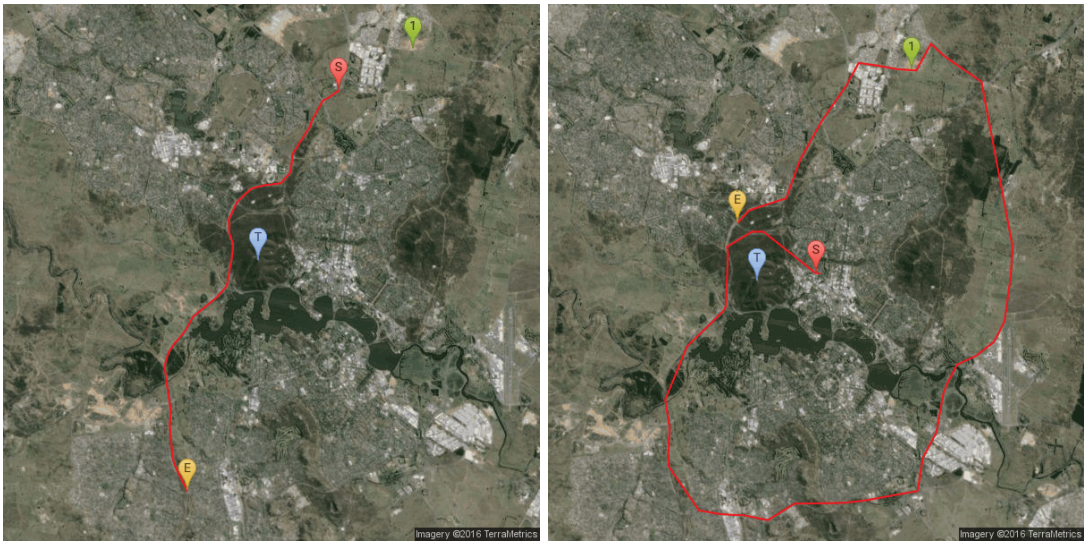

Figure 4.11: Two paths taken by the mobile sensor during TDOA-FDOA localization experiment (courtesy of Junming Wei [Wei and Yu, 2016]). From left to right: (a) Short path; (b) Long path.

4.6.2 TDOA-FDOA localization

We collect TDOA-FDOA measurements using two synchronised software define radios (SDRs), where one is placed at a fixed known location, while the other is placed on a car driven on two different paths. The paths are illustrated in Figure 4.11. We also collect measurements at two different frequencies, namely the FM signal at 106.3 MHz, and TV signal at 226.5 MHz.

The performance of our method is compared to the results obtained in Wei and Yu [2016], and is presented in Table 4.1.

Table 4.1: Performance comparison of our new method and existing methods. The error metric is the final localization error in meters.

	FM short	TV short	FM long	TV long
No. of TDOA & FDOA	26	26	131	98
RMSE EKF [Wei and Yu, 2016]	145 m	112 m	90 m	86 m
RMSE WLS [Wei and Yu, 2016]	249 m	64 m	62 m	80 m
New method	119 m	37 m	57 m	47 m
CRLB [Wei and Yu, 2016]	105 m	17 m	16 m	13 m

From Table 4.1, we can see that the newly proposed method outperforms the other methods, and is the closest to the theoretical Cramer Rao lower bound (CRLB). The new method even outperforms the weighted least square (WLS) method that jointly optimises all measurements. This can be attributed to the ability of our method to effectively approximate the measurement likelihood, and the ability to recursively discard outlier measurements that will otherwise corrupt the estimate.

4.7 Summary

This chapter proposes a novel, simple, memory efficient, robust and *measurement-wise* recursive time-difference-of-arrival (TDOA) and frequency-difference-of-arrival (FDOA) based algorithm for the localization of a stationary emitter on a known 2D plane. Similar to MIGE, the method approximates the nonlinear measurement likelihood using a degenerate Gaussian around a tangential straight line, with a variance that depends on both the measurement noise and the sensor-target geometry. The probability density is then updated by the simple element-wise addition. The proposed method is compared to well-known methods and shown to have superior performance. For TDOA only localization, we also show that the method is robust enough that the initial estimate (prior) can be improved even when the sensors are poorly placed, or when the curve cannot be well approximated by a straight line (Sec. 4.6.1.2). The method also inherently trims away outlier measurements based on the computed uncertainty. The method is also applied to real TDOA and TDOA-FDOA measurement data, which shows convergence to the correct emitter location.

Chapter 5

Bayesian Monocular Visual SLAM

This chapter discusses a new robust monocular simultaneous localization and mapping (SLAM) system to provide low drift visual odometry and 3D reconstruction result. We overcome the limitations of using sparse features in low/repetitive textured scenes for pose estimation, without relying on restrictive motion dynamics assumptions. This is accomplished by using dense optical flow with estimated uncertainty as the input to our visual odometry method. Combining with an existing robust SLAM back-end [Cheng et al., 2015], the method achieves significant robustness with respect to the sensing uncertainty and loop-closure outliers. The contributions of this work are threefold:

- An accurate dense optical flow with estimated uncertainty is proposed. Based on DCFlow [Xu et al., 2017], we improve the existing optical flow accuracy by including the epipolar constraint into the cost function used to compute the matching pixel. A principled method to estimate the uncertainty of a dense optic flow is also proposed, by fitting a bivariate Gaussian function to the matching cost. We named the method Bayes Dense Flow.
- A robust visual odometry method is presented. An efficient RANSAC sampling is used to select inlier correspondences based on their estimated uncertainty. We then propose a new Mahalanobis eight-point algorithm to estimate the inter-frame camera motion. These ensure that a robust camera motion is estimated from the dense correspondences, by applying appropriate weighting for each correspondence with respect to their corresponding uncertainty. The visual odometry method is further improved by fusing with the pose estimate obtained through the perspective-n-point (PnP) method, which utilizes a set of previously triangulated scene points.
- Based on the MIGE (Chapter 3), an efficient approximate Bayesian 3D triangulation method is proposed, which allows the 3D scene points to be reconstructed along with a measure of uncertainty. The triangulation method uses a minimal representation parametric form, which also allows for a simple recursive fusion of consecutive 3D scene points estimates. Combined with the existing SLAM back-end [Cheng et al., 2015], the method is applied for aerial navigation and mapping.

To the best of our knowledge, this is the first research work that achieves the robustness for both of the front-end and back-end SLAM, and its application for aerial navigation and mapping in an unstructured environment with dynamic objects.

The chapter is organised as follows. Section 5.1 discusses some related work. Section 5.2 presents the 3D scene points triangulation in greater details, where the measurement noise is defined on the image plane. Section 5.3 details the improvement to DCFlow and optical flow uncertainty estimation, Section 5.4 describes our new robust visual odometry, Section 5.5 covers the loop closure computation to further reduce visual odometry drift. Section 5.6 provides a quick overview of the proposed visual SLAM. Section 5.7 shows the experimental results that verify the performance of our proposed method, and Section 5.8 presents a summary of our work.

5.1 Related Work

Visual simultaneous localization and mapping (SLAM) refers to a method that uses visual sensor (*i.e.* camera) to estimate the pose of the camera (self-localization) and the location of landmark (environment mapping) at the same time. Visual SLAM is categorized based on the sensing modality used. Existing unmanned aerial vehicle (UAV) visual odometry and SLAM systems have used stereo camera [Hrabar et al., 2005; Heng et al., 2011], multi-camera [Yang et al., 2017], omni-directional camera [Hrabar and Sukhatme, 2003; Demonceaux et al., 2006], IMU-camera system [Cheviron et al., 2007; Wang et al., 2013; Weiss et al., 2013], IMU-camera-sonar [Chowdhary et al., 2013], GPS-IMU-camera system [Templeton et al., 2007], laser-IMU-camera system [Achtelik et al., 2009; Bachrach et al., 2009; Shen et al., 2011], external camera [Park et al., 2005; Klose et al., 2010], or a monocular downward-facing camera [Weiss et al., 2011b,a; Lee et al., 2011].

A monocular camera system is the most interesting due to the generality where only a single camera is required. The other benefits of a monocular system are the energy efficiency, low cost, light weight, easy installation and maintenance. In our work, we do not assume the IMU or GPS data is available, due to the difficulty to synchronise the sensors, and the often non-constant relative pose between the sensors with pan-tilt camera system. As previously mentioned, existing UAV monocular visual odometry methods primarily use a downward-looking camera [Herisse et al., 2008; Lee et al., 2011; Weiss et al., 2011b, 2013; Wang et al., 2013; Chowdhary et al., 2013] that simplifies the problem of visual odometry, as the estimation of the forward motion is known to be more error-prone [Song et al., 2016; Oliensis, 2005]. A downward looking camera also allows easier initialisation and tracking of features as the visible scene is assumed to be mostly planar [Wang et al., 2013; Caballero et al., 2009]. However, commercially available UAVs are usually equipped with a single front facing camera. This makes existing methods not suitable for such hardware setup. Using a front facing camera also makes teleoperation in a highly unstructured environment possible, as it allows the pilot to see and avoid obstacles. An existing visual odometry work that uses front facing camera assumes partially a structured

scene with known objects [Artieda et al., 2009], which makes their method unsuitable for general unstructured scenes.

On the other hand, most visual odometry methods suitable for large forward motion (*e.g.* [Klein and Murray, 2007; Song et al., 2016; Fanani et al., 2017]) use sparse feature points matched between images to compute the inter-frame motion. However, in some scenes, the sparsely matched features may be clustered around a small region of the image or encounter problems with planar degeneracy [Hartley and Zisserman, 2003]. This may result in an inaccurate motion estimate. Existing methods (*e.g.* [Bradler et al., 2015; Fanani et al., 2017]) improve the feature matching accuracy by also assuming that the motion dynamics of the vehicle/robot is known or calibrated. This makes their method not general enough to be directly applied to other vehicles with different motion dynamics. Some methods also use learning-based [Song et al., 2016] or convolutional neural network (CNN) trained [Fanani et al., 2017] ground height estimation. This again makes the method not suitable for general use when the ground surface is different from the training data.

Independent of visual odometry and SLAM research, optical flow has also achieved significant improvement. Currently, one of the most accurate optical flow algorithms suitable for large motion was proposed by Xu et al. [2017]. Similar to [Chen and Koltun, 2016], they compute the optical flow by operating directly on the four-dimensional cost volume. The data term they used is the dot product between the feature vectors, where the feature vector trained by a convolutional neural network (CNN) describes the local visual cue. The edge-aware spatial regularisation is enforced by adapting semi-global matching (SGM) [Drory et al., 2014] with structural edge detector (SED) [Dollár and Zitnick, 2015]. Forward-backward consistency was checked, where inconsistent matches are discarded. The semi-dense optical flow is then interpolated using the well known EpicFlow interpolation method [Revaud et al., 2015]. An extra postprocessing step fits homography to the computed flow field to improve optical flow estimate at low textured region that is roughly planar (*e.g.* ground). More recently, CNN, pyramidal feature extraction and feature warping have also been applied successfully to compute optical flow [Hui et al., 2018][Sun et al., 2018]. However, like most state-of-the-art optical flow methods, the uncertainty of the optical flow is not computed.

It was known that the information about the optical flow uncertainty is useful for later processing, where each flow vector can be appropriately weighted. Estimation of the optical flow uncertainty has been done using image gradient [Heeger, 1988][Simoncelli et al., 1991], which depends on the contrast of neighbouring pixels. Optical flow uncertainty has also been estimated from the min-marginal map of a dynamic Markov Random Field approach [Glocker et al., 2008]. However, their method only provides a very local estimation of the uncertainty, and is not suitable for more recent optical flow methods that utilize more advanced feature matching and regularization techniques. The bootstrap resampling method [Kybic and Nieuwenhuis, 2011] has also been proposed to estimate optical flow uncertainty, but it requires multiple iterations of the optical flow to be performed to accurately capture the uncertainty. More recently, Mac Aodha et al. [2013] propose to estimate the optical flow uncertainty

using learning methods, but this method requires extra training and does not reflect the actual uncertainty of a particular optical flow method. Probabilistic methods have also been used to compute the optical flow [Wannenwetsch et al., 2017][Piao et al., 2014], but the achievable optical flow accuracy is still lower than other methods. In comparison, our proposed method directly makes use of the full discrete cost volume of DCFlow [Xu et al., 2017], which allows simple incorporation of extra matching cost (epipolar) constraints to improve the optical flow accuracy, and direct estimation of the 2D uncertainty.

5.2 3D Scene Points Triangulation

Given the camera pose from the computed camera extrinsic (Section 5.4), we can obtain the 3D scene points by triangulation. We know that 3D points that are far away or has large uncertainty in their feature location is less reliable. Thus, we use our minimal iterative Gaussian estimator (MIGE) from Chapter 3 to triangulate 3D scene points along with estimating their uncertainty information.

In this section, we follow the convention used by computer vision community, and denote the measurement vector (image coordinate) as $\boldsymbol{x}_k$, and changed the notation for state vector to $\boldsymbol{\chi}_k$.

5.2.1 3D Bearing Measurement

Assuming the intrinsic parameters of the camera is calibrated, the nonlinear, bearing-only measurement model can be written as

$$\boldsymbol{x}_k = \boldsymbol{h}(\boldsymbol{\chi}_k) + \boldsymbol{w}_k \tag{5.1}$$

$$\begin{bmatrix} \hat{u}_k \\ \hat{v}_k \end{bmatrix} = \frac{1}{z_k} \begin{bmatrix} x_k \\ y_k \end{bmatrix} + \boldsymbol{w}_k, \tag{5.2}$$

where $[\hat{u}_k, \hat{v}_k]$ are the image feature in normalized coordinates, and $\boldsymbol{\chi}_k = [x_k, y_k, z_k]$ is the coordinate of the corresponding 3D point in the camera frame at time k.

The likelihood function for a measurement ($\boldsymbol{x} = \boldsymbol{x}_k$) becomes

$$p(\boldsymbol{x}_k|\boldsymbol{\chi}_k) = \exp\left\{-\frac{1}{2}(\boldsymbol{x}_k - \boldsymbol{h}(\boldsymbol{\chi}_k))^T \boldsymbol{Y}_x (\boldsymbol{x}_k - \boldsymbol{h}(\boldsymbol{\chi}_k))\right\}. \tag{5.3}$$

Note that the normalization constant is dropped since it is not a proper probability density with respect to the state variable $\boldsymbol{\chi}_k$ in general.

From (5.2), we can append a 1 at the end of the vector, such that

$$\begin{bmatrix} \hat{u}_k \\ \hat{v}_k \\ 1 \end{bmatrix} = \begin{bmatrix} x_k/z_k \\ y_k/z_k \\ 1 \end{bmatrix} + \begin{bmatrix} \boldsymbol{w}_k \\ 0 \end{bmatrix}, \tag{5.4}$$

Then, by multiplying both sides with z_k and some rearranging, the bearing-only

measurement model can be rewritten in the state space as

$$\begin{bmatrix} x_k \\ y_k \\ z_k \end{bmatrix} = z_k \left(\begin{bmatrix} \hat{u}_k \\ \hat{v}_k \\ 1 \end{bmatrix} - \begin{bmatrix} \boldsymbol{w}_k \\ 0 \end{bmatrix} \right). \tag{5.5}$$

From (5.5), since the depth z_k is not observable from the bearing measurement, it can be seen that the likelihood function is centred around the line along the vector $[\hat{u}_k, \hat{v}_k, 1]^T$. Also, the information matrix of the noise in x–y plane is $\boldsymbol{Y}_x/({z_k}^2)$. This means that the likelihood function is an elliptical cone extending to infinity. Figure 5.1 illustrates the likelihood function which intersects with the camera image plane π, creating an uncertainty ellipse e_1 with the corresponding information matrix $\boldsymbol{Y}_x$. Its scaled-up version is the uncertainty ellipse e_2 at the feature location with corresponding information matrix $\boldsymbol{Y}_x/d^2$. In this work, the uncertainty ellipse e_1 is measured from the dense optical flow, which captures the visual and structural similarity of the local image region. This information matrix is calculated in Section 5.3.2, where

$$\boldsymbol{Y}(e_1) = \boldsymbol{Y}_x = \begin{bmatrix} \tilde{Y}_{xx} & \tilde{Y}_{xy} \\ \tilde{Y}_{xy} & \tilde{Y}_{yy} \end{bmatrix}. \tag{5.6}$$

The cone-shape likelihood can be approximated by Gaussian mixture model or particle samples. In this work, we utilise a degenerate Gaussian which is simple yet effective in approximating the measurement likelihood. This is explained in the following section.

5.2.2 Degenerate Gaussian Representation

A degenerate Gaussian is defined as a Gaussian density where one or more eigenvalues of the covariance matrix are infinite. It is thus not a proper probability distribution due to infinite area or volume. Suppose the feature coordinate $\boldsymbol{x}_k = [0, 0]$, and the corresponding 3D point is at a depth (distance along the z axis) of d. Then, the likelihood function (5.3) expressed in state space, is approximated as an elliptical cylinder function with an infinite eigenvalue in the z-axis with the information matrix

$$\boldsymbol{Y}_0 = \frac{1}{d^2} \left[\begin{array}{cc|c} Y_{xx} & Y_{xy} & 0 \\ Y_{xy} & Y_{yy} & 0 \\ \hline 0 & 0 & 0 \end{array}\right]. \tag{5.7}$$

This is illustrated in Figure 5.2(a).

For a general feature coordinate $[\hat{u}_k, \hat{v}_k]$, the cylinder function needs to be tilted

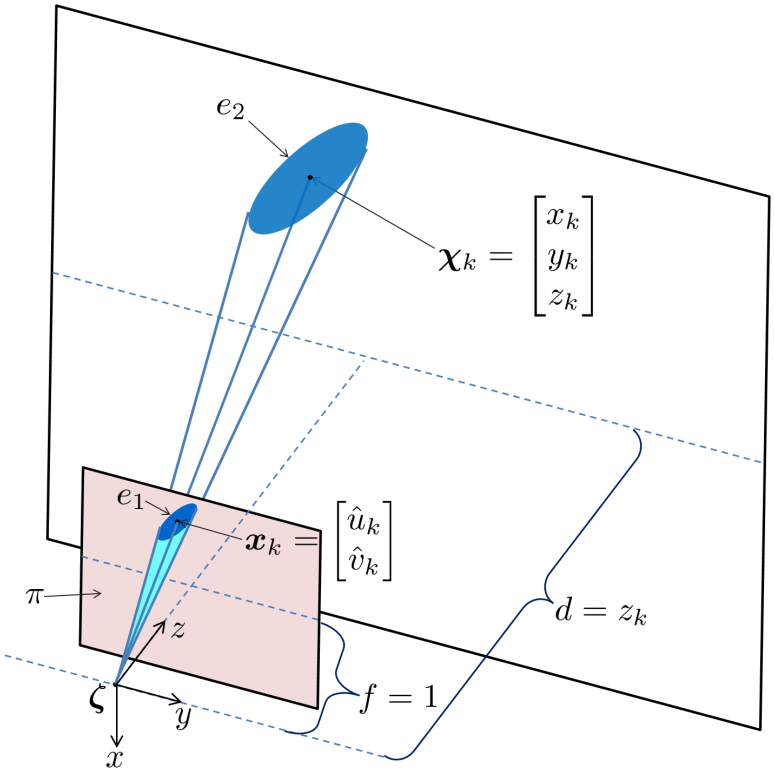

Figure 5.1: Figure shows the underlying probability distribution for one measurement (image feature position $\boldsymbol{x}$) with a level set of the probability distribution illustrated by an uncertainty ellipse e_1, where $\boldsymbol{\zeta}$ is the centre of the camera, f is the focal length (equals to one after normalisation with intrinsic camera parameter), $\boldsymbol{\chi}$ is the location of the 3D point, d is the depth of the 3D point, and e_2 is the scaled up uncertainty ellipse e_1 with respect to depth.

towards the direction of the image feature at rotation angles (α, β) as

$$\boldsymbol{Y} = \frac{1}{d^2} \boldsymbol{R}_{\alpha\beta} \left[\begin{array}{cc|c} Y_{xx} & Y_{xy} & 0 \\ Y_{xy} & Y_{yy} & 0 \\ \hline 0 & 0 & 0 \end{array}\right] \boldsymbol{R}_{\alpha\beta}^T \tag{5.8}$$

with

$$\boldsymbol{R}_{\alpha\beta} = \boldsymbol{R}_y(\beta)\boldsymbol{R}_x(\alpha) \tag{5.9}$$

$$= \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix}, \tag{5.10}$$

where $\alpha = \arctan(-\hat{v}_k / \sqrt{\hat{u}_k^2 + 1^2})$, and $\beta = \arctan(\hat{u}_k / 1)$.

Multiplying the rotation matrices into the information matrix in (5.8), the 3D

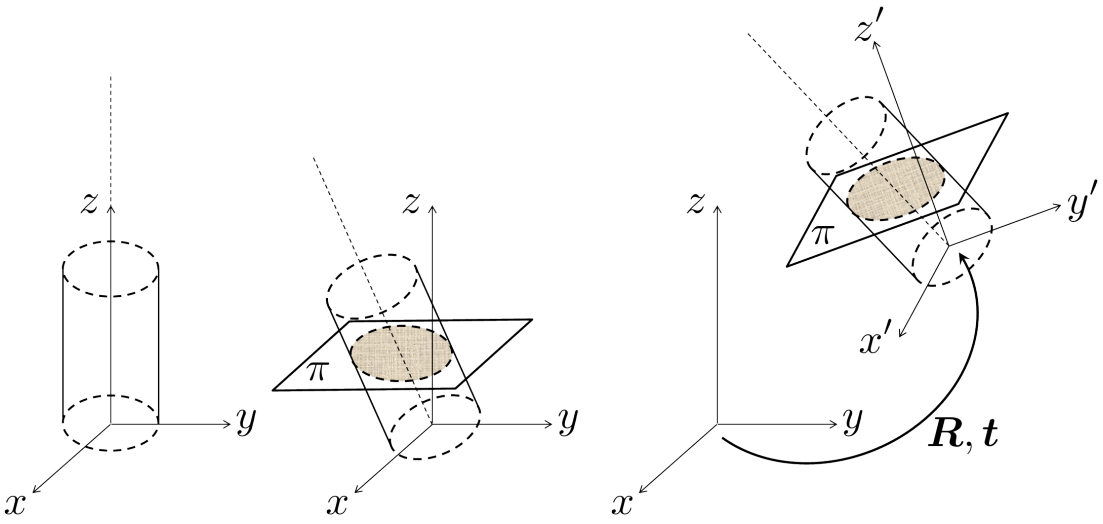

Figure 5.2: 3D degenerate Gaussian likelihood with a degenerate axis, resulting in a cylindrical distribution. From left to right cylinder: (a) degenerate Gaussian (at coordinate system xyz) with an infinite uncertainty along the z-axis; (b) degenerate Gaussian tilted towards the direction of an image feature , where the shaded cross-section is the uncertainty estimated from the optical flow at the image plane π; (c) rigid body transformation of the tilted degenerate Gaussian, where $\boldsymbol{R}, \boldsymbol{t}$ represents the rotation and translation between the coordinate systems.

information matrix given a feature coordinate $[\hat{u}_k, \hat{v}_k]$ is then

$$\boldsymbol{Y} = \frac{1}{d^2} \left[\begin{array}{cc|c} \tilde{Y}_{xx} & \tilde{Y}_{xy} & * \\ \tilde{Y}_{xy} & \tilde{Y}_{yy} & * \\ \hline * & * & * \end{array}\right], \tag{5.11}$$

where the top left block diagonal matrix corresponds to the estimated optical flow uncertainty $\boldsymbol{Y}(e_1)$, such that

$$\begin{bmatrix} \tilde{Y}_{xx} & \tilde{Y}_{xy} \\ \tilde{Y}_{xy} & \tilde{Y}_{yy} \end{bmatrix} = \boldsymbol{Y}(e_1), \tag{5.12}$$

and the $*$ in (5.11) corresponds to unknown values we need to compute.

We can solve the unknowns in (5.11) by first computing the information matrix $\boldsymbol{Y}_0$ in (5.7) as follows. Given the angles α and β (computed from the image measurement), we want the block diagonal matrix in (5.11) to be equals to the measured

optical flow uncertainty $\boldsymbol{Y}_x$ (5.6), then

$$Y_{yy} = \frac{1}{\cos^2\alpha}\tilde{Y}_{yy} \tag{5.13}$$

$$Y_{xy} = \frac{1}{\cos\alpha\cos\beta}\tilde{Y}_{xy} - \sin\alpha\tan\beta\tilde{Y}_{yy} \tag{5.14}$$

$$Y_{xx} = \frac{1}{\cos^2\beta}\tilde{Y}_{xx} - 2\sin\alpha\tan\beta\tilde{Y}_{yy} - \sin^2\alpha\tan^2\beta\tilde{Y}_{xy}. \tag{5.15}$$

The unknowns of $\boldsymbol{Y}$ in (5.11) can then be computed using equation (5.8). This is illustrated in Figure 5.2(b).

Furthermore, the camera may not be aligned to the world coordinate frame. Thus, the tilted cylindrical likelihood undergoes the rigid-body transformation $(\boldsymbol{R}, \boldsymbol{t})$ of the camera frame, giving

$$p(\boldsymbol{x}_k|\boldsymbol{\chi}) = \mathcal{G}(\boldsymbol{t}, \boldsymbol{R}\boldsymbol{Y}^{-1}\boldsymbol{R}^T), \tag{5.16}$$

where the transformation of the cylindrical function is illustrated in Figure 5.2(c).

