Mobile Sensor Systems for Field Estimation and "Hot Spot" Identification

by

Sumeet Kumar

S.M., Mechanical Engineering, Massachusetts Institute of Technology (2009)

Submitted to the Department of Mechanical Engineering in partial fulfillment of the requirements for the degree of Doctor of Philosophy at the MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2014

© Massachusetts Institute of Technology 2014. All rights reserved.
Mobile Sensor Systems for Field Estimation and “Hot Spot” Identification

by

Sumeet Kumar

Submitted to the Department of Mechanical Engineering on December 13, 2013, in partial fulfillment of the requirements for the degree of Doctor of Philosophy

Abstract

Robust, low-cost mobile sensing enables effective monitoring and management of urban environment and infrastructure which contributes towards a sustainable future. While mobile sensor systems have attracted significant attention in recent years, a large scale deployment for urban infrastructure monitoring poses many research challenges. One fundamental challenge is dealing with noisy and uncontrolled samples stemming from both noisy sensor measurements and locations, and lack of control on sensor deployment. Such conditions pose difficulties in field estimation and “hot spot” identification from sensor data. My thesis contributions aim to bridge this gap. In this thesis, I designed and developed a mobile sensor system for urban light infrastructure monitoring and studied two problems on field estimation in the presence of noisy and uncontrolled samples with general implications on mobile sensing.

As an example system, I designed and successfully tested a city-wide street light scanning mobile sensor platform. Currently, street light maintenance uses labor-intensive, poorly scalable manual inspection techniques. My system automatically maps street illumination levels and lamp infrastructure. The collected data presents challenges in identifying lamp “hot spots” from false positives and using sensor data from noisy sensor locations for estimating luminosity maps. I present an algorithm for identifying various light sources from video streams and combining that data with location information to geotag lamps. To identity the light sources, I developed a supervised classifier for lamp classification and later extended it to develop a method for estimating the height of the street lamps. As accurate car location information is critical for successful luminosity mapping, I discuss how I improved car location estimates by integrating multiple independent data streams such as vehicle speed data obtained via on-board diagnostic systems and GPS.

Towards such field deployments, one must address fundamental challenges related to both hardware implementation and data analytics. In this thesis, I addressed two problems related to mobile sensor based sampling and signal estimation: nonuniform sampling and errors in sample location.

Uniform grid sampling is often impractical in the context of mobile spatial sampling...
pling. As an alternative, I studied \( \delta \)-dense sensor arrangements as a realistic scheme for non-uniform flexible sampling. To assess its utility, I derived sufficient conditions for estimating a signal represented in a finite-dimensional basis set and present simulation results on the numerical stability of signal estimation under \( \delta \)-dense sampling. Furthermore, I present the use of proper orthogonal decomposition as a technique to obtain basis sets for signals that are solutions to a parametric differential equation.

Finally, I studied how errors in the location measurements of mobile nodes affect the overall signal estimation problem. To address such signal estimation problems, I developed a computationally efficient iterative linear estimator and compared my approach to the state of the art expectation-maximization technique. I present simulation studies on the performance of the estimator and discuss results on its numerical stability. My approach offers several orders of magnitude reduction in computational time while achieving comparable mean squared estimation error to other techniques making it an appealing candidate for real-time embedded mobile sensing applications.

Thesis Supervisor: Sanjay E. Sarma
Title: Professor
Acknowledgments

It would be an understatement that my stay at MIT has had a transformative effect on me. At this juncture, when I am preparing to submit my thesis, I reminisce on my journey and feel grateful to the individuals and groups who have crossed paths or walked along.

My research advisor, Prof. Sanjay Sarma, took me under his wing in June 2009 for my PhD studies. Though my masters was focussed in a different research area, he showed tremendous confidence in me. I have learned a lot from his bold and systematic approach to research. Whenever I went to him with questions, he not only directed me towards the solutions space but also helped me understand what are the right questions I should be asking. Apart from his advising at academic and scientific levels, his mentoring at a personal level has prepared me effectively for my future career and life. Sanjay, thank you for everything!

I am very grateful to my thesis committee members, Prof. Franz Hover, Dr. Vivek Goyal and Dr. Ajay Deshpande. I would like to thank Dr. Ajay Deshpande for first directing me to the right classes and readings that enabled me to build strong fundamentals and then having numerous research discussions over phone. I am extremely grateful to Dr. Vivek Goyal for performing critical evaluation and asking hard questions to ensure high quality of work. Prof. Franz Hover helped me to connect the dots in my thesis and understand big picture implications of the problems I have studied. I am really thankful to him for his guidance.

I am extremely grateful to Dr. Todd Thorsen who supervised my masters research at MIT and helped me acclimate to the research culture at MIT. I would also like to thank Marco for the collaboration on my masters project.