5.2.3 Re-parametrization

The approximation using the degenerate Gaussian distribution significantly simplifies the fusion of multiple measurements. Similar to MIGE, for each pixel, we propose a parametric form of the 3D distribution using 9 elements vector, which represents the coefficient of the multivariate quadratic equation of the sum of squared Mahalanobis distance equation (negative log likelihood of a Gaussian function).

Let the information matrix and information vector for a 3D measurement be

$$\boldsymbol{Y} \triangleq \boldsymbol{P}^{-1}, \qquad \boldsymbol{y} \triangleq \boldsymbol{Y}\hat{\boldsymbol{\chi}}. \tag{5.17}$$

The quadratic equation of the Gaussian exponential term (dropping the sign and half) expands as

$$\begin{aligned}
&(\boldsymbol{\chi} - \hat{\boldsymbol{\chi}})^T\boldsymbol{P}^{-1}(\boldsymbol{\chi} - \hat{\boldsymbol{\chi}}) \\
&\quad= \begin{bmatrix} x-\hat{x} \\ y-\hat{y} \\ z-\hat{z} \end{bmatrix}^T \begin{bmatrix} Y_{xx} & Y_{xy} & Y_{xz} \\ Y_{xy} & Y_{yy} & Y_{yz} \\ Y_{xz} & Y_{yz} & Y_{zz} \end{bmatrix} \begin{bmatrix} x-\hat{x} \\ y-\hat{y} \\ z-\hat{z} \end{bmatrix} \\
&\quad= (Y_{xx})x^2 + (Y_{yy})y^2 + (Y_{zz})z^2 + (2Y_{xy})xy + (2Y_{xz})xz + (2Y_{yz})yz \\
&\qquad + (-2Y_{xx}\hat{x} - 2Y_{xy}\hat{y} - 2Y_{xz}\hat{z})x + (-2Y_{xy}\hat{x} - 2Y_{yy}\hat{y} - 2Y_{yz}\hat{z})y \\
&\qquad + (-2Y_{xz}\hat{x} - 2Y_{yz}\hat{y} - 2Y_{zz}\hat{z})z \\
&\qquad + const.
\end{aligned} \tag{5.18}$$

Please note that the constant term can be dropped as it only contributes to the scaling of the exponential function. The minimal representation of the 3D Gaussian function is thus obtained using a vector of length 9 to store the coefficients of the

quadratic equation.

The degenerate Gaussian function is then represented using this new parameter ($\mathcal{P}$), such that

$$p(\boldsymbol{x}_k|\boldsymbol{\chi}_k) = \exp\left\{-\frac{1}{2}\mathcal{P}^T\boldsymbol{\xi}\right\}, \tag{5.19}$$

with

$$\begin{aligned}\mathcal{P} &\triangleq [p_1,\ p_2,\ p_3,\ p_4,\ p_5,\ p_6,\ p_7,\ p_8,\ p_9]^T && (5.20)\\ &= [Y_{xx},\ Y_{yy},\ Y_{zz},\ 2Y_{xy},\ 2Y_{xz},\ 2Y_{yz},\\ &\quad -2(Y_{xx}\hat{x} + Y_{xy}\hat{y} + Y_{xz}\hat{z}),\\ &\quad -2(Y_{xy}\hat{x} + Y_{yy}\hat{y} + Y_{yz}\hat{z}),\\ &\quad -2(Y_{xz}\hat{x} + Y_{yz}\hat{y} + Y_{zz}\hat{z})] && (5.21)\\ \boldsymbol{\xi} &\triangleq [\ x^2\quad y^2\quad z^2\quad xy\quad xz\quad yz\quad x\quad y\quad z\ ]^T. && (5.22)\end{aligned}$$

Note that the new parameter $\mathcal{P}$ is actually a concatenation of an information matrix Y and an information state estimate $\hat{y}$ but in a minimal form (equivalent to the square-root information filter), which can be recovered

$$\mathcal{P} \Longleftrightarrow \{\boldsymbol{Y}, \boldsymbol{y}\}, \tag{5.23}$$

with

$$\boldsymbol{Y} = \mathrm{Mat}(\mathcal{P}^{(1:6)}) \triangleq \begin{bmatrix} p_1 & p_4/2 & p_5/2 \\ p_4/2 & p_2 & p_6/2 \\ p_5/2 & p_6/2 & p_3 \end{bmatrix} \tag{5.24}$$

$$\hat{\boldsymbol{y}} \triangleq -\frac{1}{2}\mathcal{P}^{(7:9)}, \tag{5.25}$$

where $\mathrm{Mat}(\cdot)$ defined as a symmetric matricization operator converting the parameters into an information matrix.

Remark: Note that equation (5.18)–(5.25) corresponds to equation (3.23)–(3.29), but is reproduced here due to the use of a slightly different notation to avoid confusion and to make the chapter self-contained.

The data fusion for the triangulation of the 3D scene features are simply element-wise addition using the parametric form as

$$\mathcal{P} = \mathcal{P}_1 + \mathcal{P}_2. \tag{5.26}$$

We perform 5 iterations to compute the suitable scaling (d in equation (5.11)). In the first iteration, we use $d = 1$. Subsequent iterations use the previous estimates of depth and depth's standard deviation for refinement. 5 iterations are performed to ensure convergence (experimentally, 4 iterations are enough for convergence).

These triangulation steps are done for all inlier correspondences to produce an

almost-dense 3D reconstruction of the scene that contains both the position and uncertainty. The reconstructed 3D scene points can also be represented using a depth map (z direction distance from the first image) and depth's standard deviation (square root of the last element of $\boldsymbol{Y}^{-1}$.

5.3 Bayes Dense Flow

A new optical flow method is developed by modifying the existing DCFlow [Xu et al., 2017]. The accuracy of the optical flow estimate is improved by incorporating the epipolar constraint into the cost computation, while the optical flow uncertainty is extracted by fitting a Gaussian function to the cost volume. The resulting dense optical flow is known as Bayes Dense Flow.

5.3.1 Dense Flow with Epipolar Constraint

We propose a robust visual odometry method that uses dense optical flow as input. Optical flow is a method that estimates the motion of each individual pixel between two images. Classical optical flow algorithm optimises a cost function of the form

$$C(\boldsymbol{f}) = C_{data}(\boldsymbol{f}) + \lambda C_{reg}(\boldsymbol{f}), \tag{5.27}$$

where $\boldsymbol{f}$ is the computed optical flow, C_{data} is the data term that penalises visually dissimilar pixel, C_{reg} is the regularisation term that encourages spatially smooth variation of optical flow field, while λ controls the trade-off between the two terms.

We represent the discrete matching cost of a set of candidate pixels in the second image to a corresponding pixel in the first image using a two-dimensional (2D) cost slice. Each pixel in the first image has their respective 2D cost slices, such that the full cost volume is four-dimensional (4D). The full discrete cost volume is used to directly compute dense optical flow [Chen and Koltun, 2016][Xu et al., 2017]. The four-dimensional cost volume has the size of $M \times N \times D \times D$, where $M \times N$ is the scaled down (1/3) dimension of the input images, and D is equals to $2 * d_{max} + 1$, where d_{max} is the maximum pixel displacement between the two images.

The dense optical flow method we use for visual odometry task is a modified version of the direct cost volume optical flow (DCFlow) from Xu et al. [2017]. Currently, their method is one of the most accurate monocular dense optical flow methods for both KITTI and Sintel dataset, with a reasonably short computational time of less than 9 seconds. They have made the code public, and their method directly operates on the full cost volume. This allows us to make the necessary modifications to improve the performance, and also include a function to estimate the optical flow uncertainty. The modifications to the method are illustrated in Figure 5.3.

In most videos (*e.g.* the driving scene from KITTI dataset), large areas of the image frame are covered by low or repetitive textured surfaces such as road and wall of buildings. This makes the task of finding the correct correspondences difficult.

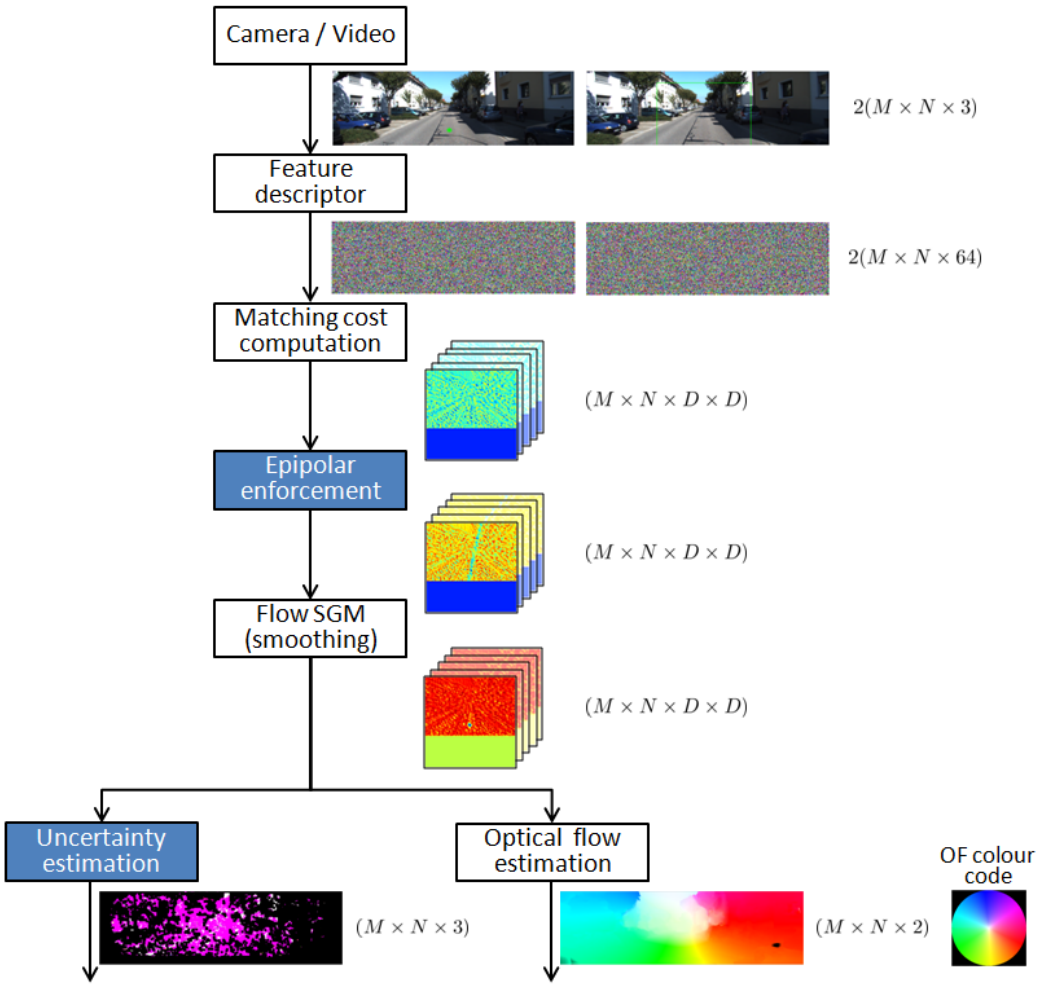

Figure 5.3: The overview of our modified optical flow framework. The blue boxes show our modifications to DCFlow [Xu et al., 2017]. The rescaling of the input image and post-processing part of the algorithm is left out due to space restriction. More details can be found in the text.

One way to reduce the ambiguity of the matching is by applying epipolar constraint into the cost function in (5.27). We encourage the correspondences to be close to the epipolar line by increasing the cost of finding a match far from the line. This is accomplished as follows. First, Shi-Tomasi corner features tracked by Kanade-Lucas-Tomasi (KLT) algorithm [Shi and Tomasi, 1994], are used as sparse correspondences for the well known eight-point algorithm [Hartley, 1997] to obtain an initial estimate of the Fundamental matrix. A truncated L2 cost is added to the cost volume to enforce the epipolar constraint based on the computed Fundamental matrix.

When the pixel in the first image corresponds to a static point of the scene, the cost of finding the match far away from the epipolar line is increased proportionately to squared distance. Conversely, when a pixel in the first image corresponds to a point on a moving object, a truncated cost is applied. This helps to avoid matches

that satisfy the epipolar constraint but are visually dissimilar to be wrongly selected.

The epipolar constraint is added to the cost function before regularisation is applied. Figure 5.4 shows an example of the epipolar constraint being added to one of the cost volume slices.

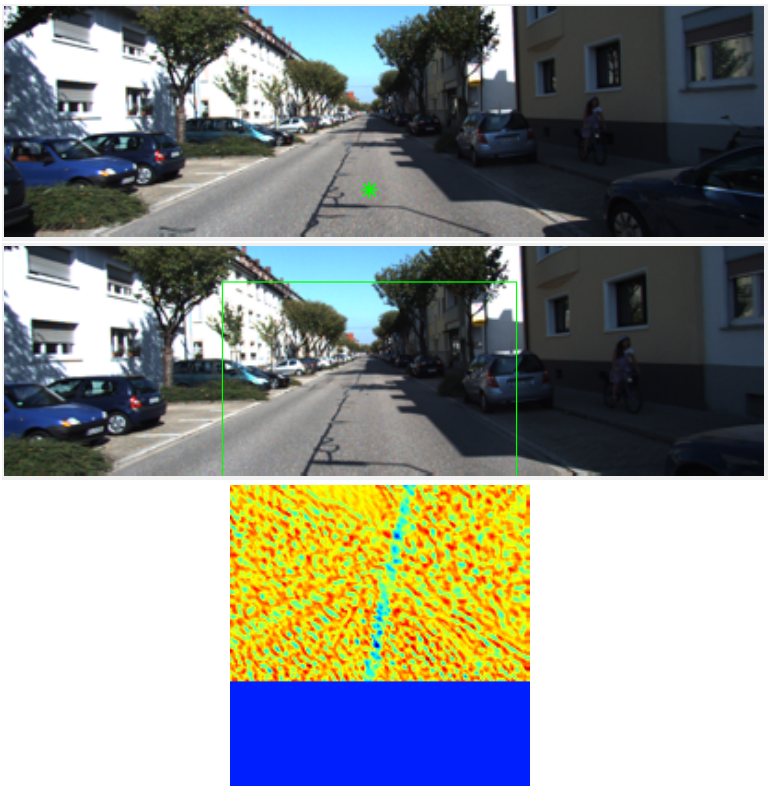

Figure 5.4: Example illustrating epipolar constraint added to a cost slice (before spatial smoothness regularisation step). From top to bottom: (a) first image with a pixel marked by a green star; (b) second image with a bounding box enclosing the candidate matching pixels for the pixel marked in the first image; (c) cost slices representing the matching cost of corresponding candidate matching pixels with addition of truncated epipolar cost. Note that the candidate matching pixels outside the boundary of the image is assigned a fixed cost (blue colour at the bottom of the cost slices).

5.3.2 Uncertainty Estimation

Like most state-of-the-art optical flow methods, DCFlow [Xu et al., 2017] implicitly assumes each correspondence has a homogeneous, isotropic Gaussian uncertainty. However, the uncertainty of each correspondence can have different magnitude and correlation, thus heteroscedastic, depending on the visual similarity of neighbouring pixels. Figure 5.5 illustrates an example of a matching cost slice, in which the negative logarithm of a unimodal Gaussian distribution is fitted to the optic flow cost output.

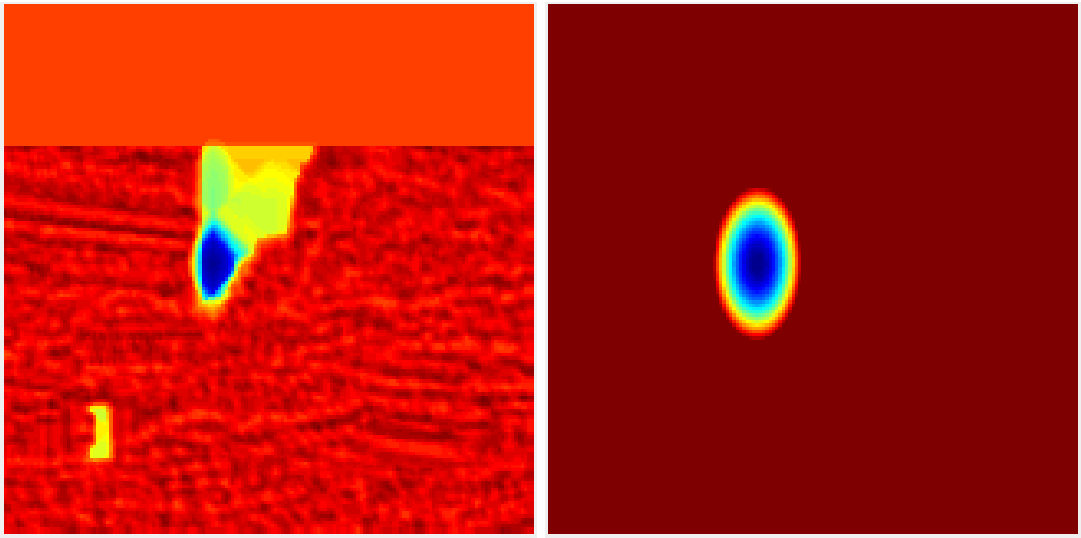

Figure 5.5: An example showing the uncertainty fitting of negative logarithm of a bivariate Gaussian to a matching cost slice (after spatial smoothness regularisation step). From left to right: (a) 2D cost slice, (b) the approximate 2D cost slice using 2D Gaussian fitting.

For a general two-dimensional Gaussian distribution, we know that the negative logarithm of the likelihood function is half of the squared Mahalanobis distance. The squared Mahalanobis distance, d_M^2 can be computed as [Mahalanobis, 1936]

$$d_M^2(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{P}) = (\boldsymbol{x} - \boldsymbol{\mu})^T \boldsymbol{P}^{-1} (\boldsymbol{x} - \boldsymbol{\mu}) \tag{5.28}$$

$$d_M{}^2 = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} \tilde{Y}_{xx} & \tilde{Y}_{xy} \\ \tilde{Y}_{xy} & \tilde{Y}_{yy} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{5.29}$$

$$= \tilde{Y}_{xx} x^2 + 2\tilde{Y}_{xy} xy + \tilde{Y}_{yy} y^2, \tag{5.30}$$

where $\boldsymbol{x}$ is the vector representing the coordinates of a point, $\boldsymbol{\mu}$ is the vector representing the coordinates of the mean (optical flow) of the Gaussian distribution, and $\boldsymbol{P}$ is the covariance matrix of the Gaussian distribution.

The elements of information matrix, $\boldsymbol{Y}$ can then be computed using linear least square equation as

$$\underbrace{\begin{bmatrix} x_1{}^2 & 2x_1y_1 & y_1{}^2 \\ x_2{}^2 & 2x_2y_2 & y_2{}^2 \\ & \vdots & \\ x_N{}^2 & 2x_Ny_N & y_N{}^2 \end{bmatrix}}_{\boldsymbol{A}} \underbrace{\begin{bmatrix} \tilde{Y}_{xx} \\ \tilde{Y}_{xy} \\ \tilde{Y}_{yy} \end{bmatrix}}_{\boldsymbol{Y}} = \underbrace{\begin{bmatrix} d_1{}^2 \\ d_2{}^2 \\ \vdots \\ d_N{}^2 \end{bmatrix}}_{\boldsymbol{d}} \tag{5.31}$$

$$\therefore \boldsymbol{Y} = (\boldsymbol{A}^T \boldsymbol{A})^{-1} \boldsymbol{A}^T \boldsymbol{d}.$$

DCFlow computes the matching cost efficiently by using $(1 - f_1 \cdot f_2)$, where f_1

and f_2 are the unit vectors representing the image feature descriptor. This results in a matching cost value between 0 (visually similar) and 1 (visually dissimilar). However, the negative logarithm of a Gaussian likelihood function has value between 0 to infinity. Thus, we can exclude pixels with high cost from our Gaussian fitting by only using pixels with a matching cost below a set threshold.

Similar to most top performance optical flow method, DCFlow has a post-processing step to remove unreliable matches from the semi-dense correspondences before EpicFlow [Revaud et al., 2015] interpolation. This is accomplished by computing the forward and backward optical flow, and removing those matches that do not satisfy the forward-backward consistency. This post-processing step changes the uncertainty estimate such that the correspondences that got removed should be assigned a high uncertainty. We replace those values with the maximum uncertainty of the optical flow estimate.

These provide us with a three channels $(\tilde{Y}_{xx}, \tilde{Y}_{xy}, \tilde{Y}_{yy})$, floating-point image encoding the information matrix for every pixel correspondences for the scaled-down pair of RGB input images. We can scale the uncertainty image back to the original resolution by applying an image resize operation. First, the information matrix parameters are converted to covariance parameters, which is scaled up to the original image resolution, followed by a multiplication of 9 (squared of image rescaling factor). The scaled-up covariance parameters are then converted back to information matrix following matrix inverse.

The estimated uncertainty can also be used to determine if the two input images are visually similar, which will be helpful when computing the loop closure constraints (section 5.5). If the two input images belong to the same scene, most of their local neighbours will have similar optical flow magnitude and direction. Regularisation step will then shrink the region of possible matching locations, and thus, the uncertainty decreases. On the other hand, if the two input images belong to different scenes, local neighbours may have very different optical flow magnitude and direction. Regularisation step will not be able to shrink the region of possible matching location, and the uncertainty is high. This is illustrated in Figure 5.6.

5.4 Robust Visual Odometry (SLAM Front-end)

The dense optical flow, dense depth estimate (prior) and their uncertainty are used to estimate the inter-frame motion. The dense optical flow correspondences are treated like conventional sparse feature matches, while the optical flow uncertainty is used during the sampling step of RANSAC, and also to apply a weighting to their corresponding equation in our Mahalanobis eight-point algorithm.

The method to accurately recover the scale of the motion is also presented. The accurate inter-frame motion estimate is obtained from fusing the Mahalanobis eight-point algorithm result and perspective-n-points result.

A new method to efficiently fuse two given depth map (triangulated scene points) is also presented, where the previous depth map estimate is fused with the current

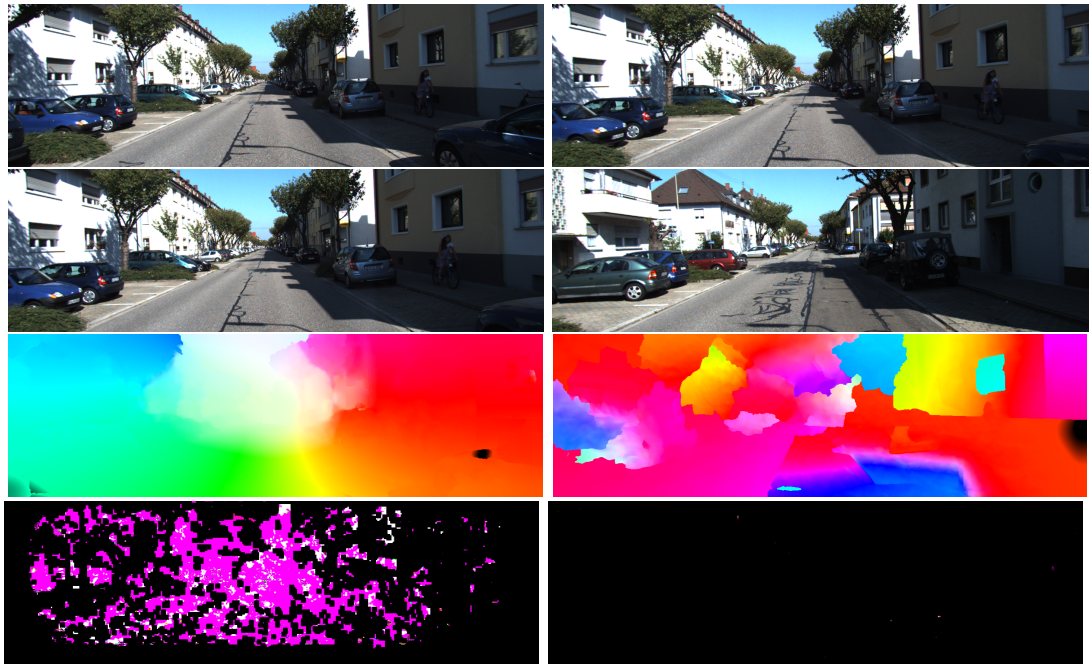

Figure 5.6: Example of estimated optical flow and uncertainty magnitude. From top to bottom: first input image, second input image, optical flow, estimated information matrix. Left column corresponds to sequential images, while right column corresponds to two input images with high structural similarity (SSIM) index, but is not of the same scene. Black colour for information matrix values corresponds to high covariance (unreliable) pixels.

estimate to obtain more accurate depth map result. The depth map is then propagated to the next frame for future computation.

Ways to determine and handle small motion in the video sequences are also discussed in the following subsections.

Remark 5.1. *Note that unless otherwise stated, the estimated poses, reconstructed 3D points and 3D points uncertainty are all expressed with respect to the previous frame. For example, at current time, a new image frame with index t is captured, we fix the coordinate frame at the pose of frame $t-1$, with z-axis pointing forward, x-axis points to the right, y-axis points downward, and the origin at the centre of the camera at frame $t-1$.*

5.4.1 Mahalanobis 8-points Algorithm

From a pair of input images, we can find a set of matching pixels $\boldsymbol{x}_i \leftrightarrow \boldsymbol{x}_i'$. Then, the fundamental matrix $\boldsymbol{F}$ satisfies

$$\boldsymbol{x}_i'^T \boldsymbol{F} \boldsymbol{x}_i = 0. \tag{5.32}$$

Each matching pixel provides a linear constraint on the elements of $\boldsymbol{F}$. Since the scale of $\boldsymbol{F}$ can be arbitrary, the solution of $\boldsymbol{F}$ can be computed using 8 sets of matching pixels. A vector of length 9 is used to represent all the elements of the

Fundamental matrix. Given n pairs of matching image features, the linear constraints can be concatenated into a matrix form as

$$\boldsymbol{A}\boldsymbol{f} = \begin{bmatrix} x'_1x_1 & x'_1y_1 & x'_1 & y'_1x_1 & y'_1y_1 & y'_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x'_nx_n & x'_ny_n & x'_n & y'_nx_n & y'_ny_n & y'_n & x_n & y_n & 1 \end{bmatrix} \boldsymbol{f} = \boldsymbol{0}. \tag{5.33}$$

The solution of $\boldsymbol{f}$ is then computed as the null space of matrix $\boldsymbol{A}$. When more than 8 noisy matching pixels are provided as input, RANSAC is applied to identify reliable matches (inliers) to compute $\boldsymbol{F}$. Given the inlier set, the solution of $\boldsymbol{f}$ is then refined by computing the corresponding right singular vector of $\boldsymbol{A}$ with the smallest singular value. This is the well-known eight-point algorithm, where sparse feature matches are typically used.

However, solving the null space of equation (5.33) only minimises the algebraic error $||\boldsymbol{x}'^T\boldsymbol{F}\boldsymbol{x}||$, which does not guarantees the minimisation of a meaningful geometrical distance. One well-known method minimises the Sampson distance [Torr and Zisserman, 1998; Zhang, 1998], which modifies the the rows of matrix $\boldsymbol{A}$ by a multiplicative scaling, such that

$$\boldsymbol{A}\boldsymbol{f} = \begin{bmatrix} \phi x'_1x_1 & \phi x'_1y_1 & \phi x'_1 & \phi y'_1x_1 & \phi y'_1y_1 & \phi y'_1 & \phi x_1 & \phi y_1 & \phi_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \phi_n x'_nx_n & \phi_n x'_ny_n & \phi_n x'_n & \phi_n y'_nx_n & \phi_n y'_ny_n & \phi_n y'_n & \phi_n x_n & \phi_n y_n & \phi_n \end{bmatrix} \boldsymbol{f} = \boldsymbol{0}, \tag{5.34}$$

where

$$\phi_i = \frac{1}{\sqrt{(\tilde{\boldsymbol{F}}\boldsymbol{x}_i)_1^2 + (\tilde{\boldsymbol{F}}\boldsymbol{x}_i)_2^2 + (\tilde{\boldsymbol{F}}^T\boldsymbol{x}'_i)_1^2 + (\tilde{\boldsymbol{F}}^T\boldsymbol{x}'_i)_2^2}}, \tag{5.35}$$

and $\tilde{\boldsymbol{F}}$ is the iteratively refined Fundamental matrix that is first initialised by computing the null space of $\boldsymbol{A}$ from (5.33). The rank 2 constraint is also enforced on the solution to obtain the final estimate of $\boldsymbol{F}$.

Instead of enforcing the rank 2 constraint at the end similar to the eight point algorithm, there are also nonlinear methods that enforce additional constraints from the start. These methods require less matching points to estimate $\boldsymbol{F}$. For example, five-point algorithm [Nister, 2004][Li and Hartley, 2006], six-point algorithm [Schaffalitzky et al., 2000], and seven-point algorithm [Hartley and Zisserman, 2003] has been proposed. However, we will focus on eight-point algorithm as the computation is the most straight forward.

Also, unlike most existing work (*e.g.* [Raguram et al., 2009]), where the error of the matching pixels' location is assumed to be isotropic Gaussian with equal variance, we propose a new algorithm to estimate inter-frame motion that uses dense optical flow with non-isotropic pixel error (uncertainty). Dense optical flow correspondences are treated similar to sparse feature matches in conventional eight-point algorithm. In the modified RANSAC step, the uncertainty (square root of trace of the covariance matrix) of the optical flow is used to guide the sampling of the matches by increasing

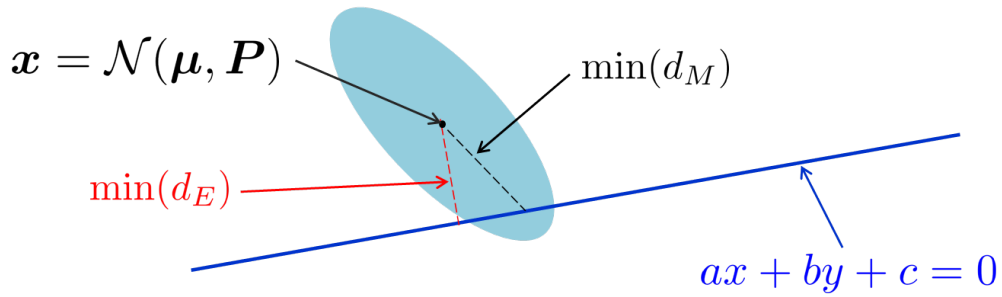

Figure 5.7: Illustrative figure showing an image feature pixel $\boldsymbol{x}$ represented as a 2-dimensional random variable with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{P}$, the epipolar line is represented as a straight line $\boldsymbol{l}$ with equation $ax + by + c = 0$, $\min(d_M)$ is the minimum Mahalanobis distance, while $\min(d_E)$ is the minimum Euclidean distance.

the likelihood of selecting correspondences (or matches) with a lower uncertainty. This is accomplished using multinomial resampling method [Douc and Cappe, 2005] commonly used in particle filter, which helps in decreasing the required number of iterations to ensure good inlier set selection.

We determine the inlier set by using both the Euclidean distance and Mahalanobis distance. The inlier must be within a threshold distance (both Euclidean and Mahalanobis) from the epipolar line. We choose the inlier set that minimises the sum of truncated distance from the epipolar line.

The optical flow uncertainty is also used to apply a weighting to each equation (row of matrix $\boldsymbol{A}$) during the refinement step of the Mahalanobis eight-point algorithm. This ensures that the solution of the Fundamental matrix minimises the squared Mahalanobis distance to all the inlier correspondences with respect to their individual uncertainty. This is illustrated in Figure 5.7.

Similar to Sampson distance, this is accomplished by applying a multiplicative scaling to each row, such that the weights in (5.34) are

$$\phi_i = \sqrt{\frac{\tilde{Y}_{xx}\tilde{Y}_{yy} - \tilde{Y}_{xy}^2}{(\tilde{\boldsymbol{F}}\boldsymbol{x_i})_1{}^2\tilde{Y}_{yy} + (\tilde{\boldsymbol{F}}\boldsymbol{x_i})_2{}^2\tilde{Y}_{xx} - 2(\tilde{\boldsymbol{F}}\boldsymbol{x_i})_1(\tilde{\boldsymbol{F}}\boldsymbol{x_i})_2\tilde{Y}_{xy}}}, \tag{5.36}$$

where the notation $(\boldsymbol{v})_k$ is the k^{th} element of the vector $\boldsymbol{v}$. (see Appendix 5.9 for proof)

Similar to the Sampson distance method, $\tilde{\boldsymbol{F}}$ is the iteratively refined Fundamental matrix that is first initialised by computing the null space of $\boldsymbol{A}$ from (5.33). The refinement step is performed for 5 iterations to ensure convergence. The method is named as Mahalanobis eight-point algorithm, and it is detailed in Algorithm 5.

From the estimated Fundamental matrix $\boldsymbol{F}$ and intrinsic camera matrix $\boldsymbol{K}$, the essential matrix $\boldsymbol{E}$ is recovered as

$$\boldsymbol{E} = \boldsymbol{K'}^T\boldsymbol{F}\boldsymbol{K}. \tag{5.37}$$

Algorithm 5: Mahalanobis eight-point algorithm for inter-frame motion estimation

Data: Dense correspondences $\{\boldsymbol{m_1}, \boldsymbol{m_2}\}$ from optical flow, uncertainty of correspondences $\{\boldsymbol{Y}\}$, intrinsic camera parameters $\boldsymbol{K}$
Result: Inter-frame motion, $\boldsymbol{R}$ and $\boldsymbol{t}$ (translation has unknown scale)

initialization ;
Compute normalised correspondences $\{\hat{\boldsymbol{m}}_1, \hat{\boldsymbol{m}}_2\}$;
for r = *1:N* **do**
    Find random 8 correspondences (with higher chance of selecting correspondences with lower uncertainty);
    Estimate Fundamental matrix $\boldsymbol{F}$ (normalised eight-point algorithm);
    Compute Euclidean distance d_E and Mahalanobis distance d_M from epipolar lines;
    Find outliers, where $d_E > \tau$ or $d_M > \tau$;
    Find number of inliers n;
    Compute truncated Mahalanobis distance d_T, where outlier has a fixed distance of τ;
    if $sum(d_T) < d_{prev.}$ **then**
        Assign $d_{prev.} = sum(d_T)$;
        Assign $\boldsymbol{F_{est}} = \boldsymbol{F}$;
        Assign inlier set to the new inlier set;
    end
    if $r > (log(1 - 0.9999)/log(1 - (n/n_{total})^8))$ **and** $r > 10$ **then**
        break;
    end
end
for r = *1:5* **do**
    Refine $\boldsymbol{F_{est}}$ by weighing all inliers equations with their corresponding weighing factor (ϕ_i from equation (5.36);
end
Denormalise $\boldsymbol{F_{est}}$ and fit to the closest rank 2 matrix;
Estimate Essential matrix $\boldsymbol{E}$;
Make $\boldsymbol{E}$ has singular values of [1,1,0];
Extract two possible rotations and translations from $\boldsymbol{E}$;
Use chirality constraint to find the correct $\boldsymbol{R}$ and $\boldsymbol{t}$;

The camera pose is represented using extrinsic camera matrix

$$\boldsymbol{T}_{ex} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0}^T & 1 \end{bmatrix}, \tag{5.38}$$

and it is related to the essential matrix by

$$\boldsymbol{E} = [\boldsymbol{t}]_\times \boldsymbol{R}, \tag{5.39}$$

where the extrinsic camera matrix (with translational scale ambiguity) is recovered following Theorem 2.12.

5.4.2 Scale Estimation

Unlike stereo visual odometry, the translational scale cannot easily be recovered due to the lack of a reliable reference (*i.e.* fixed, known stereo baseline). Thus, we can only estimate the scale by relying on other assumptions. The most common assumption used by monocular visual odometry is that the camera moves at a fixed height from a roughly planar ground. This is a good assumption for video sequence captured from any ground-based vehicle or robot. However, this does not apply to videos captured from an aerial vehicle (*e.g.* unmanned aerial vehicle (UAV)). The scale must then be determined using other methods. Two methods to recover the scale are proposed as follows.

Firstly, the scale can be determined by fitting a plane through the 3D reconstructed points that are roughly parallel to the zx (forward-right) plane of the camera axis. Assuming the ground is visible roughly in the middle of the image, we use the reconstructed points below the camera (y coordinate of the 3D points is positive), and not too far to the side (image coordinate x within half the image width from the centre) of the camera. Plane fitting provides us with a plane equation satisfying $ax + by + cz + d = 0$. The height of the plane with respect to 1 unit of inter-frame translation is then equals to $-d/b$. If the height of the camera, h is known (calibrated from training data, or estimated throughout the motion), the scale of the inter-frame translation s can be computed as $s = -(bh)/d$.

Secondly, the translational scale can also be recovered by computing the multiplicative factor that relates the current and previously computed depth map. The median of the multiplicative factor between corresponding depth values provides a robust translational scale estimate.

For ground-based vehicle/robot, we can combine the scale estimated from the ground height and depth map using a simple average. For aerial vehicle (UAV), we cannot ensure that the height from the ground is fixed, and thus, we estimate the scale using ground height for the first frame, and relies on the scale from the depth map for subsequent frames.

The height of the camera is also constantly being updated using the reconstructed 3D points of the scene, which is only used to reinitialise the translational scale when not enough ($< 5\%$) 3D points from the previous estimate overlaps the current trian-

gulated points.

The estimated scale is then multiplied to the estimated camera translation, 3D reconstructed points, depth map and uncertainty. The information matrix of the reconstructed 3D points are divided by the squared scale.

5.4.3 Inter-frame Pose Fusion

Given the dense optical flow, there are two methods to estimate the inter-frame pose/motion. One method is by using the Mahalanobis eight-point algorithm (Section 5.4.1) along with the estimated scale (Section 5.4.2). Another method is by using the perspective-n-point (PnP) method [Gao et al., 2003]. PnP uses the 2D motion of the pixels (from optical flow), 3D location of the corresponding points (estimated from 3D reconstruction) and intrinsic camera parameter to estimate the motion of the camera.

We can improve the performance of PnP method by discarding unreliable correspondences as input. We propose to use only pixels that have the following properties:

- standard deviation of the depth is less than 0.3 times the estimated depth
- square root of the trace of optical flow covariance matrix is less than $\sqrt{2}$

We then propose to fuse the two estimated poses by doing a simple average, where rotation $\boldsymbol{R}_{ave}$ and translation $\boldsymbol{t}_{ave}$ are handled separately as

$$\boldsymbol{R}_{ave} = \boldsymbol{R}_1 \exp\left(\frac{\log(\boldsymbol{R}_1{}^T \boldsymbol{R}_2)}{2}\right), \quad \boldsymbol{t}_{ave} = \frac{\boldsymbol{t}_1 + \boldsymbol{t}_2}{2}, \tag{5.40}$$

where exp is the matrix exponential function, while log is the matrix logarithm function [Moakher, 2002].

When a lot of the depth values have not converged to an accurate value, the PnP estimate may return an error prone result. Thus, we only perform the fusion when the difference in the estimated translation scale is within 30% of the scale estimated in Section 5.4.2, and the estimated rotations has a difference less than 0.5 radians. If either of these conditions are not met, we use the pose estimated from the Mahalanobis eight-point algorithm (Section 5.4.1) instead.

The 3D scene points are re-estimated as discussed in Section 5.2 using the fused camera pose estimate.

5.4.4 3D Scene Points Fusion and Propagation

Two independent estimates of the 3D scene reconstruction can be obtained for the previous frame $t-1$, where one is estimated from frame $t-2$ and $t-1$, while the second is estimated from frame $t-1$ and t. We propose a simple method to fuse the two 3D scene estimate as follows.

In the 3D reconstruction step, we do not use outlier correspondences (identified during Mahalanobis eight-point algorithm) because they may be points on moving objects (*e.g.* cars) or error prone correspondences (*e.g.* occluded or out-of-view pixels). This results in a reconstruction with some missing information.

We can perform data fusion for pixels that are triangulated for both pairs of input frames as follows. Given the means ($\hat{\chi}_1$ and $\hat{\chi}_2$) and their corresponding information matrix ($\overline{Y}_1$ and $\overline{Y}_2$), let

$$\hat{\chi}_1 = \begin{bmatrix} \overline{x}_1 \\ \overline{y}_1 \\ \overline{z}_1 \end{bmatrix}, \quad \hat{\chi}_2 = \begin{bmatrix} \overline{x}_2 \\ \overline{y}_2 \\ \overline{z}_2 \end{bmatrix} \tag{5.41}$$

$$\overline{Y}_1 = \begin{bmatrix} Y_{1,xx} & Y_{1,xy} & Y_{1,xz} \\ Y_{1,xy} & Y_{1,yy} & Y_{1,yz} \\ Y_{1,xz} & Y_{1,yz} & Y_{1,zz} \end{bmatrix}, \quad \overline{Y}_2 = \begin{bmatrix} Y_{2,xx} & Y_{2,xy} & Y_{2,xz} \\ Y_{2,xy} & Y_{2,yy} & Y_{2,yz} \\ Y_{2,xz} & Y_{2,yz} & Y_{2,zz} \end{bmatrix}. \tag{5.42}$$

Similar to equation (5.18), we can transform these information using the parametric form, where the fusion simplifies to element-wise addition as $\mathcal{P}^T = {\mathcal{P}_1}^T + {\mathcal{P}_2}^T$.

The pixels that are triangulated for only one of the input pairs of image are assigned the mean and information matrix of the respective triangulation result.

The fused depth map and reconstructed scene points for the previous frame ($t-1$) can also be propagated to the current image frame (t). This provides a prior 3D scene information for the next image frame. Given the computed camera extrinsic matrix $\boldsymbol{T}_{ex}$, the homogeneous 3D points in the previous frame $\hat{\chi}_{t-1}$ are propagated to the current image $\hat{\chi}_t$ as

$$\hat{\chi}_t = \boldsymbol{T}_{ex}\hat{\chi}_{t-1}. \tag{5.43}$$

The corresponding pixel locations of the propagated 3D scene points $\hat{\chi}_t$ are then projected into the image coordinate using the intrinsic camera parameter matrix $\boldsymbol{K}$, such that

$$\boldsymbol{q}_t = round(\boldsymbol{K}\hat{\chi}_t / \hat{\chi}_t[3]), \tag{5.44}$$

where *round* is the rounding to the nearest integer function, and $\hat{\chi}_t[3]$ is the third element (z-coordinate) of the 3D scene point.

We place an upper bound on the memory requirement of our algorithm by only storing 3D scene points in the visible region of the scene, where points that got mapped outside of the image boundary are discarded. Multiple 3D points that got mapped to the same pixel location are also discarded. These are points that are either occluded or are outside the field of view of the camera, which are less reliable to track.

The information matrices are also propagated to the current frame by multiplication of the extrinsic camera rotation as

$$\overline{Y}_t = \boldsymbol{R}_{ex}\overline{Y}_{t-1}{\boldsymbol{R}_{ex}}^T. \tag{5.45}$$

5.4.5 Small Motion Handling

The use of scaled-down images (one-third the original scale) for dense optical flow estimation cannot guarantee the accuracy of the matches when the pixel translation between two images is too small. This occurs when the vehicle moves very slowly or stops completely, causing the motion estimation to be error-prone. Small translational motion estimation is a common problem in most monocular visual odometry. This is because small parallax between two images leads to difficulty in estimating both motion and structure accurately.

We determine if the inter-frame motion is big enough using two separate conditions. First, the Shi-Tomasi corner matches have a median displacement magnitude of at least 2.5 pixels. Secondly, the third quantile (75%) of the computed optical flow has a magnitude greater than 5. If either of the two conditions is not met, the inter-frame motion is computed using perspective-n-points (PnP) method by using the previously computed depth and the motion of the corresponding pixels (optical flow).

5.4.6 Global Camera Pose Estimate

We can compute the global camera pose at the current frame $\boldsymbol{T}_t$ using the inter-frame camera pose expressed as camera extrinsic parameters (R_{ex} and t_{ex}) as follows. Let

$$\boldsymbol{T}_{ex} = \begin{bmatrix} \boldsymbol{R}_{ex} & \boldsymbol{t}_{ex} \\ \mathbf{0} & 1 \end{bmatrix}, \tag{5.46}$$

where $\mathbf{0} = [0, 0, 0]$.

Then,

$$\boldsymbol{T}_t = \boldsymbol{T}_{t-1}{\boldsymbol{T}_{ex}}^{-1}, \tag{5.47}$$

where $\boldsymbol{T}_{t-1}$ is the camera pose of the previous frame $(t-1)$.

5.5 Robust Loop Closure (SLAM Back-end)

Loop closure is possible when a previously visited location is revisited. We determine the candidate frames for loop closure in three steps. The first step is by selecting frames with their estimated poses to be less than a fixed (metric) distance away, while having a difference in frame index no less than a threshold value. The minimum frame index difference is enforced to prevent finding too many candidates within neighbouring frames. We can further reduce the possible candidates by only finding candidate loop closure images for every 10 frames.

The second step is to determine which of the candidate loop closure images are valid, by using the structural similarity index (SSIM) [Wang et al., 2004]. We discard any images that have a SSIM index less than a set threshold (experimentally set to 0.38), and keep a maximum of three candidate images with the highest SSIM index. Lastly, the dense optical flow between the images and their possible neighbours are

computed. The estimated uncertainty is used to determine if the optical flow is reliable, and only compute their inter-frame motion when the percentage of matches with an uncertainty less than a set threshold is greater than 20% (an example is shown in Figure 5.6). During the loop closure, the inter-frame motion estimation step also checks for the small motion conditions as discussed in previous sections. This provides us with close loop constraints that links temporally far away poses that are spatially close to each other.

Loop closure is accomplished by using the robust linear pose graph-based optimisation method from Cheng et al. [2015]. Similar to other pose graph SLAM, their method treats all poses of the vehicle or robot as vertices, and inter-pose constraints (*e.g.* odometry and close loop constraints) as edges. The linear pose graph-based optimisation minimises the cost function,

$$\min_{\{x_i\}} \sum_i \sum_j c_{ij} ||(\boldsymbol{z}_{ij} - h(\boldsymbol{x}_i, \boldsymbol{x}_j))||_{\mathcal{I}}, \tag{5.48}$$

where c_{ij} is a scalar weight, $\boldsymbol{z}_{ij}$ is the loop closure pose constraint between time i and j, $h(\cdot, \cdot)$ is the nonlinear function that computes the relative pose between the two input, $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are the estimated poses at time i and j respectively.

Their method is used due to the method's robustness of handling outliers in the loop closure constraints, and effectively discarding wrong edges, preventing incorrect convergence result.

5.6 Algorithm overview

Flow chart in Figure 5.8 shows our proposed SLAM framework. Given a pair of consecutive image frames taken from a monocular camera, dense optical flow and their corresponding uncertainty are computed. If the pixels motion is small, we directly compute the inter-frame pose using perspective-n-point (PnP) method. Otherwise, the inter-frame pose and 3D scene points are estimated using our proposed Mahalanobis eight-points and Bayesian triangulation algorithm.

The translational and 3D scene points scale are then estimated using the triangulated 3D scene points, using the camera height from the ground or by matching with previously computed depth map. For aerial video, we update the camera height using the propagated depth map. For ground-based vehicle, the camera height is assumed constant, where the value is calibrated from training data ($1.7m$ for KITTI dataset).

We then fuse the inter-frame pose estimate from our Mahalanobis eight-point algorithm and PnP method, which is integrated to obtain the global camera pose. The estimated 3D scene points are also fused with previous estimate, which is then propagated to be used as a prior for the next frame.

The computed camera trajectory is then used as input to the back-end of the SLAM framework, where loop closure constraints are enforced to reduce estimation drift.

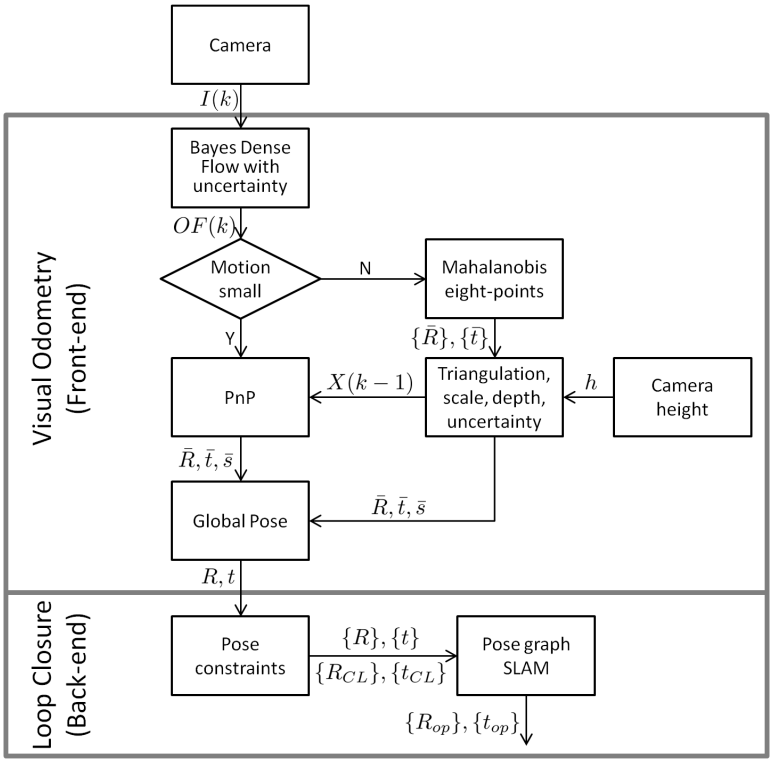

Figure 5.8: Our proposed SLAM framework. Notation k is the frame number, OF is the computed dense optical flow, $\bar{R}$ and $\bar{t}$ are the inter-frame pose, $\bar{s}$ is the estimated translational scale, X is the triangulated 3D scene points, R and t are the camera pose in global coordinate frame, subscript CL represents loop closure constraints, while subscript op represent pose-graph SLAM optimised result. The height for ground-based vehicle is assumed constant, while aerial vehicle require frequent re-estimation of the camera height.

5.7 Experimental Results

We evaluated our proposed SLAM framework using the well-known KITTI dataset [Geiger et al., 2012] and our own UAV dataset. KITTI dataset shows a camera mounted on a vehicle travelling on a roughly planar ground. The sequence 01 in particular is a challenging highway scenario, where the vehicle is travelling at high speed and there are few distinctive feature points within view. UAV dataset shows a camera mounted on a quadcopter flying in a highly unstructured outdoor environment with dynamically moving objects. The UAV also performs motions such as (almost) pure rotation and drastic height variation. These make accurate estimation of camera pose difficult for existing monocular visual odometry and SLAM.