I would like to thank all the current and former members of Sanjay’s research group for the camaraderie I shared with them on the second floor of building 35. Stephen, Isaac, Rahul, Hui, Dylan, Josh, Johannes, Abraham, Sriram, Justin, Long, Jonathan, Ed, Radu, Christian, Austin and Verena have openly brainstormed with me, encouraged me and created a collegial work environment. Special thanks to
Stephen and Sriram for proof reading the thesis draft and going through numerous edits with me. I cherish the collaboration with Abraham on the Ferrovial street light scanning project and multiple trips to Spain.

I am extremely thankful to the MIT community, Graduate Student Council (GSC) and the administrative staff for their continuous and selfless support on all fronts of my life. Leslie Regan, Joan Kravit, David Rodriguera, Cintia Castro, Jan Marie and Emilia Arimah have always provided me with an amazing administrative support.

“Friendship is the only cement that will ever hold the world together.” - Woodrow T. Wilson. I have been extremely fortunate to have my world held together by the amazing people I met, wonderful moments I experienced, stories I created and the everlasting memories I take from my graduate student life. Vivek and Sudhish have been my long-term roommates and best friends, celebrating in my joy and being compassionate in my struggles. Thank you guys for everything! I am extremely grateful to Bria for the support she provided and the wonderful memories we shared. A special thanks to Shahriar for the numerous discussions on life and career we had over coffee and lunches. I would also like to thank Harish, Rita, Ellan, Maya, Abishek, Suman, Sankha, Sara, Karthik, Dieter, Nikhil, Dipanjan, Vivek Inder, Srikanth, Sukant, Kushal, Varun, Vikrant, Raghavendra, Harshad, Sahil, Sarvee, Vignesh, Thomas, Meenal and Yuenan for all the wonderful times we had.

My family, mom, dad and my sister Shruti have always unconditionally loved me, encouraged me and believed in me. I am extremely grateful to them for their support. My aunt Gayathri and uncle Alok have always been there to lend an ear and cared for my happiness. My cousins and my relatives have constantly supported me from a far away land and I am extremely thankful to them. I attribute my accomplishments so far to my mom and dad.
This doctoral thesis has been examined by a Committee of the Department of Mechanical Engineering as follows:

Professor Sanjay E. Sarma.....................................................
Chairman, Thesis Committee
Professor of Mechanical Engineering

Professor Franz S. Hover....................................................
Member, Thesis Committee
Associate Professor of Mechanical Engineering

Dr. Vivek K. Goyal..............................................................
Member, Thesis Committee
Research Scientist at the Research Laboratory of Electronics

Dr. Ajay Deshpande..............................................................
Member, Thesis Committee
Research Staff Member at the IBM T J Watson Research Center,
Yorktown Heights, NY
Contents

1 Introduction 21
  1.1 Networking the Physical World ........................................... 21
  1.2 Mobile Sensor Systems .................................................. 22
    1.2.1 What are the main challenges? ................................. 24
  1.3 Thesis Contributions .................................................... 26
  1.4 Thesis Outline ............................................................ 27

2 A Car-Top Street Light Scanning Platform 29
  2.1 Urban Street Lighting .................................................. 30
  2.2 Initial Prototype ....................................................... 32
    2.2.1 Sensor Platform .................................................. 32
    2.2.2 Data Collection System ........................................... 34
    2.2.3 Lessons Learned .................................................. 36
  2.3 Car-mounted Autonomous Street light scanning Platform (CASP) .... 39
    2.3.1 Sensor Selection and Integration ............................... 39
    2.3.2 Car-top Platform ............................................... 40
    2.3.3 Data Logging and Management .................................. 41
    2.3.4 Field Studies .................................................... 43
  2.4 Conclusion ............................................................... 44

3 Analysis of Street Lighting Data 47
  3.1 Luminosity Maps ....................................................... 48
  3.2 Integrating OBD-II speed data with GPS measurements ............ 50
3.2.1 Vehicle Trajectory Estimation ............................................... 52
3.2.2 Empirical Validation .......................................................... 57
3.2.3 Summary ................................................................. 63
3.3 Street Lamp Identification ......................................................... 64
  3.3.1 Feature Selection .......................................................... 68
  3.3.2 Learning and Optimizing a Classifier ................................ 76
  3.3.3 Summary ................................................................. 82
3.4 Lamp Post Height Estimation ...................................................... 84
  3.4.1 Perspective Projection .................................................... 84
  3.4.2 Optical Flow .............................................................. 87
  3.4.3 Algorithm for Height Estimation ..................................... 89
  3.4.4 Summary ................................................................. 90
3.5 A Three-dimensional Light Field Model ...................................... 91
3.6 Conclusion ................................................................. 93