5.7.1 Ground-based Vehicle

The dataset we used to verify the performance of our proposed algorithm for ground-based vehicle/robot is taken from KITTI benchmark. For optical flow evaluation, we use the flow 2015 dataset [Menze and Geiger, 2015], while for the odometry, we use the odometry dataset [Geiger et al., 2012]. For both experiments, we use the monocular RGB images (*image_2* folder). In the odometry experiment, we assume the camera is $1.7m$ above the ground, with zero pitch.

Due to the post-processing part of the DCFlow code not made available, we can only verify the optical flow result before homography fitting is applied to the EpicFlow [Revaud et al., 2015] interpolated results. Based on KITTI 2015 optical flow dataset, by applying our epipolar constraint on the cost volume, we achieved a 0.6% improvement in accuracy (in terms of less than 3 pixels endpoint error criterion). The improvement is small due to the epipolar truncation cost being set very low to accommodate for dynamic pixels in the scene. However, we can visually observe a noticeable improvement in the optical flow estimation for the ground pixels, not reflected by the large (3 pixels error) KITTI accuracy metric. We also implemented a homography fitting step based on the description of their paper.

The uncertainty estimate for the dense optical flow is visually inspected, where it was observed that occluded, out-of-bound or textureless regions of the image have high uncertainty value.

For ground-based vehicle's visual odometry result, we compare our performance with existing methods, specifically VISO2-M [Geiger et al., 2011], MLM-SFM [Song et al., 2016], PMO [Fanani et al., 2017] and DOF-1DU+LC [Ng et al., 2017]. We selected a few of the available sequences that contain slow moving vehicle in an urban environment (sequence 00), fast moving vehicle on a highway (sequence 01) and vehicle travelling in a loop (sequence 06) to gauge the performance of our proposed methods. The results are summarised in Table 5.1 and Table 5.2.

seq	DOF-2DU		DOF-2DU+PnP		DOF-2DU+PnP+LC	
	rot (deg/m)	trans (%)	rot (deg/m)	trans (%)	rot (deg/m)	trans (%)
00	0.0076	1.80	0.0067	1.57	**0.0045**	**1.07**
01	0.0082	**0.97**	**0.0050**	1.03	**0.0050**	1.03
06	0.0047	**0.96**	**0.0039**	1.11	**0.0039**	1.17

Table 5.1: Ablation study of our new proposed methods for selected KITTI dataset. "DOF-2DU" is the pose estimate of our Mahalanobis eight-point algorithm using dense optical flow with 2-dimensional uncertainty, "+PnP" is the fused pose estimate with perspective-n-point, and "+LC" is the inclusion of loop closure.

Note that VISO2-M [Geiger et al., 2011] and MLM-SFM [Song et al., 2016] methods fail to estimate the visual odometry for sequence 01 due to the highly repeated structures of the scene, which cannot be reliably matched by the sparse feature matching technique their methods employ. Figure 5.9 shows our estimated trajec-

seq	VISO2-M		MLM-SFM		PMO		DOF-1DU+LC		DOF-2DU+PnP+LC	
	rot (deg/m)	trans (%)	rot (deg/m)	trans (%)	rot (deg/m)	trans (%)	rot (deg/m)	trans (%)	rot (deg/m)	trans (%)
00	0.0209	11.91	0.0048	2.04	0.0042	1.09	0.0117	2.03	0.0045	1.07
01	n/a	n/a	n/a	n/a	0.0038	1.32	0.0107	1.149	0.0050	1.03
06	0.0157	4.74	0.0081	2.09	0.0044	1.31	0.0054	1.05	0.0039	1.17

Table 5.2: Comparison of visual odometry accuracy for VISO2-M [Geiger et al., 2011], MLM-SFM [Song et al., 2016], PMO [Fanani et al., 2017], dense optical flow with 1D uncertainty and loop closure (DOF-1DU+LC) [Ng et al., 2017] and our new proposed methods (DOF-2DU+PnP+LC) for selected KITTI dataset.

tory for the vehicle's motion.

From the estimated motion trajectory (Figure 5.9) and computed error from the ground truth (Table 5.2), we can observe that our proposed method achieved very accurate estimation of translation. This is achieved without using bundle adjustment, motion model or ground segmentation used by other state-of-the-art methods. From Table 5.1, we can also observe an improvement in the rotation estimate after fusing the Mahalanobis eight-point algorithm and PnP result.

5.7.2 Aerial Vehicle

Since our visual odometry method does not rely on restrictive motion model of the vehicle, we can easily apply our proposed method with slight modification to aerial vehicles (*e.g.* UAV). The difference with ground-based vehicle is that the camera height is not assumed constant, but is updated for each motion. This is because the unmanned aerial vehicle (UAV) can change its height arbitrarily.

Another challenge of quadcopter UAV visual odometry comes from the fact that it can rotate its yaw with no translation. This makes the pose estimation and 3D scene reconstruction highly under-constraint and error prone. We also incorporated such motion in the video sequences we use in our experiment.

For UAV video with fast motion and drastic height change, we manually select 12 images that are that are spatially close to each other to compute the loop closure constraints. This is because of the difficulty in identifying the same scene using structural similarity index (SSIM), when the scene consists of highly repetitive structures and the high degrees of freedom of the UAV motion compared to ground-based vehicle.

5.7.2.1 Small Translation with Rotation

We captured 500 frames of video from a quadcopter flying among some trees, where the scene has highly repetitive, unstructured and dynamic objects (*e.g.* leaves, cars). Due to the lack of ground truth unlike KITTI dataset, we evaluate the scale drift by reversing the frames and appended them to the end of the video, where the last frame coincides with the first frame. Figure 5.10 shows the result of our experiment.

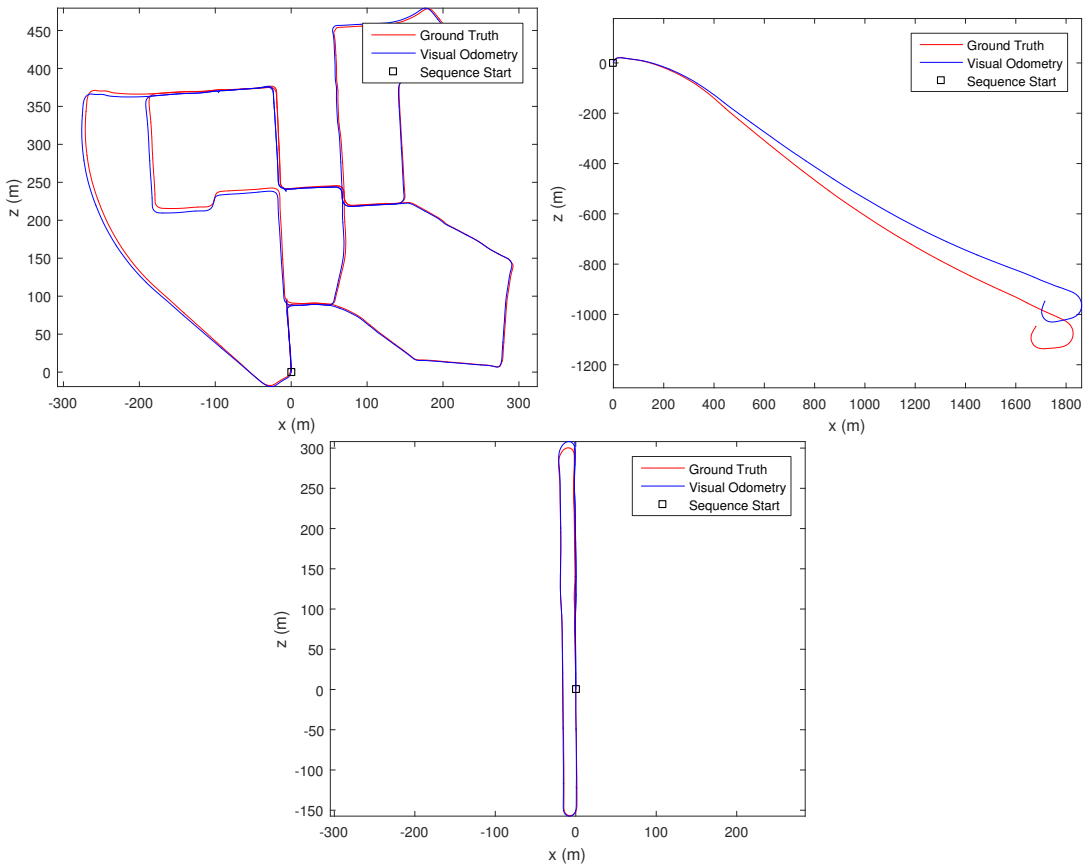

Figure 5.9: Comparison of the estimated motion trajectory and the ground truth motion. From top left to bottom: (a) sequence 00; (b) sequence 01; (c) sequence 06.

Remark 5.2. *We do not compute the loop closure constraint for this video sequence. The result shown in Figure 5.10 is pure visual odometry.*

From Figure 5.10(f), we can see that the translation scale difference remains close to zero, which shows that the scale drift is small. We also observed sudden spikes in the third plot, which corresponds to small motion as can be seen from the middle plot of Figure 5.10.

As a comparison, we also evaluated VISO2-M [Geiger et al., 2011] method on the same UAV video, using the constant camera height assumption. Figure 5.10(a)(b)(c) shows the result. We observed that the estimated pose has very large translational magnitude (wrong) when the quadcopter rotates the yaw with negligible translational motion (*e.g.* at frame 200, 150 and 100). From the third plot of Figure 5.10(c), we can also see that although the estimated scale does not drift (due to fixed camera height assumption), the estimated translational magnitude fluctuates erratically throughout the video sequence.

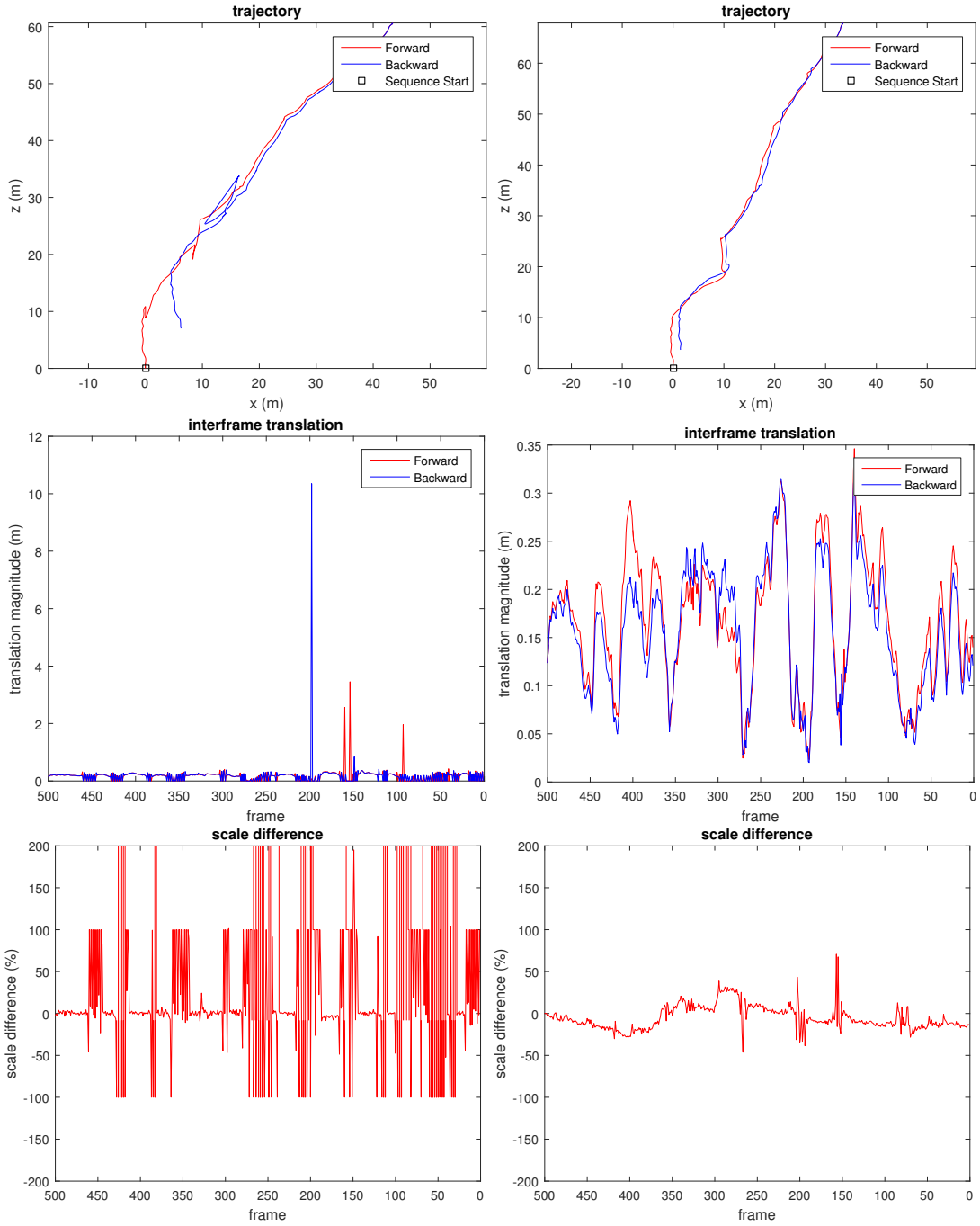

Figure 5.10: Plots evaluating the scale drift of the visual odometry on UAV video. Left column is VISO2-M (a)(b)(c), Right column is our new method (d)(e)(f). From top to bottom: (a,d) estimated motion trajectory; (b,e) inter-frame translation magnitude; (c,f) percentage scale difference (difference between the translation magnitude divided by forward magnitude).

5.7.2.2 Fast Motion With Drastic Height Changes

For the next experiment with UAV video, we captured 563 frames of a UAV flying at high speed with drastic variation in height. We have also marked some trees with yellow tapes (1m apart) to calibrate the first translational scale, and also to obtain a measure of scale drift after the UAV returns to the same spot. The error in the estimated position can also be visually observed by comparing the location of the reconstructed scene points. We plot 3D scene points with a depth standard deviation less than 0.2m for the first frame and the last frame. Figure 5.11 and Figure 5.12 shows our result.

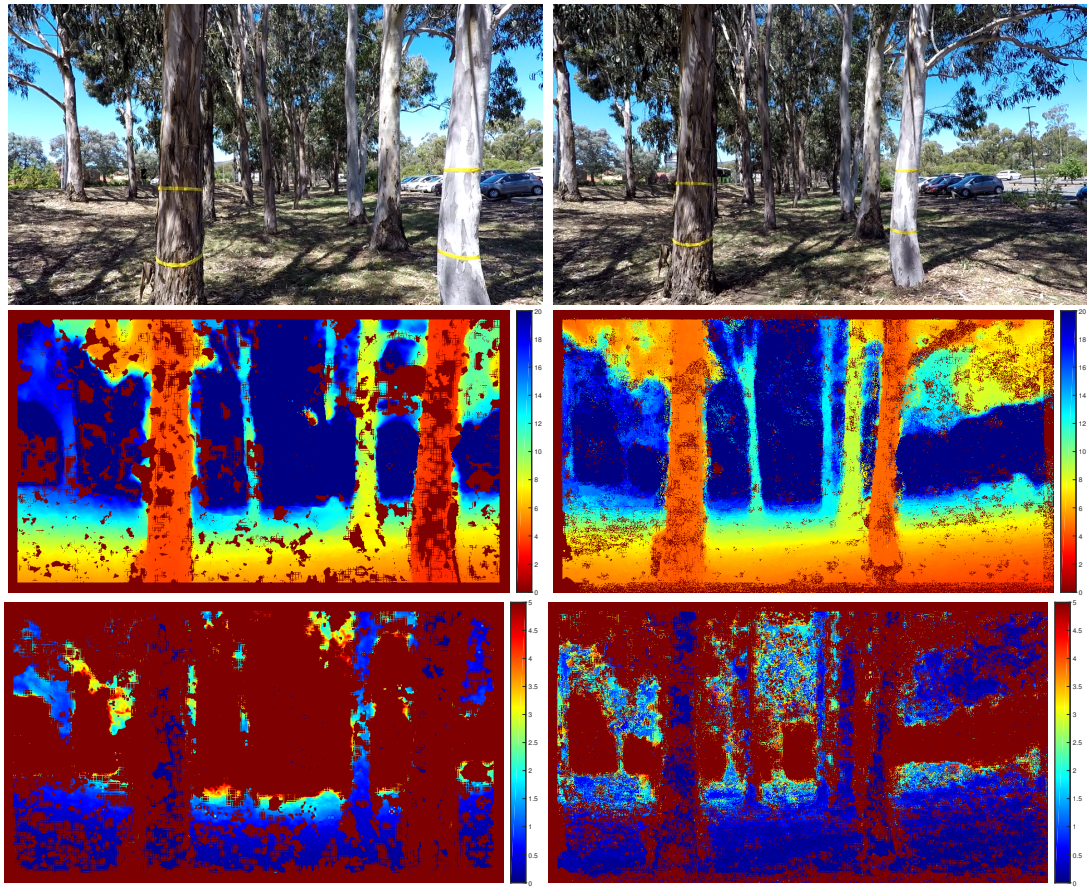

Figure 5.11: Estimated depth with standard deviation. From top to bottom: input frame, estimated depth, estimated depth standard deviation. The first column is frame 0, second column is frame 562. The scale of the colour code is in meters. Pixels that are identified as outliers are not triangulated and appears dark red in the middle plot.

From Figure 5.12(c), we can see that the error of the estimated camera pose and reconstructed 3D scene points is very small. The scale drift computed from the known distance between the tape is +5.36%. We have also computed the distance between the farthest point from the starting location, compared to GPS measurement

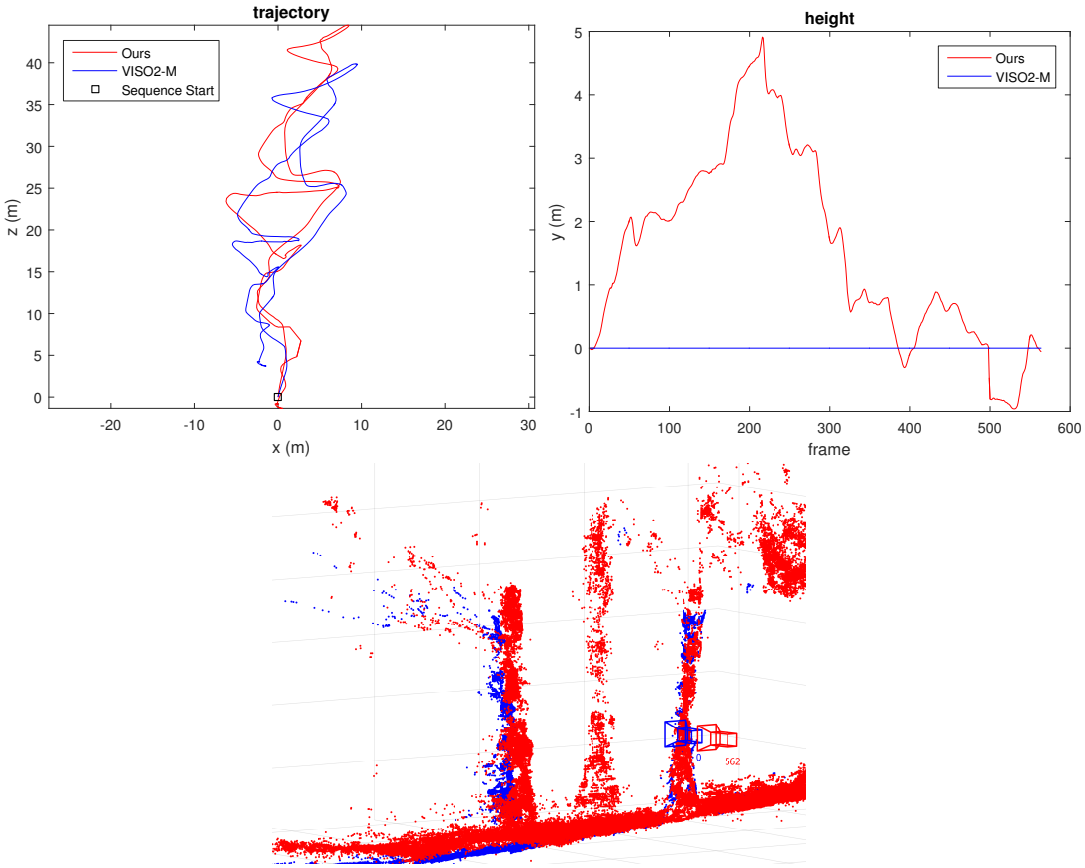

Figure 5.12: Fast moving UAV video result. From top left to bottom: (a) the estimated trajectory; (b) estimated UAV height (zero at starting height, and positive is downwards); (c) our 3D reconstruction result of the first frame (blue) and last frame (red).

and VISO2-M result. Result in Table 5.3 shows that our method agrees with GPS measurement more closely compared to VISO2-M method. Thus, this verifies that our method can accurately estimate the camera motion, regardless of the motion dynamics of the vehicle or scene structure.

Method	Distance of farthest point to origin (m)
GPS	45.81
Ours	45.50
VISO2-M	40.93

Table 5.3: Comparison of estimated distance of the farthest point from origin.

5.8 Summary

In this chapter, a novel monocular visual SLAM method that is suitable for any camera undergoing general SE(3) motion is described. The proposed method can be used in a number of practical areas where an estimate of the camera location and dense reconstruction of the scene is required. The monocular visual SLAM method utilises dense optical flow and estimates its corresponding uncertainty, which are then used as input to the new Mahalanobis eight-point algorithm. Based on MIGE, a new 3D scene point triangulation method is also proposed that achieves accurate estimate of the location and uncertainty. The performance of the new method is evaluated using simulation and real data, and compares favourably with other state-of-the-art methods that rely on additional assumptions on motion dynamics and ground segmentation, assumptions which our method does not require.

The videos showing the visual SLAM results can be found in the following link: https://www.youtube.com/watch?v=rtDui6iayLU&list=PLRKZhEGYIuwOu-mhbEJPM-V__WWcRCfPW.

5.9 Appendix: Proof of Mahalanobis eight-point algorithm

Given an initial Fundamental matrix estimate $\boldsymbol{F}$, homogeneous coordinates of matching pixels in both images $\boldsymbol{x}_i$ and $\boldsymbol{x}'_i$. Since optical flow estimates the motion of each pixels in the first image to the second image, the error is only present in coordinate of the second image pixel $\boldsymbol{x}'_i$. From Figure 5.7, let mean $\boldsymbol{\mu} = [x_0, y_0]^T$, information matrix $\boldsymbol{Y} = \boldsymbol{P}^{-1} = \begin{bmatrix} \tilde{Y}_{xx} & \tilde{Y}_{xy} \\ \tilde{Y}_{xy} & \tilde{Y}_{yy} \end{bmatrix}$, and a point on the line be $[x_1, y_1]^T = [x_1, \frac{-ax_1-c}{b}]$.

First, we calculate the minimum Mahalanobis distance between the line $\boldsymbol{l}$ and the mean image feature location $\boldsymbol{\mu}$. The minimum Mahalanobis distance is equals to the square root of the minimum squared Mahalanobis distance. The squared Mahalanobis distance ${d_M}^2$ between the feature pixel and the epipolar line is computed as follows.

$$ {d_M}^2 = \begin{bmatrix} x_1 - x_0 \\ \frac{-ax_1-c}{b} - y_0 \end{bmatrix}^T \begin{bmatrix} \tilde{Y}_{xx} & \tilde{Y}_{xy} \\ \tilde{Y}_{xy} & \tilde{Y}_{yy} \end{bmatrix} \begin{bmatrix} x_1 - x_0 \\ \frac{-ax_1-c}{b} - y_0 \end{bmatrix} \tag{5.49} $$

Expanding (5.49) and computing the first derivative of ${d_M}^2$ with respect to x_1 equals to zero provides us the solution of x_1 where ${d_M}^2$ is minimum. We then substitute this solution of x_1 back into (5.49) and apply a square root to obtain the equation of the minimum Mahalanobis distance, $\min(d_M)$ as follows.

$$ \min(d_M) = |ax_0 + by_0 + c| \sqrt{\frac{\tilde{Y}_{xx}\tilde{Y}_{yy} - \tilde{Y}_{xy}^2}{a^2\tilde{Y}_{yy} + b^2\tilde{Y}_{xx} - 2ab\tilde{Y}_{xy}}} \tag{5.50} $$

Since the original eight point algorithm minimises $|ax_0 + by_0 + c|$, the multiplica-

tive scaling is thus

$$\phi = \sqrt{\frac{\tilde{Y}_{xx}\tilde{Y}_{yy} - \tilde{Y}_{xy}^2}{a^2\tilde{Y}_{yy} + b^2\tilde{Y}_{xx} - 2ab\tilde{Y}_{xy}}}. \tag{5.51}$$

This completes the proof.

Chapter 6

Path Smoothing

This chapter presents our new path smoothing method based on window-based weighted average for a video stabilization application. The window-based weighted average method is also known as convolution, which is a well known smoothing method among the signal processing and computer vision community. It is commonly used to smooth noisy measurements in the Euclidean space. We extend this method to rotation smoothing, and propose an efficient method that uses parallel pairwise rotation averages. The key contributions of this chapter are threefold:

- The well known vector convolution method is reformulated into a set of parallelisable pairwise averaging tree. Similar method is then applied to perform rotation smoothing. The pairwise averaging tree when combined with an existing method to compute weighted average of two rotations ensures that the solution remains on the $SO(3)$ manifold.

- The performance of the propose method is verified through extensive simulation and real data experiments and compared to [Jia and Evans, 2014]. The smoothness and deviation from input metrics proposed in [Jia and Evans, 2014] are used to compare the performance of our methods.

- The proposed rotation smoothing method is applied to a video stabilization task, which shows smooth camera motion with minimal black border intrusion.

The rest of the chapter is organised as follows. Section 6.1 discusses some related work. Section 6.2 presents our reformulated vector convolution using a sequence of parallelisable weighted pairwise average. Section 6.3 introduces our rotation smoothing method similar to our reformulated vector convolution. A number of other methods with Gaussian weights are also proposed based on well known rotation averaging methods. Section 6.4 shows our experimental results using simulation and real data for video stabilization application. Finally, the chapter finishes with a summary in Section 6.5.

6.1 Related Works

A 3D pose is in the special Euclidean group $SE(3)$, which contains the rotation and translation component. In our review of existing translation smoothing methods, we will look at the more widely studied works on image smoothing due to their similarity. In particular, a sequence of translations can be interpreted as a single line of colour image pixels. Thus, similar techniques can be applied to obtain a smooth translational trajectory.

Gaussian convolution is the most well-known image smoothing technique in computer vision community [Deng and Cahill, 1993]. A convolution can be written as

$$(f * g)(t) = \int_{\infty}^{-\infty} f(\boldsymbol{\tau}) g(t - \boldsymbol{\tau}) d\boldsymbol{\tau} \tag{6.1}$$

and the discrete form is

$$(f * g)(t) = \sum_{\tau=1}^{n} f(\tau) g(t - \tau) \tag{6.2}$$

The convolution operation can be seen as a weighted average of neighbouring vectors. When applied to smooth a sequence of noisy input, a sliding window approach is used to compute the smoothed result.

For impulse noise, it was known that a median filter is very effective in obtaining a smoother value [Tukey, 1977]. [Huang, 1997] used a median filter and convolutional smoothing to remove image noise for video compression application.

Due to the relative simplicity and the mature research works that already exist, we will not focus on proposing a new translation smoothing method. Instead, we focus our effort on the more challenging task of designing a new method suitable for rotation smoothing.

Unlike translation, the variable of interest in 3D rotation smoothing algorithm lies on Special Orthogonal $SO(3)$ manifold. In general, the solution of a simple weighted average of $SO(3)$ does not remain on the $SO(3)$ manifold. Thus, more sophisticated methods that ensures the solution remains on the manifold, or ones that project an arbitrary point back to the manifold are required.

In order to apply similar technique as vector smoothing, we explore the literature on rotation average methods. Moakher [2002] presented a method to compute the weighted average of two rotation matrices. An alternative method is to use rotations in quaternion form to find the weighted average of two rotations, using a method called *slerp* [Shoemake, 1985]. *Slerp* relies on linear interpolation on non-unique quaternion representation ($\boldsymbol{q} = -\boldsymbol{q}$), which suffers from rotation direction potentially changing abruptly. This means that the weighted average found using *slerp* may not lie on the shortest geodesic curve between the two rotations. Thus, the weighted average using rotation matrices is more suitable and stable.

On the other hand, an iterative algorithm to compute the Geodesic L_2 mean of multiple rotational matrices was presented in Hartley et al. [2011, 2012]. Suppose $\boldsymbol{R}_i$ is the input rotations matrix within a sliding window, and $\boldsymbol{R}$ is the smoothed rotation

matrix in the middle of the window. Then, the rotation minimizing cost function for Geodesic L_2 mean is, $C(\boldsymbol{R}) = \sum_{i=1}^{n} d_{\angle}(\boldsymbol{R}, \boldsymbol{R}_i)^2$. Geodesic L_2 Mean is also known as the Karcher mean of rotations, or the geometric mean [Moakher, 2002].

In addition, Geodesic L_q Mean can be found by using Iteratively Reweighted Least Square (IRLS) Method. IRLS is also called the L_q Weiszfeld Algorithm [Aftab et al., 2015]. It is a method that minimises the cost function, $C_q(\boldsymbol{R}) = \sum_{i=1}^{n} d_{\angle}(\boldsymbol{R}, \boldsymbol{R}_i)^q$. As the name suggests, it relies on the iterative Geodesic L_2 Mean algorithm by adding an extra reweighing factor, $w_i = d_{\angle}(\boldsymbol{R}, \boldsymbol{R}_i)^{q-2}$. This reweighing for different $\boldsymbol{R}$ and $\boldsymbol{R}_i$ pair changes the gradient descent Geodesic L_2 Mean method to solve for Geodesic L_q Mean instead. The distance, $d_{\angle}(\boldsymbol{R}, \boldsymbol{R}_i) = (1/\sqrt{2})||\log(\boldsymbol{R}^{-1}\boldsymbol{R}_i)||_F$, where $||*||_F$ is the Frobenius norm of the matrix, and the scaling $(1/\sqrt{2})$ ensures that $d_{\angle}(*,*)$ represents the angular distance between the two rotations.

When $q = 1$, the algorithm finds the Geodesic L_1 Mean solution, which is known to be more robust against outliers in the input. However, Geodesic L_1 mean computation is also known to be slower than Geodesic L_2 mean, and there is also a possibility of the solution getting stuck when $\boldsymbol{R}$ is equal to one of the input, $\boldsymbol{R}_j \in \boldsymbol{R}_i$. [Aftab et al., 2015] When $\boldsymbol{R}$ is equal to one of the input $\boldsymbol{R}_j$, the weighing factor, w_j (in Algorithm 8) becomes $\frac{1}{0} = \infty$. Thus, all the other weighing factors will be insignificant compared to the weighing factor of $\boldsymbol{R}_j$, and the rotation will be unchanged.

We can partially overcome the slow convergence by choosing $1 < q < 2$, as discussed by Aftab et al. [2015]. However, this does not solve the problem of the solution getting stuck when $\boldsymbol{R}$ is equal to $\boldsymbol{R}_j$, because w_j still approaches ∞ as $\boldsymbol{R}$ approaches $\boldsymbol{R}_j$, albeit at a slower rate.

Like all iterative algorithms, there needs to be a good initial estimate of $\boldsymbol{R}$. As suggested by Aftab et al. [2015], the initial estimate can be found by Chordal L_2 Mean, which has a closed-form solution.

Chordal L_2 Mean is defined as the rotation which minimises the cost, $C(\boldsymbol{R}) = \sum_{i=1}^{n} d_{chord}(\boldsymbol{R}, \boldsymbol{R}_i)^2$. It is also named the projected or induced arithmetic mean [Moakher, 2002; Sarlette and Sepulchre, 2009]. The algorithm to compute Chordal L_2 Mean is given by Hartley et al. [2012], which uses Singular Value Decomposition (SVD) instead of polar decomposition used in [Moakher, 2002; Sarlette and Sepulchre, 2009]. Reprojecting the algebraic sum of the rotational matrices onto the orthogonal $SO(3)$ manifold is also called the Orthogonal Procusthes Problem [Schönemann, 1966] [Everson, 1997].

Other cost functions have also been proposed to smooth an input sequence of rotations. Jia and Evans [2014] proposed to minimise the cost function

$$\min_{\{\mathbf{R}_i\}} \sum_{i=1}^{N} d(\tilde{\mathbf{R}}_i, \mathbf{R}_i) + \alpha \sum_{i=1}^{N-1} d(\mathbf{R}_i, \mathbf{R}_{i+1}), \tag{6.3}$$

where $d(*,*)$ is any suitable distance metric in $SO(3)$, α is the scalar factor controlling the smoothness of the output trajectory (a trade-off against deviation from input - the

first summation term in the cost function), $\tilde{\mathbf{R}}_i$ is the input (measured) orientation at the i^{th} instance, $\mathbf{R}_i$ is the smoothed orientation at the i^{th} instance.

After a cost function is specified, it can be optimised using iterative methods like gradient descent [Hartley et al., 2011], Newton's method [Jia and Evans, 2013], Lagrangian Duality [Fredriksson and Olsson, 2013], or Iteratively Reweighted Least Square [Aftab et al., 2015; Chatterjee and Govindu, 2013].

On the other hand, there are also works based on stochastic filtering methods [Ertürk, 2002; Glover and Kaelbling, 2013]. However, the $SO(3)$ manifold is not convex. Thus, stochastic filtering method requires an unbiased prior, which is not always available in practice. Also, due to the "future measurements" not being used, the output of stochastic filtering methods may produce a result that has a bigger bias from the actual motion compared to methods that uses that extra information.

In comparison, we proposed a new efficient rotation smoothing method that is inspired by the Gaussian convolution method commonly used for vector smoothing. The weight assigned to the middle of the sliding window is the highest, and decreases to zero the farther away it is from the middle. This ensures that the solution stays close to the input rotation, following the assumption that close-by neighbours tend to have a similar orientation. The weighted average of rotation is then computed efficiently using the proposed pairwise weighted averaging tree.

6.2 Translation smoothing

For translation smoothing, we use the well established method of window-based weighted average, which is also known as convolution method. The vector convolution is reformulated into a sequence of pairwise weighted averages, which are parallelisable for faster computation.

6.2.1 Pairwise Gaussian Weighted Average of 2^n Vectors

A Gaussian filter kernel can be calculated using

$$G(t|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(t-\mu)^2}{2\sigma^2}}. \tag{6.4}$$

For a discrete case, to ensure that the resulting filtered signal has the same scale as the original, we can make sure that the elements of the kernel sum to 1 by a simple normalisation, such that

$$G_{norm}(t|\mu,\sigma^2) = \frac{G(t|\mu,\sigma^2)}{\sum(G(t|\mu,\sigma^2))}. \tag{6.5}$$

The Gaussian Kernel values are calculated by using t with equal spacing, centred around zero ($\mu = 0$). Otherwise, for temporally invariant Gaussian Kernel, the value of t is set to the time at each measurement, and μ is the time of the middle entry of the input (mean of Gaussian distribution).

We then introduce a method to rearrange a normalised weighted sum of two vectors, into a form similar to the equation of weighted average of two rotations as shown in (6.9).

A normalised weighted sum of two vectors $(\boldsymbol{x}_1, \boldsymbol{x}_2)$ with different weighing factor (g_1, g_2) is equivalent to the difference in value $\boldsymbol{x}_2 - \boldsymbol{x}_1$ multiplied by the ratio of the weighing factor of $\boldsymbol{x}_2$ to the total weighing factor, added to $\boldsymbol{x}_1$. The derivation is included as

$$\begin{aligned} \frac{1}{g_1+g_2}(g_1\boldsymbol{x}_1 + g_2\boldsymbol{x}_2) &= \frac{1}{g_1+g_2}((g_1+g_2)\boldsymbol{x}_1 + g_2(\boldsymbol{x}_2 - \boldsymbol{x}_1)) && (6.6)\\ &= \boldsymbol{x}_1 + \frac{g_2}{g_1+g_2}(\boldsymbol{x}_2 - \boldsymbol{x}_1) && (6.7)\\ &= \boldsymbol{x}_1 + \lambda(-\boldsymbol{x}_1 + \boldsymbol{x}_2). && (6.8) \end{aligned}$$

We propose that the weighted average of 2^n vectors can be decomposed into a sequence of pairwise averages. This will be useful in the next section. The operation can be illustrated as shown in Figure 6.1.

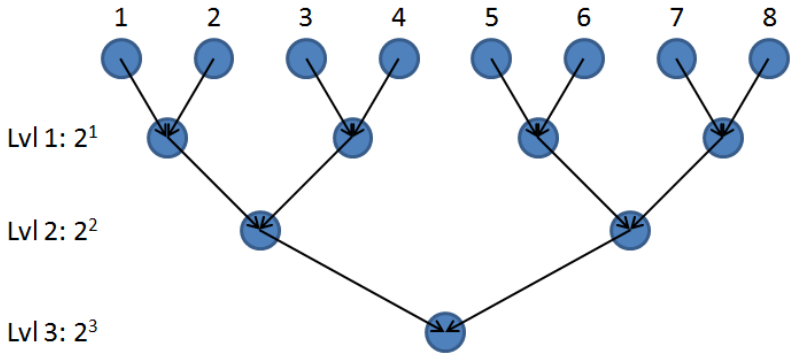

Figure 6.1: 2^n Averaging Tree

Each arrow in Figure 6.1 shows a normalised weighted average operation between two vectors. The updated weighing factor of each average operation is the sum of the corresponding weighing factors of that average.

This method is equivalent to a weighted averaging filter (convolution), with window size of 2^n, which is capable of smoothing an otherwise noisy input vector (e.g. translational component of $SE(3)$).

6.3 Rotation smoothing

Here, a number of window-based rotation smoothing methods are proposed. They are designed to operate on orthogonal rotational matrices.

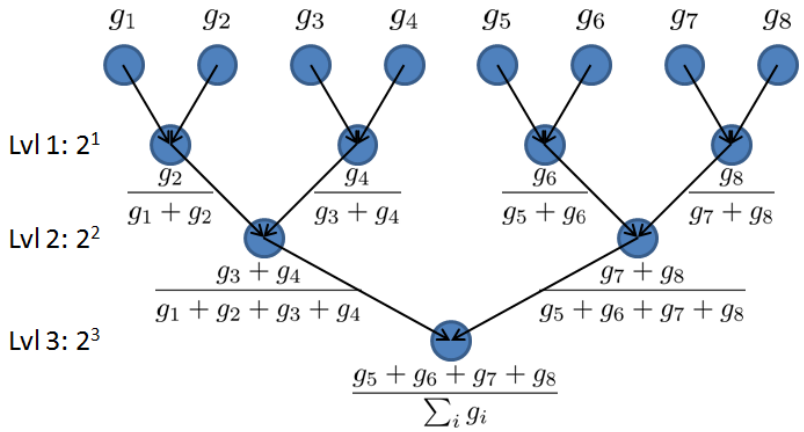

Figure 6.2: 2^n Weighted Averaging Tree, with their weight, λ in (6.9) shown below the nodes

6.3.1 Gaussian Weighted Average of 2^n Rotations

The weighted average of two rotations, which lies on the shortest geodesic curve connecting the two rotations can be calculated as follows.

$$R_{weightedAve} = R_1.exp(\lambda.log(R_1^T R_2)) \tag{6.9}$$

where, $log(*)$ is the matrix logarithm, and $exp(*)$ is the matrix exponential. The exponential and logarithm of the rotation group are also discussed in Moakher's paper [Moakher, 2002].

In the rest of this section, we propose a generalised method to perform weighted average of multiple rotations deterministically.

In order to find a weighted average of rotations in a window size of 2^n, a method similar to that presented in Section 6.2.1 may be used. Instead of vectors, each circle (node) represents a rotation. This is well defined because the weighted average of two input rotations can be calculated exactly using (6.9).

Similar to weighted average of 2^n vectors, only the ratio of their corresponding weight matters. After each pairwise average, their resulting weighing factor is equivalent to the sum of their corresponding weights. Figure 6.2 illustrates this concept with an example.

With this generalised method in finding the weighted average of 2^n rotations, we can then do Gaussian filtering in a similar way to the vector case (Section 6.2.1).

Although for the case of rotation, Bingham Distribution is more appropriate due to the wrap around effect, but as discussed in [Kurz et al., 2013], it was shown that for standard deviation less than 11°, Gaussian Distribution is a good approximation.

It is also noted that there is a need to account for the delay introduced by the filter, which is equivalent to $\frac{2^n+1}{2}$, as can be seen from the graph in Figure 6.1. The resulting average is thus a value for rotation between the forth and fifth values used.

In order to reposition the averaged value to align to an input time interval, we

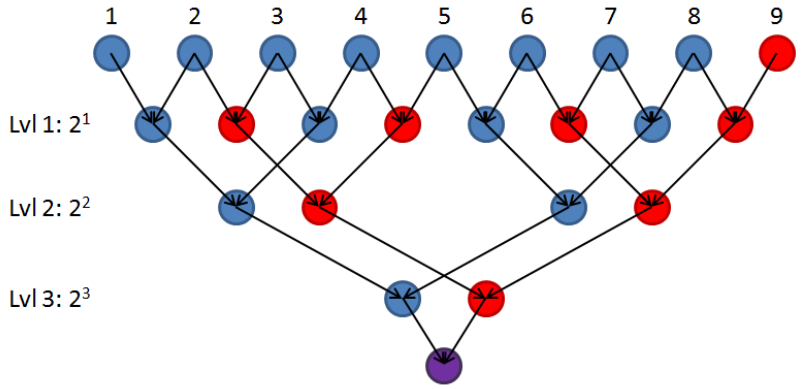

Figure 6.3: 2^n Averaging Tree with Value Reposition

can do another average between subsequent averaged value as shown in Figure 6.3. This is equivalent to an interpolation step of the two consecutive rotation averages.

The red circles are the values used and computed for one time step after the blue circles, and the purple value is the repositioned average (which is aligned to the fifth input value).

By combining the proposed pairwise weighted average of 2^n rotations and the Gaussian Filter technique, we have found a way to smooth a sequence of 3D rotation data.

It is noted that every layer (or level) contains completely independent computations of pairwise rotation averaging. Thus, they are parallelisable for faster computation. For example, Figure 6.3 shows window size of 9, and there are a total of 15 pairwise averages, but after parallelisation, only 4 dependent levels are left (a potential for 3.75× shorter computation time).

We can summarise the pairwise method in Algorithm 6 as follows.

Algorithm 6: Gaussian Weighted Pairwise Average on $SO(3)$

Data: Given a sequence of rotations $\{\boldsymbol{R}_k\}$
Result: Smooth rotations $\{\tilde{\boldsymbol{R}}_k\}$
while *time k is increasing* **do**
  for *Level 1 parallel loop* **do**
    Compute $\boldsymbol{R}_{lvl1,i} = \boldsymbol{R}_{2i-1} \exp(g_i . \log(\boldsymbol{R}_{1+2i}^T \boldsymbol{R}_{2i}))$, where $i \in [1, N]$, and N is half the window size ;
  end
  for *Level 2 parallel loop* **do**
    Compute $\boldsymbol{R}_{lvl2,i} = \boldsymbol{R}_{lvl1,2i-1} \exp(\lambda_i . \log(\boldsymbol{R}_{lvl1,1+2i}^T \boldsymbol{R}_{lvl1,2i}))$, where λ is the updated weighing factor (Fig. 6.2)
  end
  ... (more parallel loops until the last two averages) ;
end

6.3.2 Gaussian Weighted Geodesic L_2 Mean

We have also explored the possibility of using different weighing factors, g_i when doing the iterative Geodesic L_2 distance minimisation. In essence, we just modify the 3^{rd} line of Algorithm 1 in [Hartley et al., 2012]. The new method is shown in Algorithm 7.

Algorithm 7: Gaussian Weighted Geodesic L_2 Mean on $SO(3)$

Data: Given a sequence of rotations $\{\boldsymbol{R}_k\}$
Result: Smooth rotations $\{\tilde{\boldsymbol{R}}_k\}$
while *time k is increasing* **do**
  Set $\boldsymbol{R} := \boldsymbol{R}_{mid}$. Choose a tolerance $\epsilon > 0$;
  while *true* **do**
    Compute $\boldsymbol{S} := \sum_{i=1}^{n} g_i . \log(\boldsymbol{R}^T \boldsymbol{R}_i)$;
    if $||\boldsymbol{S}|| < \epsilon$ **then**
      return $\tilde{\boldsymbol{R}}_k = \boldsymbol{R}$;
    end
    Update $\boldsymbol{R} := \boldsymbol{R} . \exp(\boldsymbol{S})$;
  end
end

6.3.3 Gaussian Weighted Geodesic L_q Mean

Similar to Weighted Geodesic L_2 Mean, we can add an extra weighing factor, g_i to the Geodesic L_q Mean. The resulting algorithm is given in Algorithm 8.

Algorithm 8: Gaussian Weighted Geodesic L_q Mean on $SO(3)$

Data: Given a sequence of rotations $\{\boldsymbol{R}_k\}$
Result: Smooth rotations $\{\tilde{\boldsymbol{R}}_k\}$
while *time k is increasing* **do**
  Set $\boldsymbol{R} := \boldsymbol{R}_{initial}$. Choose a tolerance $\epsilon > 0$;
  while *true* **do**
    Compute $\boldsymbol{S} := \left(\sum_{i=1}^{n} g_i . w_i . \log(\boldsymbol{R}^T \boldsymbol{R}_i) \right) / \left(\sum_{i=1}^{n} w_i \right)$, where,
    $w_i = (d_{\angle}(\boldsymbol{R}, \boldsymbol{R}_i))^{q-2}$;
    if $||\boldsymbol{S}|| < \epsilon$ **then**
      return $\tilde{\boldsymbol{R}}_k = \boldsymbol{R}$;
    end
    Update $\boldsymbol{R} := \boldsymbol{R} . \exp(\boldsymbol{S})$;
  end
end

6.3.4 Gaussian Weighted Chordal L_2 Mean

Instead of a simple algebraic sum in the original Chordal L_2 Mean, we can do a weighted sum instead. The newly defined C_e is as shown in Line 1 of Algorithm 9.

Algorithm 9: Gaussian Weighted Chordal L_2 Mean on $SO(3)$

Data: Given a sequence of rotations $\{\boldsymbol{R}_k\}$
Result: Smooth rotations $\{\tilde{\boldsymbol{R}}_k\}$
1 **while** *time k is increasing* **do**
2 Compute $C_e = \sum_{i=1}^{n} g_i.\boldsymbol{R}_i \in \mathbb{R}^{3\times 3}$;
3 Compute SVD, $\boldsymbol{C}_e = \boldsymbol{U}\ \boldsymbol{D}\ \boldsymbol{V}^T$, where diagonal elements of $\boldsymbol{D}$ is arranged in descending order ;
4 **if** $det(\boldsymbol{U}\boldsymbol{V}^T) \geq 0$ **then**
5 $\boldsymbol{R}_k = \boldsymbol{U}\boldsymbol{V}^T$;
6 **else**
7 $\boldsymbol{R}_k = \boldsymbol{U}.\text{diag}([1, 1, -1]).\boldsymbol{V}^T$;
8 **end**
9 **end**

The addition of Gaussian weighs to the Chordal L2 Mean method makes the algorithm more robust against averaging large angular motion, because the weight given to the middle of the window is higher than those further to the edges. Thus, preserving the continuity of the motion.

The results using our Pairwise Average method, Jia and Evan's method, the Weighted Geodesic L_2 Mean method, and other window-based methods discussed are presented and compared in Section 6.4.

6.4 Experimental Results

Simulation is performed to evaluate the performance of our proposed rotation smoothing method. Real data experiments were also conducted by using rotation estimated from inertial sensor. The smoothed rotational trajectory is then applied to video stabilization task.

6.4.1 Simulation

The simulated ground truth data is obtained by having a combination of sinusoidal and constant change in each 'ZYX' rotation angles (simulated smooth motion), then a Gaussian noise with standard deviation of 0.1 radian is added to each of the rotation angles. This is then transformed into Rotation matrix using MATLAB's in-build function, *angle2dcm* to obtain the simulated noisy input.

To represent the smoothness of the rotation (orientation) sequence, in Figure 6.4 we plotted the relative rotational angle (distance metric chosen) between consecutive orientations.

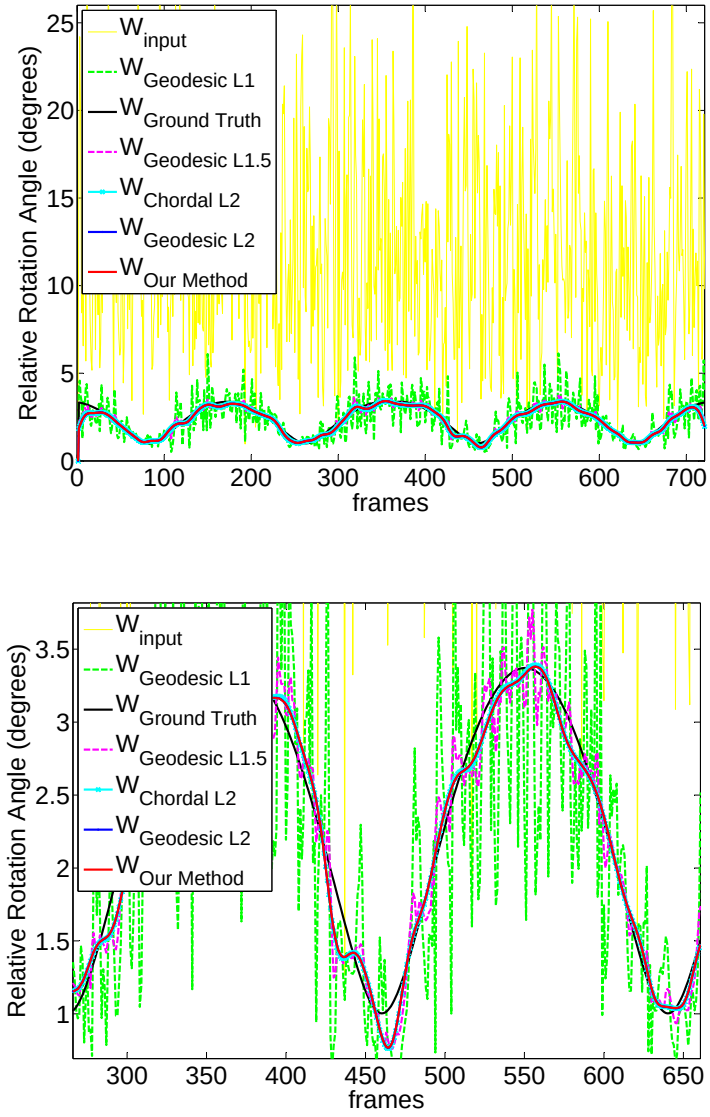

Figure 6.4: Simulation Result of Relative Rotational Angle of Consecutive Orientations, Window Size = 65, Standard Deviation = 8. From left to right: (a) Whole sequence; (b) Zoom in.