4 Arrangements and Deployment of Mobile Sensor Nodes .......... 95
  4.1 Stable δ-Dense Sensor Arrangements for Sampling Physical fields 99
    4.1.1 Geometry of δ-Dense Sensor Arrangements ...................... 101
    4.1.2 The Sensor Arrangement Problem ................................. 101
    4.1.3 Sufficient Conditions for Stable δ-Dense Sensor Arrangements 102
    4.1.4 Numerical Simulations Showing the Stability of Linear Estima-
        tors in δ-Dense Sensor Arrangements ................................ 108
  4.2 Obtaining Basis Functions for Physical Fields .................. 109
    4.2.1 Example Field Modeling through POD and Field Estimation
        through δ-Dense Sensor Arrangement .............................. 111
  4.3 Conclusion ................................................................. 117

5 Signal Estimation from Samples with Location Errors ........ 119
  5.1 Notation ................................................................. 121
  5.2 Parameter Estimation .................................................... 123
  5.3 Iterative Linear Estimator (ILE) ..................................... 124
5.4 Cramér-Rao Lower Bound and Expectation-Maximization Algorithm

5.4.1 Numerical Integration ........................................ 128

5.4.2 Cramér-Rao Lower Bound and its Numerical Approximation . 131

5.4.3 An Expectation-Maximization Algorithm for Maximum Likelihood Estimation ......................... 132

5.5 Numerical Simulations ........................................ 134

5.5.1 Performance of ILE ......................................... 135

5.5.2 Convergence and Stability of ILE .......................... 138

5.5.3 Comparison of ILE with CRB and EM algorithm ........ 142

5.6 A Joint Maximum-a-Posteriori (JMAP) Estimator ............. 145

5.7 Conclusion .................................................... 147

6 Conclusion .................................................... 151

6.1 Summary of Contributions ..................................... 151

6.2 Future Work .................................................. 152
List of Figures

2-1 Sensors and electronics in the initial prototype. Courtesy: Google Images. ........................................ 34
2-2 (a) Initial prototype of the sensor platform and (b) Successfully integrated car-mounted sensor platform deployed for field testing. .................................................. 35
2-3 Architecture of the Data Collection System [110]. ................................................................. 36
2-4 Luminosity maps of certain roads in (a) Birmingham, UK and (b) Torrejon, Spain. ....................... 37
2-5 Light sensor measurements by the three light sensors oriented in different directions as show in Fig. 2-2. ................................................................. 38
2-6 Images from surveillance cameras providing geometric information of the street lamps. ......................... 38
2-7 Sensors and electronics in CASP. Courtesy: Google Images. .................................................. 40
2-8 (a) Light Sensor Array and (b) Assembled car-top sensor platform. ........................................... 42
2-9 Architecture of the Data Logging System [110]. ................................................................. 43
2-10 Data collection loop. ........................................................................................................ 44
3-1 Color scheme adopted to create the two-dimensional luminosity maps. ........................................ 48
3-2 Two examples of street lighting levels indicated by vertical cylinders. ......................................... 49
3-3 Variation of $l_{\text{avg}}$ with traveled distance. ........................................................................... 50
3-4 Simplified vehicle motion model in Cartesian coordinates ....................................................... 52
3-5 (a) Scaling and bias estimates as a function of time; (b) OBD data (blue), raw accelerometer integral (red) and scaling, bias corrected accelerometer integral (green). ......................................................... 60
3-6 Comparison of vehicle reference trajectory (in black) with EKF trajectory estimate (in red) for different GPS sampling time. 60
3-7 Variation of $\epsilon_d$ with $T_s$ 61
3-8 Comparison of the performance metrics for different GPS outage probability: (a) $\epsilon_r$ and (b) $\epsilon_d$. The solid lines indicate the mean of the metric while the dashed lines indicate mean $\pm \frac{1}{2}$ standard deviation. 62
3-9 Contour plots of the performance metrics (a) $\epsilon_r$ (m) and (b) $\epsilon_d$ (%) with the variation in GPS downsampling factor and IMU/OBD/KF sampling time. 62
3-10 Example images collected by the mobile sensor platform. 64
3-11 Aspects of Image Classification 66
3-12 For the original images in the top row the bottom row contains the corresponding binary image. 68
3-13 Example images of lamps. 69
3-14 Example images of other bright objects in the scene. 70
3-15 Normalized histogram of grayscale intensity. 70
3-16 Black and white binary image. 71
3-17 Example texts added by the camera on the images. 74
3-18 (a) Variation of 10-fold misclassification rate with $p$ and (b) Variation of size of the variable set selected by the Fisher score ranking with $p$. 80
3-19 Schematic of the steps involved in creating the hybrid lamp classifier. 83
3-20 Geometry of the car-mounted imaging system. 84
3-21 Perspective projection of a camera imaging system [33]. 85
3-22 Axes definition and imaging geometry. 86
3-23 (a) Example grayscale image frame, (b) Magnitude of $x$–direction intensity gradient and (c) Magnitude of $y$–direction intensity gradient. 88
3-24 A model of lamp light field. 91
4-1 Field estimation and “hot spot” identification. 96
4-2 A δ-dense sensor arrangement in 2D with the sampling locations at $x_i$. Each square partition $S_i$ is of size $δ \times δ$ and has one sampling point.