From Figure 6.4, we can see that our method has successfully produce a smoothed orientation data (red) very close to the ground truth (black).

In order to determine how close our Pairwise method approximate Geodesic L_2 mean, we can check the norm of $\mathbf{r}$ in Line 3 of Algorithm 7. In Figure 6.5, we can see that the Pairwise average method is accurate up to a tolerance, $\epsilon < 10^{-3}$, while Chordal L2 Mean method is up to 7 times less accurate.

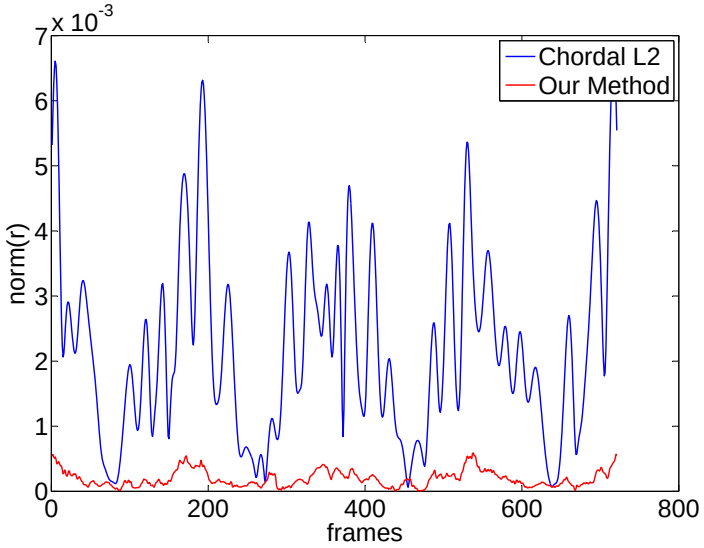

Figure 6.5: Simulation Result of Difference to Geodesic L_2 Mean as Illustrated by Norm of $\boldsymbol{r}$ in Algorithm 7, Window Size = 65, Standard Deviation = 8

Figure 6.6 shows the simulation result by superimposing previous frames onto the current frame to produce artificial motion blur. More motion blur corresponds to a shaky video.

6.4.2 Video Stabilisation - Walking Sequence

In most mobile robotics systems, inertial measure unit (IMU) is a crucial component for estimation of the robot's relative location and orientation. The IMU is also present in most smartphones these days. In the following experiments, the gyroscope in IMU is used to estimate the camera orientation at each image frame.

By using the video sequence tested by Jia and Evans [2014], we compare our result in this subsection. Similar to their method, we assume the input video sequence has undergone rolling shutter rectification.

The camera is assumed to follow a pure rotational camera model, and the difference between the smoothed and original camera orientation is used to warp the input video by a Homography (projective transformation).

We know that the Gaussian kernel's standard deviation, σ is related to the factor α in (6.3). Thus, we tune the σ until our smoothed curve lies close to that obtained

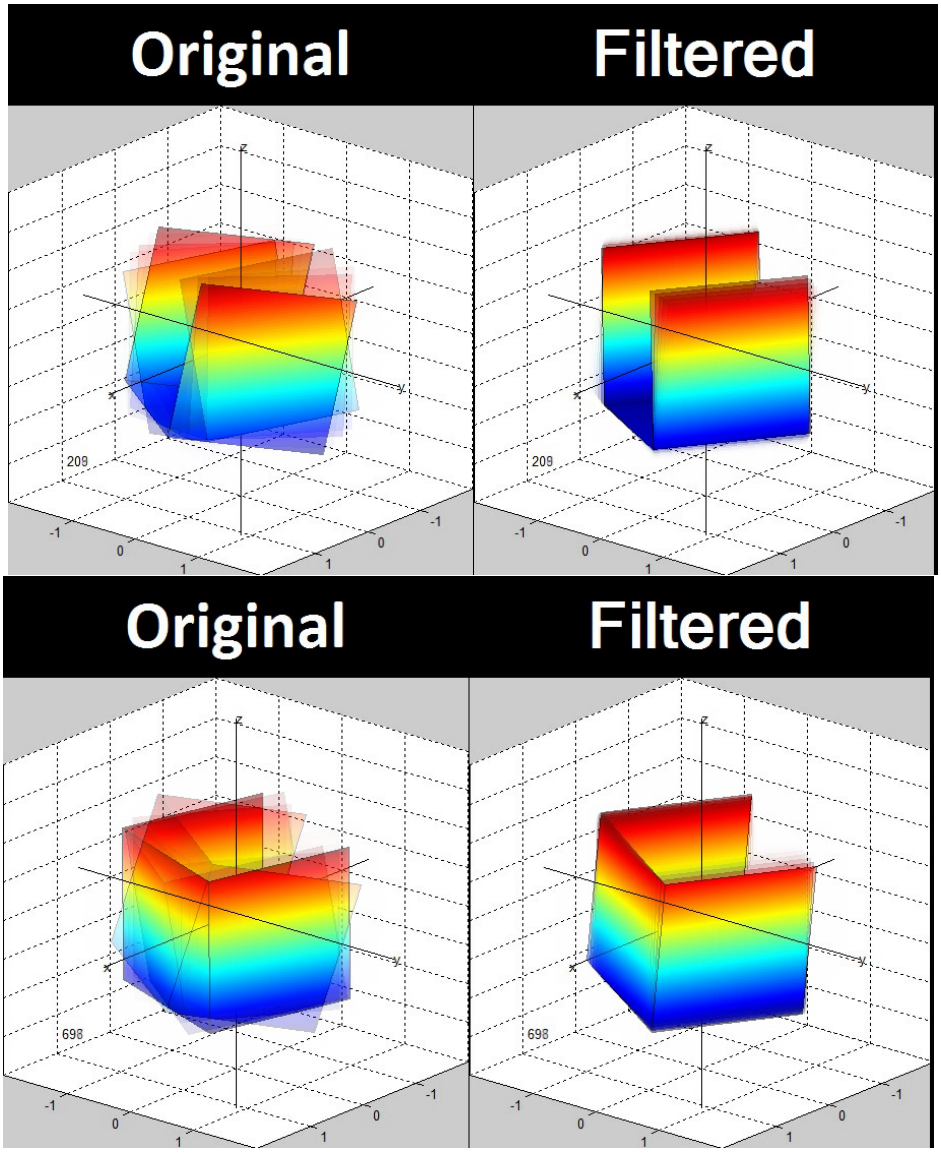

Figure 6.6: Simulation Result at the Corresponding Frames using Our Pairwise Average Method, where the Motion is Represented by Motion Blur. From left to right: (a) 209^{th} frame; (b) 698^{th} frame.

by Jia and Evans method [Jia and Evans, 2014].

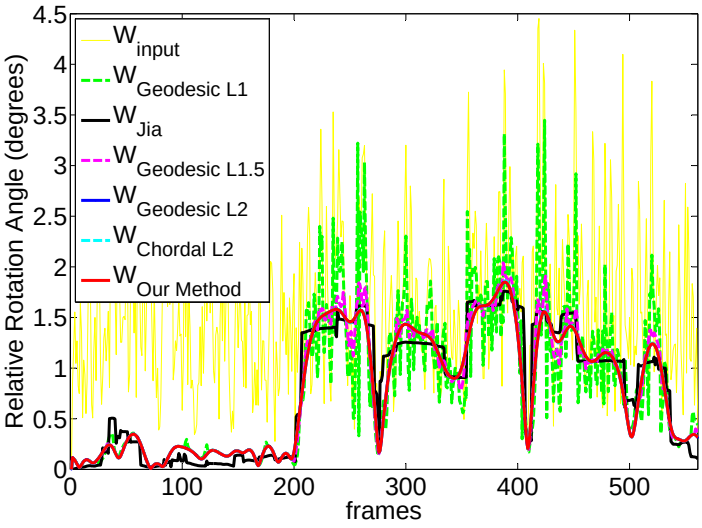

Figure 6.7: Relative Rotational Angle of Consecutive Orientations in Our Pairwise Smoothing Result (Red) VS Jia and Evans's Smoothing Result (Black), and Other Window-based Smoothing Methods (Test Video in [Jia and Evans, 2014])

In Figure 6.7, we can see that the result from our pairwise method (Red) is smoother than Jia and Evans's result (Black). This could be due to the extra constraint included by Jia and Evans to restrict the maximum angular deviation from the input rotation, as can be seen from Figure 6.8.

The maximum angular deviation is added to ensure that the warped frame has an upper limit on the amount of black border intruding into the view. This was done by reprojection of the gradient to be within the set bound [Jia and Evans, 2014]. We did not have this because we found that our method produces little, instantaneous black border intrusion (< 5 continuous frames, or $< 0.167s$) to justify the extra computation.

Figure 6.9 contains a boxplot showing the perturbation represented by the relative angle (geodesic distance) between consecutive orientations. From this figure, we can also see that Jia and Evans method produces a result with higher median than the other window-based methods we have proposed.

From the third plot in Figure 6.10, we can also see that our method follows the mean of the input (blue curve) more closely than Jia and Evans's method.

We have also implemented the Pairwise Average method, Geodesic L_2 Mean, and Chordal L_2 Mean in C++ to compare the computational speed between the three smoothing methods. These are included in parenthesis of the last column in Table 6.1.

The matrix operations in C++ are programmed with the help of Eigen 3.2.4 library [Jacob and Guennebaud, 2015]. It is noted that the C++ implementation has

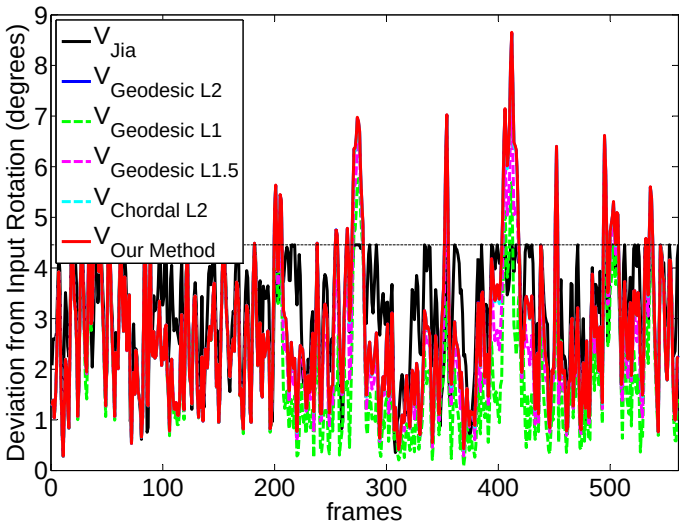

Figure 6.8: Rotational Angle Deviation from Input in Our Pairwise Smoothing Result (Red), Jia and Evans's Smoothing Result(Black), and Other Window-based Smoothing Methods (Test Video in [Jia and Evans, 2014])

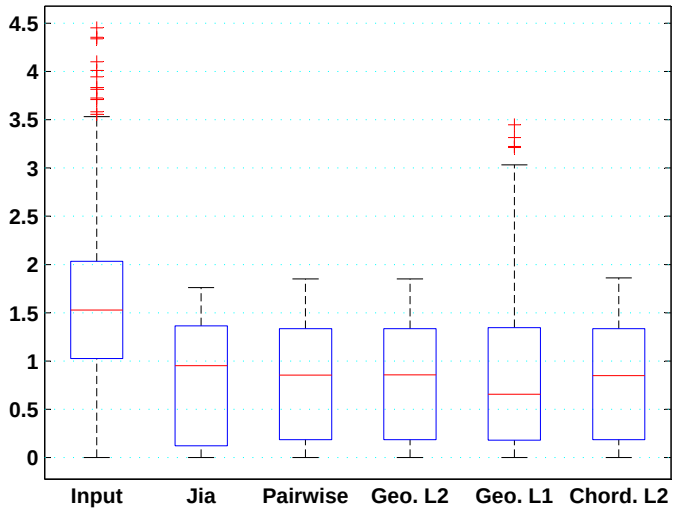

Figure 6.9: Boxplot Showing the Distribution of the Relative Angle between Consecutive Orientations (Test Video in [Jia and Evans, 2014]). Red line is the median of the distribution, top and bottom line of the box represents 75th and 25th percentiles respectively, and red "+" shows the outliers

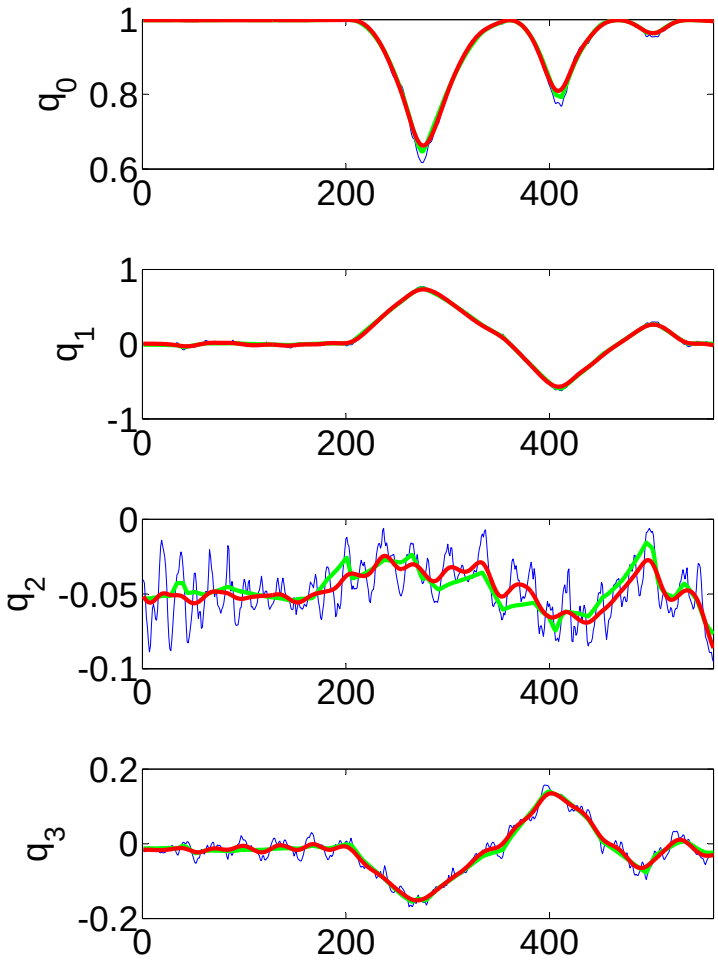

Figure 6.10: Quaternion Representation of Input (Blue), Jia and Evans's Result (Green), and Our Pairwise Method (Red) (Test Video in [Jia and Evans, 2014])

Table 6.1: Comparisons of Our Pairwise Method, Jia and Evan's, Geodesic L_2, Geodesic L_1, Geodesic $L_{1.5}$ Mean, and Chordal L_2 Mean on the video used in [Jia and Evans, 2014]. The Geodesic L_2 Distance is the Square of Relative Rotational Angle between Consecutive Frames. Numbers in Parenthesis is the Computation Time in C++ Implementation, Non-Parenthesised are MATLAB Implementation

	Geodesic L_2 Distance Sum		For 561 Frames
	Relative Rotation	Deviation from Input	Comp. Time (s)
Input	1737.96	0	-
Pairwise Method	**532.61**	5382.8	4.23 (**0.79**)
Jia and Evans's	559.96	6067.5	28.88
Geodesic L_2	**532.96**	5367.3	4.50 (1.44)
Geodesic L_1	657.89	3301.8	82.37
Geodesic $L_{1.5}$	554.68	4550.2	34.88
Chordal L_2	533.98	5322.9	1.21 (**0.01**)

lower precision than MATLAB's implementation, and the iterative Geodesic L_2 Mean needs a higher tolerance, ϵ, and setting a maximum number of iterations for the program to converge.

In C++ implementation of Geodesic L_2 Mean, $\epsilon = 10^{-3}$, and maximum number of iterations is set to 6, whereas MATLAB implementation has $\epsilon = 10^{-6}$ with no upper bound on maximum number of iterations.

Figure 6.11, 6.12 and 6.13 shows the feature trajectories in the next 10 frames to visualise the difference in camera motion after stabilization.

6.4.3 Video Stabilisation - Standing Sequence

Another experiment was conducted using a Sony Xperia Z2 smartphone and its onboard gyroscope. A short video is taken by a person at a T-junction making occasional panning of the camera. This video represents a slightly different camera motion, which examines videos with smaller camera shake than the previous video.

Different stabilization methods are tested, along with the same parameters used for the previous video. Figure 6.14 shows the smoothness metric, while Figure 6.15 shows the deviation from the original path.

From Figure 6.15, it is clear that Jia and Evan's method has overly compensated for the camera motion, since there are large and wide peaks that corresponds to motion that are not caused by noise but are removed by their method. On the other hand, the other window-based averaging method remove only the noisy camera motion.

Table 6.2 shows the comparison of the two performance metrics between different rotation smoothing methods tested.

From Table 6.2, we can again observe that although Jia and Evan's method produces a result that is smoother, it deviates from the input camera motion a lot more

Figure 6.11: Video Stabilisation Input Video (used in[Jia and Evans, 2014]) at the Corresponding Frames. From top to bottom: (a) 36^{th} frame; (b) 456^{th} frame; (c) 502^{th} frame.

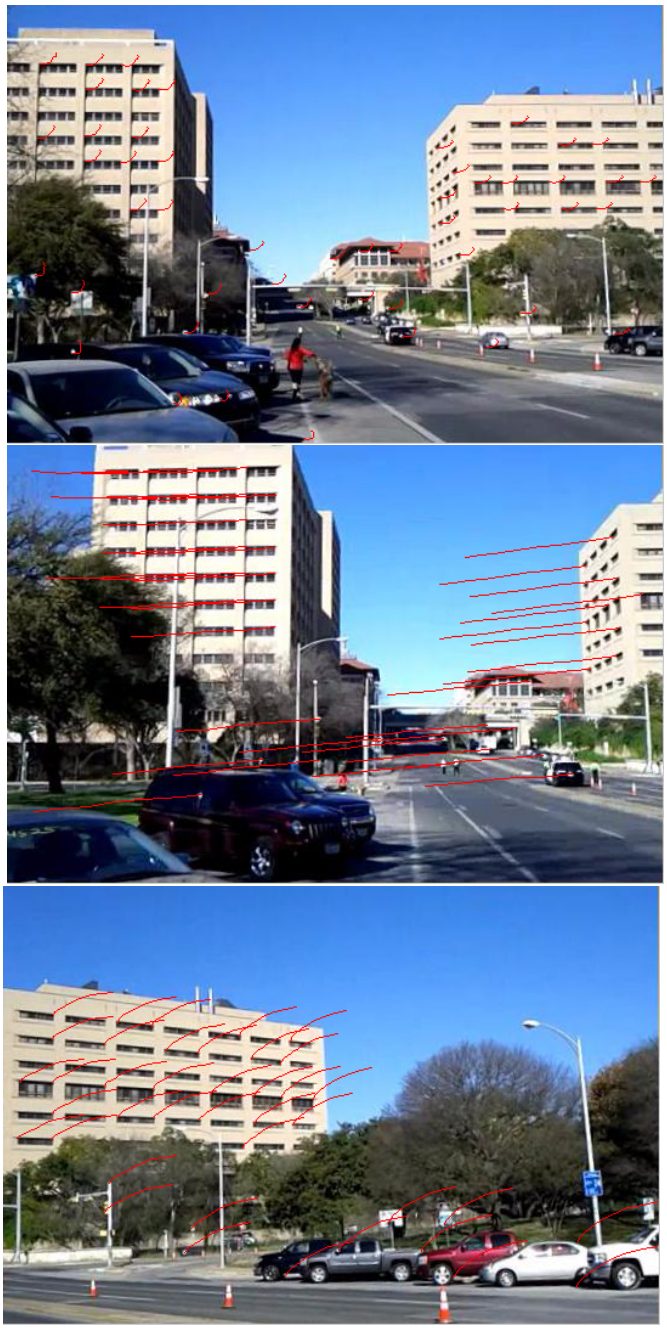

Figure 6.12: Video Stabilisation Result with Our Pairwise Method at the Corresponding Frames. From top to bottom: (a) 36^{th} frame; (b) 456^{th} frame; (c) 502^{th} frame.

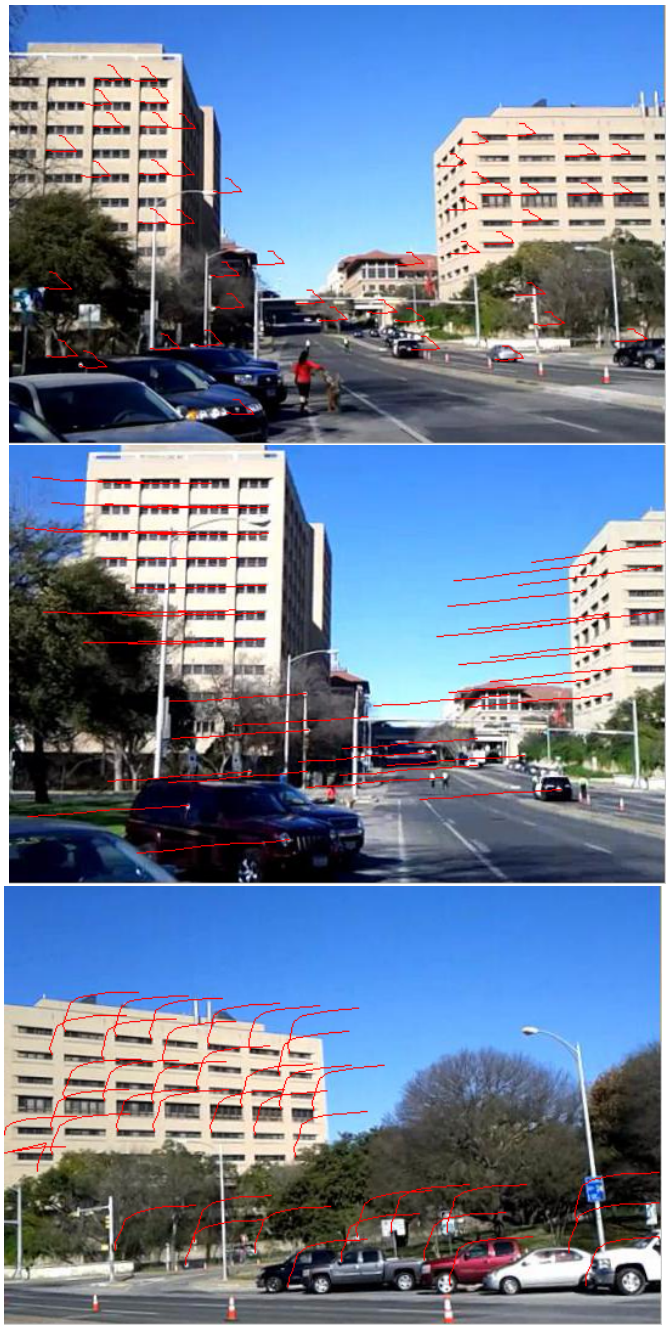

Figure 6.13: Video Stabilisation Result with Jia and Evans's Method at the Corresponding Frames. From top to bottom: (a) 36^{th} frame; (b) 456^{th} frame; (c) 502^{th} frame.

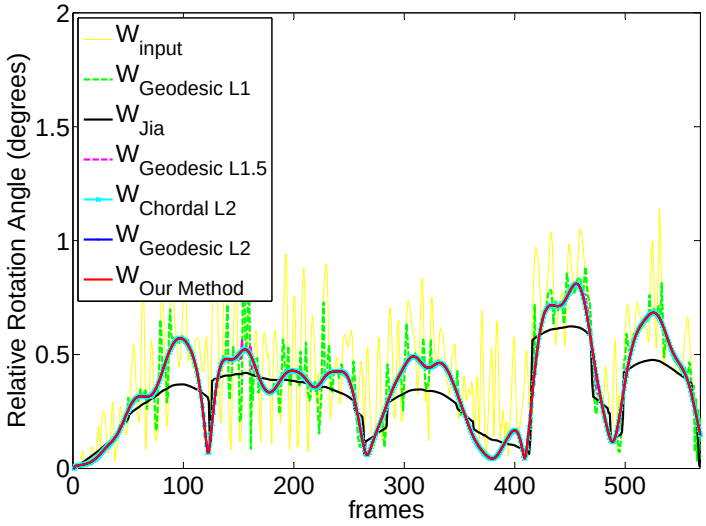

Figure 6.14: Relative Rotational Angle of Consecutive Orientations in Our Pairwise Smoothing Result (Red) VS Jia and Evans's Smoothing Result (Black), and Other Window-based Smoothing Methods (Standing Sequence)

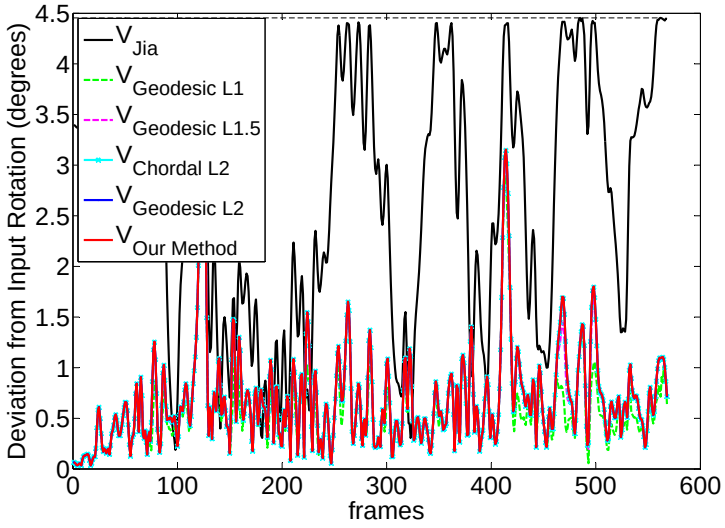

Figure 6.15: Rotational Angle Deviation from Input in Our Pairwise Smoothing Result (Red), Jia and Evans's Smoothing Result(Black), and Other Window-based Smoothing Methods (Standing Sequence)

Table 6.2: Comparisons of Our Pairwise Method, Jia and Evan's, Geodesic L_2, Geodesic L_1, Geodesic $L_{1.5}$ Mean, and Chordal L_2 Mean on the standing video sequence. The Geodesic L_2 Distance is the Square of Relative Rotational Angle between Consecutive Frames. Numbers in Parenthesis is the Computation Time in C++ Implementation, Non-Parenthesised are MATLAB Implementation

	Geodesic L_2 Distance Sum		For 568 Frames
	Relative Rotation	Deviation from Input	Comp. Time (s)
Input	159.87	0	-
Pairwise Method	98.43	419.17	4.87 (**0.78**)
Jia and Evans's	**72.51**	3783.30	28.96
Geodesic L_2	98.47	417.54	4.15 (1.46)
Geodesic L_1	104.01	319.36	20.87
Geodesic $L_{1.5}$	98.77	407.17	4.56
Chordal L_2	98.49	416.74	1.63 (**0.01**)

than the other window-based methods.

The resulting video from using Jia and Evan's method also look very similar to the one obtained using the window-based method. However, Jia and Evan's method has larger black border intrusion for this video sequence.

6.5 Summary

This chapter discusses a new method to smooth a noisy input in the Special Euclidean Group, $SE(3)$. The translational part of $SE(3)$ is smoothed by simple vector convolution with a Gaussian kernel, and an analogous method to smooth input rotation in the Special Orthogonal Group, $SO(3)$ is proposed.

It is shown that the pairwise average method (Section 6.3.1) is superior to the method presented by Jia and Evans [2014] in rotation smoothing. The pairwise averaging method presented is shown to closely approximate the weighted L_2 mean method (Section 6.3.2), while being approximately 1.8 times faster in computation speed. The chapter also presents an alternative method that has a much shorter computation time, called Weighted Chordal L_2 Mean (Section 6.3.4). Table 6.1 and 6.2 summarises the experimental results between the different rotation smoothing methods.

Additionally, it was also showed that the pairwise method successfully minimises both the relative orientation angle in consecutive frames (smoothness metric), while maintaining small deviation from the input rotation sequence. This is a trade-off controlled by the Gaussian kernel's standard deviation, σ, similar to the scalar factor, α in equation (6.3) presented in [Jia and Evans, 2014].

Due to the use of a Gaussian convolution, the method does not introduce drift and scale changes (sum of weights = 1), and is temporally invariant. The pairwise

computation is also parallelisable, fast ($\approx 1.4ms$/data point), and has a deterministic computation time. Unlike iterative methods, it also does not require any initial estimate, which has been shown to sometimes affect the convergence of the iterative algorithm.

The only drawback in the proposed Pairwise Rotations Averaging method is the delay introduced by the technique, which is half the window length used in the convolution. However, if there is extra information about the "future" motion available, the solution is ensured to stay close to the mean of the actual trajectory, as shown in the quaternion plot (Figure 6.10). Finally, the video stabilisation output of the walking sequence is illustrated in features trajectory shown in Figure 6.12, showing improved video stabilization performance.

Chapter 7

Conclusions and future work

This thesis covers the development of an efficient approximate Bayesian filter and its application to real-world complex systems. The main conclusions and future work are summarized in this chapter. The introductory chapters (1 and 2) are excluded from this discussion because they do not contain new research material.

7.1 Conclusions

Chapter 3 describes the design of a new approximate Bayesian filter (MIGE) that exploits the geometrical aspect of a measurement likelihood. The method is built upon the use of a degenerate Gaussian function to approximate a nonlinear likelihood function arising in various sensing problems in target-tracking and localization tasks. A degenerate Gaussian function allows infinite uncertainty along some directions representing cylindrical or planar likelihood functions which are used to approximate the cone-shape density in bearing sensing or shell-shape density in range sensing. The standard Gaussian parameters of mean and covariance are ill-defined in these functions, and thus we apply a new parametrization method consisting of a minimal set of coefficients for the quadratic terms. The performance of MIGE is evaluated using Monte Carlo simulations for bearing-only and range-only target localizations. Results show improved performance compared to the state-of-the-art nonlinear filters in terms of accuracy, consistency, and computational/memory requirement.

Chapter 4 describes a new recursive localization method that uses passive time-difference-of-arrival (TDOA) and frequency-different-of-arrival (FDOA) measurements. The recursive method is based on MIGE, and localizes the position of an unknown stationary target using TDOA and FDOA measurements received by pairs of synchronized and localized radio sensors. This method is designed to handle different challenging scenarios, which include the presence of noise, missing measurements and outliers. The method updates the location estimate with each new measurement and approximates the underlying measurement likelihood with a degenerate Gaussian. The position estimation accuracy is tested using Monte Carlo simulations. The method is also evaluated using experimental tests on real measurement data collected from software defined radios (SDRs), which shows improved localization

accuracy when compared to existing methods.

Chapter 5 describes a novel monocular visual SLAM method that is suitable for any camera undergoing general $SE(3)$ motion. This method has potential to be applied to many practical areas where the camera location and dense reconstruction of the scene are required. The method uses dense optical flow and estimates the corresponding uncertainties, which are then used as input to our new Mahalanobis eight-point algorithm. Based on MIGE, an efficient 3D point triangulation method is used to produce accurate estimate of the location and uncertainty. The current implementation of our visual SLAM does not compute the uncertainty of the estimated pose in manner similar to Extended Kalman filter. This is because estimating pose uncertainty by subjecting dense pixel correspondences to EKF filtering means we have to operate on very large covariance and Jacobian matrices. This incurs high computation and memory cost. Thus, in the inter-frame pose fusion, we use a simple average for both rotation and translation. In the triangulation step, we assume that the error in rotation and translation is negligible compared to the error of the image feature position. The proposed visual SLAM method is successfully applied to video captured from camera mounted on aerial and ground vehicles. The experiments show improved localization and mapping performance, effectively handling low or repetitive textured scenes, purely rotational motions and drastic camera height variations.

Chapter 6 describes an efficient window-based weighted average for $SE(3)$ path smoothing. A pairwise weighted averaging tree is designed to efficiently perform weighted average of a large number of variables by leveraging on the parallelisable layers. The proposed translational smoothing is mathematically equivalent to the well-known vector convolution. The pairwise weighted averaging tree is applied to rotation smoothing, where it is experimentally shown to be close to the weighted L_2 mean method, while being 1.8 times faster in computational speed. The pairwise rotation averaging method is also shown to be able to successfully minimise both the relative orientation angle between consecutive frames (smoothness metric), while remaining close to the input rotations. The method also does not introduce drift or scale changes. Unlike iterative methods, it does not require any initial estimate, and has deterministic computational time. Like all window-based smoothing methods, the proposed method introduces a delay equivalent to half the window size. However, the inclusion of the "future" motion allows the solution to remain close to the mean of the actual trajectory. The rotation smoothing method is applied to video stabilization task and shows a reduction in undesirable camera shake while introducing minimal black border intrusion.

7.2 Future work

The current derivation of the approximate Bayesian filter called minimal iterative Gaussian estimator (MIGE) in Chapter 3 only covers the measurement likelihood up to the three-dimensional case. Future research can be conducted to extend the esti-

mator to solve higher dimensional problems. Despite the fact that the geometrical interpretation is not as intuitive in a higher dimension, one can still define a degenerate Gaussian likelihood with the corresponding transformation matrix in the Special Orthogonal group $SO(n)$ manifold. Applying the same logic, one can also derive the minimal parametrization method for high dimensional cases.

Multiple extensions of the TDOA-FDOA localization work discussed in Chapter 4 are possible. First, the current two-dimensional localization method can be extended to the three-dimensional cases, where the need for the prior knowledge of emitter height is removed. This can be accomplished by using the planar degenerate Gaussian likelihood to approximate the hyperboloid (TDOA) and conical (FDOA) measurement likelihood. Second, the current localization method assumes that the emitter is stationary. In a future work, the localization method can be extended to track mobile emitter using TDOA-FDOA measurements. This can be accomplished by incorporating a motion model and state prediction step into our method, similar to EKF. Third, the current method assumes that the sensors' locations are known accurately, but this is only an approximation. In the future, we will perform an analysis on the effects of sensor location error on the localization result.

The monocular visual SLAM method discussed in Chapter 5 uses dense optical flow to obtain a robust estimate of the inter-frame camera pose. A simplifying assumption is used, such that the estimated camera pose is assumed to have no uncertainty. This assumption circumvents the need to estimate the pose uncertainty, which is computationally expensive with the currently available methods. For example, the extended Kalman filter (EKF) is not suitable for dense correspondences, which involves the computation of very large covariance and Jacobian matrices. In the future, efforts can be directed to investigate a more efficient method for estimating the uncertainty of the pose, which will allow a weighted fusion of the pose estimated based on their corresponding uncertainty. The estimated uncertainty of the camera pose also allows the propagation of error to the triangulated 3D scene points.

Another possible extension of the monocular visual SLAM work is to include a smoothing step in the (almost) densely reconstructed 3D scene points (or depth map). This allows missing or more uncertain points of the image to get an improvement in terms of accuracy and reduction in uncertainty by using information from the surrounding pixels that are more accurately localized. The justification of such techniques is similar to the spatial smoothness terms commonly used in optical flow method, where close-by pixels are found to have similar motion and depth.

Improvement can also be made to the current back-end of our monocular visual SLAM that uses the structural similarity index (SSIM) to identify images of the same scene. The SSIM works well for ground-based vehicles, where their motion is highly constrained. However, for an unmanned aerial vehicle (UAV), the high freedom of motion causes some images of the same scene to have low SSIM value. Thus, a different method is required to more robustly identify images of the same scene. The method needs to be robust enough to work even for highly unstructured environments and drastic changes in viewpoint, similar to the challenging

scenarios observed from the UAV videos in our experiments. Methods using bag of words [Gálvez-López and Tardos, 2012; Kejriwal et al., 2016] may be used for this purpose.

In the video stabilization work presented in Chapter 6, we use camera pose estimated from an external inertial measurement unit (IMU). However, most cameras do not have the capability to easily obtain synchronized images with an IMU. A poor synchronisation can lead to poor video stabilization results. Thus, a possible extension is to apply our proposed monocular visual SLAM method (in Chapter 5), where the camera pose estimate can better reflect the observed motion, without requiring additional IMU hardware. Our monocular visual SLAM method also provides an accurate estimation of the (almost) dense depth map. This allows the use of spatially varying warp to synthesis novel views for video stabilization suitable for removing translational vibrations.

Bibliography

Achtelik, M.; Bachrach, A.; He, R.; Prentice, S.; and Roy, N., 2009. Stereo vision and laser odometry for autonomous helicopters in gps-denied indoor environments. In *Unmanned Systems Technology XI*, vol. 7332, 733219. International Society for Optics and Photonics. (cited on page 74)

Aftab, K.; Hartley, R.; and Trumpf, J., 2015. Generalized weiszfeld algorithms for lq optimization. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 37, 4 (April 2015), 728–745. doi:10.1109/TPAMI.2014.2353625. (cited on pages 107 and 108)

Alspach, D. and Sorenson, H., 1972. Nonlinear bayesian estimation using gaussian sum approximations. *IEEE Transactions on Automatic Control*, 17, 4 (Aug 1972), 439–448. doi:10.1109/TAC.1972.1100034. (cited on pages 2, 16, and 40)

Anderson, B. D. O. and Moore, J. B., 1979. *Optimal filtering*. Prentice-Hall. (cited on pages 39, 40, and 47)

Arasaratnam, I. and Haykin, S., 2009. Cubature kalman filters. *IEEE Transactions on automatic control*, 54, 6 (2009), 1254–1269. (cited on page 40)

Artieda, J.; Sebastian, J. M.; Campoy, P.; Correa, J. F.; Mondragón, I. F.; Martínez, C.; and Olivares, M., 2009. Visual 3-d slam from uavs. *Journal of Intelligent and Robotic Systems*, 55, 4-5 (2009), 299. (cited on page 75)

Arulampalam, M. S.; Maskell, S.; Gordon, N.; and Clapp, T., 2002. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50, 2 (2002), 174–188. (cited on pages 2, 16, 17, and 50)

Bachrach, A.; He, R.; and Roy, N., 2009. Autonomous flight in unknown indoor environments. *International Journal of Micro Air Vehicles*, 1, 4 (2009), 217–228. (cited on page 74)

Bao, F.; Cao, Y.; Webster, C.; and Zhang, G., 2014. A hybrid sparse-grid approach for nonlinear filtering problems based on adaptive-domain of the zakai equation approximations. *Salud Colectiva*, 2, 1 (2014), 784–804. (cited on page 16)

Bar-Shalom, Y.; Li, X. R.; and Kirubarajan, T., 2004. *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons. (cited on pages 19, 50, and 54)

Bay, H.; Tuytelaars, T.; and Van Gool, L., 2006. Surf: Speeded up robust features. In *European conference on computer vision*, 404–417. Springer. (cited on page 27)

Bayes, T. and Price, R., 1763. An essay towards solving a problem in the doctrine of chances. by the late rev. mr. bayes, frs communicated by mr. price, in a letter to john canton, amfrs. *Philosophical Transactions (1683-1775)*, (1763), 370–418. (cited on pages 1 and 10)

Bell, B. M. and Cathey, F. W., 1993. The iterated kalman filter update as a gauss-newton method. *Automatic Control IEEE Transactions on*, 38, 2 (1993), 294–297. (cited on page 48)

Bernardo, J. M. and Smith, A. F. M., 1994. Bayesian theory. *Journal of the Royal Statistical Society*, 15, 19 (1994), 13–23. (cited on page 10)

Black, M. J. and Anandan, P., 1991. Robust dynamic motion estimation over time. In *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR'91., IEEE Computer Society Conference on*, 296–302. IEEE. (cited on page 29)

Bradler, H.; Anne Wiegand, B.; and Mester, R., 2015. The statistics of driving sequences – and what we can learn from them. In *The IEEE International Conference on Computer Vision (ICCV) Workshops*. (cited on page 75)

Brox, T.; Bruhn, A.; Papenberg, N.; and Weickert, J., 2004. High accuracy optical flow estimation based on a theory for warping. *Computer Vision-ECCV 2004*, (2004), 25–36. (cited on page 28)

Caballero, F.; Merino, L.; Ferruz, J.; and Ollero, A., 2009. Vision-based odometry and slam for medium and high altitude flying uavs. *Journal of Intelligent and Robotic Systems*, 54, 1-3 (2009), 137–161. (cited on page 74)

Cai, Z.; Gland, F. L.; and Zhang, H., 1995. An adaptive local grid refinement method for nonlinear filtering. (cited on page 16)

Carter, G., 1981. Time delay estimation for passive sonar signal processing. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 29, 3 (1981), 463–470. (cited on page 55)

Carter, G. C., 1987. Coherence and time delay estimation. *Proceedings of the IEEE*, 75, 2 (1987), 236–255. (cited on page 55)

Chan, Y. T. and Ho, K. C., 1994. A simple and efficient estimator for hyperbolic location. *IEEE Transactions on Signal Processing*, 42, 8 (Aug 1994), 1905–1915. doi:10.1109/78.301830. (cited on page 56)

Chatterjee, A. and Govindu, V., 2013. Efficient and robust large-scale rotation averaging. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, 521–528. doi:10.1109/ICCV.2013.70. (cited on page 108)

Chen, Q. and Koltun, V., 2016. Full flow: Optical flow estimation by global optimization over regular grids. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4706–4714. (cited on pages 75 and 82)

Chen, Z., 2003. Bayesian filtering: From kalman filters to particle filters, and beyond. *Statistics*, (2003). (cited on pages 10, 18, 40, and 41)

Cheng, J.; Kim, J.; Shao, J.; and Zhang, W., 2015. Robust linear pose graph-based slam. *Robotics and Autonomous Systems*, 72 (2015), 71–82. (cited on pages 7, 73, and 95)

Cheviron, T.; Hamel, T.; Mahony, R.; and Baldwin, G., 2007. Robust nonlinear fusion of inertial and visual data for position, velocity and attitude estimation of uav. In *Robotics and Automation, 2007 IEEE International Conference on*, 2010–2016. IEEE. (cited on page 74)

Chin, W. H.; Ward, D. B.; and Constantinides, A. G., 2002. Semi-blind mimo channel tracking using auxiliary particle filtering. In *Global Telecommunications Conference, 2002. GLOBECOM '02. IEEE*, 322–325 vol.1. (cited on page 10)

Choi, K. H.; Ra, W. S.; Park, J. B.; and Yoon, T. S., 2013. Compensated robust least-squares estimator for target localisation in sensor network using time difference of arrival measurements. *IET Signal Processing*, 7, 8 (October 2013), 664–673. doi:10.1049/iet-spr.2012.0374. (cited on pages 56, 57, 65, 66, and 67)

Chowdhary, G.; Johnson, E. N.; Magree, D.; Wu, A.; and Shein, A., 2013. Gps-denied indoor and outdoor monocular vision aided navigation and control of unmanned aircraft. *Journal of Field Robotics*, 30, 3 (2013), 415–438. (cited on page 74)

Civera, J.; Davison, A. J.; and Montiel, J. M., 2008. Inverse depth parametrization for monocular slam. *IEEE transactions on robotics*, 24, 5 (2008), 932–945. (cited on page 39)

Demonceaux, C.; Vasseur, P.; and Pegard, C., 2006. Omnidirectional vision on uav for attitude computation. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, 2842–2847. doi:10.1109/ROBOT.2006.1642132. (cited on page 74)

Deng, G. and Cahill, L. W., 1993. An adaptive gaussian filter for noise reduction and edge detection. In *1993 IEEE Conference Record Nuclear Science Symposium and Medical Imaging Conference*, 1615–1619 vol.3. doi:10.1109/NSSMIC.1993.373563. (cited on page 106)

Dollár, P. and Zitnick, C. L., 2015. Fast edge detection using structured forests. *IEEE transactions on pattern analysis and machine intelligence*, 37, 8 (2015), 1558–1570. (cited on page 75)

Douc, R. and Cappe, O., 2005. Comparison of resampling schemes for particle filtering. In *ISPA 2005. Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis, 2005.*, 64–69. doi:10.1109/ISPA.2005.195385. (cited on page 89)

Doucet, A.; De Freitas, N.; and Gordon, N., 2001. *Sequential Monte Carlo methods in practice*. Springer. (cited on page 41)

Drory, A.; Haubold, C.; Avidan, S.; and Hamprecht, F. A., 2014. Semi-global matching: A principled derivation in terms of message passing. In *GCPR. Proceedings*, 8753, 43–53. doi:10.1007/978-3-319-11752-2_4. 1. (cited on page 75)

Efron, B. and Tibshirani, R., 1993. An introduction to the bootstrap. *Monographs on Statistics and Applied Probability, Chapman and Hall, London*, , 57 (1993), 436. (cited on page 17)

Ertürk, S., 2002. Real-time digital image stabilization using kalman filters. *Real-Time Imaging*, 8, 4 (2002), 317–328. (cited on page 108)

Everson, R., 1997. Orthogonal, but not orthonormal, procrustes problems. In *Advances in Computational Mathematics . (Submitted). Available from http://www.ee.ic.ac.uk/research/neural/everson*. (cited on page 107)

Fanani, N.; Stürck, A.; Ochs, M.; Bradler, H.; and Mester, R., 2017. Predictive monocular odometry (pmo): What is possible without ransac and multiframe bundle adjustment? *Image and Vision Computing*, (2017). (cited on pages 75, 97, and 98)

Fathy, M. E.; Hussein, A. S.; and Tolba, M. F., 2011. Fundamental matrix estimation: A study of error criteria. *Pattern Recognition Letters*, 32, 2 (2011), 383–391. (cited on page 34)

Fischler, M. A. and Bolles, R. C., 1981. *Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography*. ACM. (cited on page 29)

Fletcher, F.; Ristic, B.; and Musicki, D., 2007. Recursive estimation of emitter location using tdoa measurements from two uavs. In *2007 10th International Conference on Information Fusion*, 1–8. IEEE. (cited on page 56)

Fowler, M. L. and Hu, X., 2008. Signal models for tdoa/fdoa estimation. *IEEE Transactions on Aerospace and Electronic Systems*, 44, 4 (2008), 1543–1550. (cited on page 56)

Fredriksson, J. and Olsson, C., 2013. Simultaneous multiple rotation averaging using lagrangian duality. In *Computer Vision âĂŞ ACCV 2012* (Eds. K. Lee; Y. Matsushita; J. Rehg; and Z. Hu), vol. 7726 of *Lecture Notes in Computer Science*, 245–258. Springer Berlin Heidelberg. ISBN 978-3-642-37430-2. doi:10.1007/978-3-642-37431-9_19. (cited on page 108)

Gálvez-López, D. and Tardos, J. D., 2012. Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics*, 28, 5 (2012), 1188–1197. (cited on page 130)

Gao, X.-S.; Hou, X.-R.; Tang, J.; and Cheng, H.-F., 2003. Complete solution classification for the perspective-three-point problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25, 8 (Aug 2003), 930–943. doi:10.1109/TPAMI.2003.1217599. (cited on page 92)

Geiger, A.; Lenz, P.; and Urtasun, R., 2012. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages xviii, 28, 49, 52, 96, and 97)

Geiger, A.; Ziegler, J.; and Stiller, C., 2011. Stereoscan: Dense 3d reconstruction in real-time. In *Intelligent Vehicles Symposium (IV)*. (cited on pages 97, 98, and 99)

Girshick, R.; Donahue, J.; Darrell, T.; and Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 580–587. (cited on page 27)

Glocker, B.; Paragios, N.; Komodakis, N.; Tziritas, G.; and Navab, N., 2008. Optical flow estimation with uncertainties through dynamic mrfs. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, 1–8. IEEE. (cited on page 75)

Glover, J. and Kaelbling, L. P., 2013. Tracking 3-d rotations with the quaternion bingham filter. (2013). (cited on page 108)

Gordon, N. J.; Salmond, D. J.; and Smith, A. F. M., 2002. Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE Proceedings F - Radar and Signal Processing*, 140, 2 (2002), 107–113. (cited on pages 2 and 41)

Gustafsson, F., 2010. Particle filter theory and practice with positioning applications. *IEEE Aerospace and Electronic Systems Magazine*, 25, 7 (2010), 53–82. (cited on page 17)

Gut, A., 2009. An intermediate course in probability. *Springer Texts in Statistics*, (2009), xvi+303. (cited on page 11)

Han, T.; Lu, X.; and Lan, Q., 2010. Pattern recognition based kalman filter for indoor localization using tdoa algorithm. *Applied Mathematical Modelling*, 34, 10 (2010), 2893–2900. (cited on page 56)

Harris, C. and Stephens, M., 1988. A combined corner and edge detector. In *Alvey vision conference*, vol. 15, 10–5244. Citeseer. (cited on page 27)

Hartley, R.; Aftab, K.; and Trumpf, J., 2011. L1 rotation averaging using the weiszfeld algorithm. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 0 (2011), 3041–3048. doi:http://doi.ieeecomputersociety.org/10.1109/CVPR.2011.5995745. (cited on pages 106 and 108)

Hartley, R.; Trumpf, J.; Dai, Y.; and Li, H., 2012. Rotation averaging. doi:10.1007/s11263-012-0601-0. (cited on pages 106, 107, and 112)

Hartley, R. and Zisserman, A., 2003. *Multiple view geometry in computer vision*. Cambridge university press. (cited on pages 22, 23, 26, 31, 32, 33, 34, 35, 75, and 88)

Hartley, R. I., 1997. In defense of the eight-point algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19, 6 (Jun 1997), 580–593. doi:10.1109/34.601246. (cited on pages 33 and 83)

Heeger, D. J., 1988. Optical flow using spatiotemporal filters. *International journal of computer vision*, 1, 4 (1988), 279–302. (cited on page 75)

Heng, L.; Lee, G. H.; Fraundorfer, F.; and Pollefeys, M., 2011. Real-time photo-realistic 3d mapping for micro aerial vehicles. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 4012–4019. doi:10.1109/IROS.2011.6095058. (cited on page 74)

Herisse, B.; Russotto, F. X.; Hamel, T.; and Mahony, R., 2008. Hovering flight and vertical landing control of a vtol unmanned aerial vehicle using optical flow. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 801–806. doi:10.1109/IROS.2008.4650731. (cited on page 74)

Ho, Y. and Lee, R., 1964. A bayesian approach to problems in stochastic estimation and control. *IEEE Transactions on Automatic Control*, 9, 4 (Oct 1964), 333–339. doi:10.1109/TAC.1964.1105763. (cited on pages 1, 9, and 10)

Horn, B. K. and Schunck, B. G., 1981. Determining optical flow. *Artificial intelligence*, 17, 1-3 (1981), 185–203. (cited on page 28)

Hrabar, S. and Sukhatme, G. S., 2003. Omnidirectional vision for an autonomous helicopter. In *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, vol. 1, 558–563 vol.1. doi:10.1109/ROBOT.2003.1241653. (cited on page 74)

Hrabar, S.; Sukhatme, G. S.; Corke, P.; Usher, K.; and Roberts, J., 2005. Combined optic-flow and stereo-based navigation of urban canyons for a uav. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 3309–3316. doi:10.1109/IROS.2005.1544998. (cited on page 74)

Huang, G. P.; Mourikis, A. I.; and Roumeliotis, S. I., 2010. Observability-based rules for designing consistent ekf slam estimators. *International Journal of Robotics Research*, 29, 5 (2010), 502–528. (cited on pages 2 and 40)

Huang, S. and Dissanayake, G., 2007. Convergence and consistency analysis for extended kalman filter based slam. *IEEE Transactions on robotics*, 23, 5 (2007), 1036–1049. (cited on page 39)

Huang, S. J., 1997. Adaptive noise reduction and image sharpening for digital video compression. In *Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on*, vol. 4, 3142–3147. IEEE. (cited on page 106)

Hui, T.-W.; Tang, X.; and Loy, C. C., 2018. Liteflownet: A lightweight convolutional neural network for optical flow estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 8981–8989. (cited on page 75)

Ibragimov, I. A. and Has'Minskii, R. Z., 2013. *Statistical estimation: asymptotic theory*, vol. 16. Springer Science & Business Media. (cited on page 19)

Ito, K. and Xiong, K., 2000. Gaussian filters for nonlinear filtering problems. *IEEE transactions on automatic control*, 45, 5 (2000), 910–927. (cited on page 40)

Jacob, B. and Guennebaud, G., 2015. Eigen @ONLINE. Accessed: 2015-04-21. (cited on page 117)

Jia, C. and Evans, B., 2013. 3d rotational video stabilization using manifold optimization. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, 2493–2497. doi:10.1109/ICASSP.2013.6638104. (cited on page 108)

Jia, C. and Evans, B., 2014. Constrained 3d rotation smoothing via global manifold regression for video stabilization. *Signal Processing, IEEE Transactions on*, 62, 13 (July 2014), 3293–3304. doi:10.1109/TSP.2014.2325795. (cited on pages xxiii, 105, 107, 115, 117, 118, 119, 120, 121, and 125)

Jia, Y.; Shelhamer, E.; Donahue, J.; Karayev, S.; Long, J.; Girshick, R.; Guadarrama, S.; and Darrell, T., 2014. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, 675–678. ACM. (cited on page 27)

Julier, S. J. and Uhlmann, J. K., 2004. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92, 3 (2004), 401–422. (cited on pages 13, 14, and 56)

Kailath, T., 1970. The innovations approach to detection and estimation theory. *Proceedings of the IEEE*, 58, 5 (1970), 680–695. (cited on page 13)

Kalman, R. E., 1960. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering Transactions*, 82 (1960), 35–45. (cited on pages 1, 12, and 56)

Kalman, R. E., 1963. New methods in wiener filtering theory. *In proceedings of the first symposium on Engineering Application of Random Function Theory and Probability (J.L. Bogdanoff and*, (1963). (cited on page 12)

Kalman, R. E. and Bucy, R. S., 1961. New results in linear filtering and prediction theory. *Journal of basic engineering*, 83, 1 (1961), 95–108. (cited on page 1)

Kanatani, K.; Sugaya, Y.; and Niitsuma, H., 2008. Triangulation from two views revisited: Hartley-sturm vs. optimal correction. *In practice*, 4 (2008), 5. (cited on page 34)

Kejriwal, N.; Kumar, S.; and Shibata, T., 2016. High performance loop closure detection using bag of word pairs. *Robotics and Autonomous Systems*, 77 (2016), 55–65. (cited on page 130)

Kitagawa, G., 1996. Monte carlo filter and smoother for non-gaussian nonlinear state space models. *Journal of Computational & Graphical Statistics*, 5, 1 (1996), 1–25. (cited on page 17)

Klein, G. and Murray, D., 2007. Parallel tracking and mapping for small ar workspaces. In *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, 225–234. IEEE. (cited on page 75)

Klose, S.; Wang, J.; Achtelik, M.; Panin, G.; Holzapfel, F.; and Knoll, A., 2010. Markerless, vision-assisted flight control of a quadrocopter. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, 5712–5717. IEEE. (cited on page 74)

Kolmogorov, A. N., 1941. Stationary sequences in hilbert space. *Bull. Math. Univ. Moscow*, 2, 6 (1941), 1–40. (cited on page 1)

Kolmogorov, A. N.; Doyle, W. L.; and Selin, I., 1962. Interpolation and extrapolation of stationary random sequences. (1962). (cited on page 13)

Kotaru, M.; Joshi, K.; Bharadia, D.; and Katti, S., 2015. Spotfi: Decimeter level localization using wifi. In *ACM SIGCOMM Computer Communication Review*, vol. 45, 269–282. ACM. (cited on pages 52 and 53)

Kramer, S. C. and Sorenson, H. W., 1988. Recursive bayesian estimation using piece-wise constant approximations. *Automatica*, 24, 6 (1988), 789–801. (cited on page 16)

Kurz, G.; Gilitschenski, I.; Julier, S.; and Hanebeck, U. D., 2013. Recursive estimation of orientation based on the bingham distribution. *CoRR*, abs/1304.8019 (2013). http://arxiv.org/abs/1304.8019. (cited on page 110)

Kybic, J. and Nieuwenhuis, C., 2011. Bootstrap optical flow confidence and uncertainty measure. *Computer Vision and Image Understanding*, 115, 10 (2011), 1449–1462. (cited on page 75)

Lanzisera, S.; Zats, D.; and Pister, K. S. J., 2011. Radio frequency time-of-flight distance measurement for low-cost wireless sensor localization. *IEEE Sensors Journal*, 11, 3 (March 2011), 837–845. doi:10.1109/JSEN.2010.2072496. (cited on page 52)

Lee, G. H.; Fraundorfer, F.; and Pollefeys, M., 2011. Mav visual slam with plane constraint. In *2011 IEEE International Conference on Robotics and Automation*, 3139–3144. doi:10.1109/ICRA.2011.5980365. (cited on page 74)

Lefebure, M.; Alvarez, L.; Esclarin, J.; and Sánchez, J., 1999. A pde model for computing the optical flow. In *Proc. XVI congreso de ecuaciones diferenciales y aplicaciones*, 1349–1356. (cited on page 28)

Li, H. and Hartley, R., 2006. Five-point motion estimation made easy. In *18th International Conference on Pattern Recognition (ICPR'06)*, vol. 1, 630–633. doi:10.1109/ICPR.2006.579. (cited on pages 32 and 88)

Li, M. and Mourikis, A. I., 2013. High-precision, consistent ekf-based visualâĂŞinertial odometry. *International Journal of Robotics Research*, 32, 6 (2013), 690–711. (cited on pages 2 and 40)

Li, S.; Hedley, M.; Collings, I. B.; and Humphrey, D., 2015. Tdoa-based localization for semi-static targets in nlos environments. *IEEE Wireless Communications Letters*, 4, 5 (2015), 513–516. (cited on page 56)

Lin, T. T. and Yau, S. S., 1967. Bayesian approach to the optimization of adaptive systems. 3, 2 (1967), 77–85. (cited on page 10)

Liu, C.; Yuen, J.; and Torralba, A., 2011. Sift flow: Dense correspondence across scenes and its applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33, 5 (May 2011), 978–994. doi:10.1109/TPAMI.2010.147. (cited on page 28)

Liu, J. and RongChen, 1998. Sequential monte carlo methods for dynamic systems. *Publications of the American Statistical Association*, 93, 443 (1998), 1032–1044. (cited on page 17)

Longuet-Higgins, H. C., 1981. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293, 5828 (1981), 133. (cited on page 36)

Lowe, D. G., 2004. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60, 2 (2004), 91–110. doi:10.1023/B:VISI.0000029664.99615.94. http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94. (cited on page 27)

Lucas, B. D.; Kanade, T.; et al., 1981. An iterative image registration technique with an application to stereo vision. (1981). (cited on page 28)

Mac Aodha, O.; Humayun, A.; Pollefeys, M.; and Brostow, G. J., 2013. Learning a confidence measure for optical flow. *IEEE transactions on pattern analysis and machine intelligence*, 35, 5 (2013), 1107–1120. (cited on page 75)

Mahalanobis, P. C., 1936. On the generalized distance in statistics. *Proceedings of the National Institute of Sciences*, 2 (1936), 49–55. (cited on page 85)

Maz'ya, V. and Schmidt, G., 1996. On approximate approximations using gaussian kernels. *IMA Journal of Numerical Analysis*, 16, 1 (1996), 13–29. (cited on page 40)

Meinhold, R. J. and Singpurwalla, N. D., 1983. Understanding the kalman filter. *The American Statistician*, 37, 2 (1983), 123–127. (cited on page 1)

Menze, M. and Geiger, A., 2015. Object scene flow for autonomous vehicles. In *Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 97)

Menze, M.; Heipke, C.; and Geiger, A., 2018. Object scene flow. *ISPRS Journal of Photogrammetry and Remote Sensing (JPRS)*, (2018). (cited on page 49)

Metropolis, N.; Rosenbluth, A. W.; Rosenbluth, M. N.; Teller, A. H.; and Teller, E., 2004. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21, 6 (2004), 1087–1092. (cited on page 41)

Metropolis, N. and Ulam, S., 1949. The monte carlo method. *Astrophysics and Space Science*, 44, 247 (1949), 335–341. (cited on page 41)

Miao, F.; Yang, D.; Wang, R.; Wen, J.; Wang, Z.; and Lian, X., 2014. A moving sound source localization method based on tdoa. In *INTER-NOISE and NOISE-CON Congress and Conference Proceedings*, vol. 249, 4159–4165. Institute of Noise Control Engineering. (cited on page 56)

Moakher, M., 2002. Means and averaging in the group of rotations. *SIAM Journal on Matrix Analysis and Applications*, 24, 1 (2002), 1–16. doi:10.1137/S0895479801383877. (cited on pages 92, 106, 107, and 110)

Montiel, J. M.; Civera, J.; and Davison, A. J., 2006. Unified inverse depth parametrization for monocular slam. Robotics: Science and Systems. (cited on page 39)

Musicki, D.; Kaune, R.; and Koch, W., 2010. Mobile emitter geolocation and tracking using tdoa and fdoa measurements. *IEEE Transactions on Signal Processing*, 58, 3 (2010), 1863–1874. (cited on pages 56 and 57)

Nagel, H.-H. and Enkelmann, W., 1986. An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, , 5 (1986), 565–593. (cited on page 28)

Ng, Y.; Kim, J.; and Li, H., 2017. Robust dense optical flow with uncertainty for monocular pose-graph slam. In *Australasian Conference on Robotics and Automation*. (cited on pages 97 and 98)

Nister, D., 2004. An efficient solution to the five-point relative pose problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26, 6 (June 2004), 756–770. doi:10.1109/TPAMI.2004.17. (cited on pages 32 and 88)

Oliensis, J., 2005. The least-squares error for structure from infinitesimal motion. *International Journal of Computer Vision*, 61, 3 (2005), 259–299. (cited on page 74)

Park, S.; Won, D.; Kang, M.; Kim, T.; Lee, H.; and Kwon, S., 2005. Ric (robust internal-loop compensator) based flight control of a quad-rotor type uav. In *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*, 3542–3547. IEEE. (cited on page 74)

Piao, D.; Menon, P. G.; and Mengshoel, O. J., 2014. Computing probabilistic optical flow using markov random fields. In *International Symposium Computational Modeling of Objects Represented in Images*, 241–247. Springer. (cited on page 76)

Press, S. J., 2003. Subjective and objective bayesian statistics. *Publications of the American Statistical Association*, 100, 469 (2003), 355–355. (cited on page 10)

Raguram, R.; Frahm, J. M.; and Pollefeys, M., 2009. Exploiting uncertainty in random sample consensus. In *2009 IEEE 12th International Conference on Computer Vision*, 2074–2081. doi:10.1109/ICCV.2009.5459456. (cited on page 88)

Rappaport, T. S.; Reed, J. H.; and Woerner, B. D., 1996. Position location using wireless communications on highways of the future. *IEEE communications Magazine*, 34, 10 (1996), 33–41. (cited on page 57)

Rea, M.; Fakhreddine, A.; Giustiniano, D.; and Lenders, V., 2017. Filtering noisy 802.11 time-of-flight ranging measurements from commoditized wifi radios. *IEEE/ACM Transactions on Networking*, 25, 4 (Aug 2017), 2514–2527. doi:10.1109/TNET.2017.2700430. (cited on pages 52 and 53)

Revaud, J.; Weinzaepfel, P.; Harchaoui, Z.; and Schmid, C., 2015. Epicflow: Edge-preserving interpolation of correspondences for optical flow. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1164–1172. (cited on pages 75, 86, and 97)

Robert, C. P., 1994. The bayesian choice: a decision-theoretic motivation. *Journal of the American Statistical Association*, 91, 433 (1994). (cited on page 10)

Robinson, A., 2010. Randomization, bootstrap and monte carlo methods in biology. *Journal of the Royal Statistical Society*, 170, 3 (2010), 856–856. (cited on page 41)

Rubin, D. B., 1987. *Multiple Imputation for Nonresponse in Surveys*. Wiley. (cited on page 17)

Sampson, P. D., 1982. Fitting conic sections to âĂIJvery scatteredâĂİ data: An iterative refinement of the bookstein algorithm. *Computer Graphics and Image Processing*, 18, 1 (1982), 97 – 108. doi:https://doi.org/10.1016/0146-664X(82)90101-0. http://www.sciencedirect.com/science/article/pii/0146664X82901010. (cited on page 33)

Sarlette, A. and Sepulchre, R., 2009. Consensus optimization on manifolds. *SIAM Journal on Control and Optimization*, 48, 1 (2009), 56–76. (cited on page 107)

Schaffalitzky, F.; Zisserman, A.; Hartley, R. I.; and Torr, P. H. S., 2000. A six point solution for structure and motion. In *Computer Vision - ECCV 2000*, 632–648. Springer Berlin Heidelberg, Berlin, Heidelberg. (cited on pages 32 and 88)

Schau, H. and Robinson, A., 1987. Passive source localization employing intersecting spherical surfaces from time-of-arrival differences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 35, 8 (1987), 1223–1225. (cited on page 55)

Schmid, C. and Zisserman, A., 1998. The geometry and matching of curves in multiple views. In *European Conference on Computer Vision*, 394–409. Springer. (cited on page 27)

Schmidt, S. F., 1981. The kalman filter-its recognition and development for aerospace applications. *Journal of Guidance, Control, and Dynamics*, 4, 1 (1981), 4–7. (cited on page 1)

Schönemann, P. H., 1966. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31, 1 (Mar 1966), 1–10. doi:10.1007/BF02289451. http://dx.doi.org/10.1007/BF02289451. (cited on page 107)

Seitz, S. M. and Baker, S., 2009. Filter flow. In *Computer Vision, 2009 IEEE 12th International Conference on*, 143–150. IEEE. (cited on page 28)

Shen, S.; Michael, N.; and Kumar, V., 2011. Autonomous multi-floor indoor navigation with a computationally constrained mav. In *2011 IEEE International Conference on Robotics and Automation*, 20–25. doi:10.1109/ICRA.2011.5980357. (cited on page 74)

Shi, J. and Tomasi, C., 1994. Good features to track. In *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*, 593–600. IEEE. (cited on page 83)

Shoemake, K., 1985. Animating rotation with quaternion curves. *In Computer Graphics: Proceedings of SIGGRAPH âĂŹ85*, (1985), 245–254. (cited on page 106)

Simoncelli, E. P.; Adelson, E. H.; and Heeger, D. J., 1991. Probability distributions of optical flow. In *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR'91., IEEE Computer Society Conference on*, 310–315. IEEE. (cited on page 75)

Smith, G. L.; Schmidt, S. F.; and McGee, L. A., 1962. *Application of statistical filter theory to the optimal estimation of position and velocity on board a circumlunar vehicle*. National Aeronautics and Space Administration. (cited on page 13)

Sola, J.; Vidal-Calleja, T.; Civera, J.; and Montiel, J. M. M., 2012. Impact of landmark parametrization on monocular ekf-slam with points and lines. *International journal of computer vision*, 97, 3 (2012), 339–368. (cited on page 2)

Solomon, H., 1978. Buffon needle problem, extensions, and estimation of pi. *Geometric of probability*, (1978). (cited on page 41)

Song, S.; Chandraker, M.; and Guest, C. C., 2016. High accuracy monocular sfm and scale correction for autonomous driving. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38, 4 (April 2016), 730–743. doi:10.1109/TPAMI.2015.2469274. (cited on pages 74, 75, 97, and 98)

Sorenson, H. W., 1985. *Kalman filtering : theory and application*. The Institute of Electrical and Electronics Engineers,Inc. (cited on pages 2, 13, and 40)

Sorenson, H. W. and Alspach, D. L., 1971. Recursive bayesian estimation using gaussian sums. *Automatica*, 7, 4 (1971), 465–479. (cited on pages 2, 16, 17, 40, and 41)

Spragins, J., 1965. A note on the iterative application of bayes' rule. *IEEE Transactions on Information Theory*, 11, 4 (1965), 544–549. (cited on page 10)

Stano, P.; Lendek, Z.; Braaksma, J.; Babuska, R.; de Keizer, C.; and Arnold, J., 2013. Parametric bayesian filters for nonlinear stochastic dynamical systems: A survey. *IEEE transactions on cybernetics*, 43, 6 (2013), 1607–1624. (cited on page 40)

Sun, D.; Roth, S.; and Black, M. J., 2010. Secrets of optical flow estimation and their principles. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, 2432–2439. IEEE. (cited on page 29)

Sun, D.; Yang, X.; Liu, M.-Y.; and Kautz, J., 2018. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. (cited on page 75)

Sun, M. and Ho, K. C., 2010. Tdoa localization using closed-form solution, University of Missouri, Computational Intelligence, Signal Processing. http://cisp.ece.missouri.edu/code.php. Accessed: 2016-09-1. (cited on pages 56, 66, 67, and 68)

Templeton, T.; Shim, D. H.; Geyer, C.; and Sastry, S. S., 2007. Autonomous vision-based landing and terrain mapping using an mpc-controlled unmanned rotorcraft. In *Robotics and automation, 2007 IEEE international conference on*, 1349–1356. IEEE. (cited on page 74)

Thrun, S.; Burgard, W.; and Fox, D., 2005. *Probabilistic robotics*. (cited on page 39)

Torr, P. and Zisserman, A., 1998. Robust computation and parametrization of multiple view relations. In *Computer Vision, 1998. Sixth International Conference on*, 727–732. IEEE. (cited on pages 33 and 88)

Tukey, J. W., 1977. *Exploratory Data Analysis*. Addison Wesley, 1 edn. ISBN 0201076160. (cited on page 106)

Uhlmann, J. K., Simon J. Julier, 1997. A new extension of the kalman filter to nonlinear systems. 3068 (1997), 182–193. (cited on page 40)

Wan, E. A. and Van Der Merwe, R., 2000. The unscented kalman filter for nonlinear estimation. In *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*, 153–158. Ieee. (cited on page 13)

Wang, C.-L.; Wang, T.-M.; Liang, J.-H.; Zhang, Y.-C.; and Zhou, Y., 2013. Bearing-only visual slam for small unmanned aerial vehicles in gps-denied environments. *International Journal of Automation and Computing*, 10, 5 (2013), 387–396. (cited on page 74)

Wang, G.; So, A. M.-C.; and Li, Y., 2016. Robust convex approximation methods for tdoa-based localization under nlos conditions. *IEEE Transactions on Signal Processing*, 64, 13 (2016), 3281–3296. (cited on page 56)

Wang, Z.; Bovik, A. C.; Sheikh, H. R.; and Simoncelli, E. P., 2004. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13, 4 (2004), 600–612. (cited on page 94)

Wannenwetsch, A. S.; Keuper, M.; and Roth, S., 2017. Probflow: Joint optical flow and uncertainty estimation. In *2017 IEEE International Conference on Computer Vision (ICCV)*, 1182–1191. IEEE. (cited on page 76)

Wei, J. and Yu, C., 2016. Performance evaluation of practical passive source localization using two software defined radios. *IEEE Communications Letters*, 20, 9 (Sept 2016), 1880–1883. doi:10.1109/LCOMM.2016.2582797. (cited on pages xxi, 56, 57, 63, and 71)

Weiss, S.; Achtelik, M.; Kneip, L.; Scaramuzza, D.; and Siegwart, R., 2011a. Intuitive 3d maps for mav terrain exploration and obstacle avoidance. *Journal of Intelligent & Robotic Systems*, 61, 1-4 (2011), 473–493. (cited on page 74)

Weiss, S.; Achtelik, M. W.; Lynen, S.; Achtelik, M. C.; Kneip, L.; Chli, M.; and Siegwart, R., 2013. Monocular vision for long-term micro aerial vehicle state estimation: A compendium. *Journal of Field Robotics*, 30, 5 (2013), 803–831. (cited on page 74)

Weiss, S.; Scaramuzza, D.; and Siegwart, R., 2011b. Monocular-slam–based navigation for autonomous micro helicopters in gps-denied environments. *Journal of Field Robotics*, 28, 6 (2011), 854–874. (cited on page 74)

Whitley, D., 1994. A genetic algorithm tutorial. In *Statistics and Computing*, 65–85. (cited on page 17)

Wiener, N., 1949. Extrapolation, interpolation, and smoothing of stationary time series: With engineering applications. *Mit Press*, 113, 21 (1949), 1043–54. (cited on page 13)

Wiener, N. and Hopf, E., 1931. On a class of singular integral equations. *Proc. Prussian Acad. Math.-Phys. Ser., page 696pp*, (1931). (cited on page 1)

Wold, H., 1938. *A study in the analysis of stationary time series*. Ph.D. thesis, Almqvist & Wiksell. (cited on page 13)

Xu, G. and Zhang, Z., 1996. Epipolar geometry in stereo, motion and object recognition, volume 6 of computational imaging and vision. (cited on page 32)

Xu, J.; Chang, H.-w.; Yang, S.; and Wang, M., 2012. Fast feature-based video stabilization without accumulative global motion estimation. *IEEE Transactions on Consumer Electronics*, 58, 3 (2012). (cited on page 27)

Xu, J.; Ranftl, R.; and Koltun, V., 2017. Accurate optical flow via direct cost volume processing. In *2017 IEEE Conference on Computer Vision and Pattern Recognition*. (cited on pages xxi, 7, 73, 75, 76, 82, 83, and 84)

Xu, W.; Quitin, F.; Leng, M.; Tay, W. P.; and Razul, S. G., 2015. Distributed localization of a rf target in nlos environments. *IEEE Journal on Selected Areas in Communications*, 33, 7 (2015), 1317–1330. (cited on page 57)

Yang, S.; Scherer, S. A.; Yi, X.; and Zell, A., 2017. Multi-camera visual slam for autonomous navigation of micro aerial vehicles. *Robotics and Autonomous Systems*, 93 (2017), 116–134. (cited on page 74)

Yeredor, A. and Angel, E., 2011. Joint tdoa and fdoa estimation: A conditional bound and its use for optimally weighted localization. *IEEE Transactions on Signal Processing*, 59, 4 (2011), 1612–1623. (cited on pages 56 and 57)

Yin, J.; Wan, Q.; Yang, S.; and Ho, K. C., 2016. A simple and accurate tdoa-aoa localization method using two stations. *IEEE Signal Processing Letters*, 23, 1 (Jan 2016), 144–148. doi:10.1109/LSP.2015.2505138. (cited on page 56)

Zhang, T.; Wu, K.; Song, J.; Huang, S.; and Dissanayake, G., 2017. Convergence and consistency analysis for a 3-d invariant-ekf slam. *IEEE Robotics and Automation Letters*, 2, 2 (2017), 733–740. (cited on page 40)

Zhang, Z., 1998. Determining the epipolar geometry and its uncertainty: A review. *International Journal of Computer Vision*, 27, 2 (Mar 1998), 161–195. doi:10.1023/A:1007941100561. https://doi.org/10.1023/A:1007941100561. (cited on pages 33, 34, and 88)

Zhong, X.; Tay, W. P.; Leng, M.; Razul, S. G.; and See, C. M. S., 2016. Tdoa-fdoa based multiple target detection and tracking in the presence of measurement errors and biases. In *Signal Processing Advances in Wireless Communications (SPAWC), 2016 IEEE 17th International Workshop on*, 1–6. IEEE. (cited on page 56)

Zhou, B.; Lapedriza, A.; Xiao, J.; Torralba, A.; and Oliva, A., 2014. Learning deep features for scene recognition using places database. In *Advances in neural information processing systems*, 487–495. (cited on page 27)