4-3 A sensor arrangement in 1D with the sampling locations at $x_i$ and partitions of size $δ$ denoted by $S_i$.

4-4 Variation of the mean of condition number of $B$ and $B_2$ with $δ$ for 2D a) orthonormal polynomial basis functions; b) orthonormal cosine basis functions. For $δ \leq 0.025$ I obtain small condition numbers implying higher numerical stability and robustness to noise. The error bars indicate $1/2$ standard deviations.

4-5 (a) Variation of diffusivity, source term and chemical concentration profile in the domain; (b) First 5 POD basis functions and the corresponding singular values ($|σ_i/10|$) generated for the model in (4.36); (c) Decay of singular values with the POD basis function number.

4-6 (a) Variation of the mean of condition number of $B$ and $B_2$ with $δ$ for the PODs $(α = 0.01)$; (b) MSE in reconstructing the test function through δ-dense sampling with different magnitudes of additive noise $(α)$. I report error in direct reconstruction through weighted least squares (4.5) in blue and least squares (4.6) in red. The error bars indicate $1/2$ standard deviations.

4-7 Comparison of the MSE with uniform sampling and δ-dense sampling. Note that the weighted least squares (WLS) and least squares (LS) estimator are identical for uniform sampling. $N = 1/δ$. The error bars indicate $1/2$ standard deviations.

4-8 Comparison of the MSE with three different choices of basis function sets. POD refers to the POD basis function set, Trig refers to the trigonometric basis function set and Poly refers to the polynomial basis function set. For all the simulations $α = 0.01$ and the least squares method (4.6) was used.
4-9 Comparison of the MSE with the three basis function sets when there is an approximate knowledge of model parameters. (a) $D'(x_1) = D(x_1) + 0.01 \sin(5x_1)$; (b) $D'(x_1) = D(x_1) + 0.01 \cos(10x_1)$; (c) $S'(\mu) = \exp(-500(x_1 - \mu)^2)$; (d) $S(\mu) = \exp(-20(x_1 - \mu)^2)$. 

5-1 Comparison of the histogram of the sample $y_i$ (Hist) with numerical approximation of the pdf through Gauss-Hermite quadrature (GHQ) and Monte Carlo integration (MCI); $\sigma = 0.053$. 

5-2 Comparison of the mean of the squared estimation error $||\hat{a} - a||^2$ with ILE, BE and KLE for trigonometric basis functions. The error bars indicate 1/2 standard deviations. (a) Comparison of the performance with the variation in $p$. Solid line corresponds to $\gamma = 1$, dashed line $\gamma = 3$, dotted line $\gamma = 5$ and for all the simulations $N = 200$; (b) Comparison of the performance of ILE against BE with the variation in $N$. The percentage sample location error ($p$) is indicated on the left of each set of plots and $\gamma = 1$. 

5-3 Comparison of the mean of the squared estimation error $||\hat{a} - a||^2$ with ILE, BE and KLE for polynomial basis functions. The error bars indicate 1/2 standard deviations. (a) Comparison of the performance with the variation in $p$. Solid line corresponds to $\gamma = 1$, dashed line $\gamma = 3$, dotted line $\gamma = 5$ and for all the simulations $N = 100$; (b) Comparison of the performance of ILE against BE with the variation in $N$. The percentage sample location error ($p$) is indicated on the left of each set of plots and $\gamma = 1$. 

5-4 Comparison of $\Lambda$ with the variation in $p$. 

5-5 Variation of the mean of the computational time for ILE with % sample location error ($p$) for different % additive noise ($\gamma$) and the number of sampling points ($N$).
5-6 Comparison of the mean of the squared estimation error $\| \hat{a} - a \|^2$ with ILE, BE and KLE for two-dimensional polynomial basis functions. The error bars indicate 1/2 standard deviations. (a) Comparison of the performance with the variation in $p$. Solid line corresponds to $\gamma = 1$, dashed line $\gamma = 3$, dotted line $\gamma = 5$ and for all the simulations $N = 25$; (b) Comparison of the performance of ILE against BE with the variation in $N$. The percentage sample location error ($p$) is indicated on the left of each set of plots and $\gamma = 1$.

5-7 Variation of the normalized standard deviation of the norm of the converged parameter estimates (std$||\hat{a}||/||a||$) with $p$ for one-dimensional polynomial basis functions.

5-8 Variation of the moving average of the squared estimation error with the iteration number for seven randomly chosen simulation settings with the one-dimensional polynomial basis functions.

5-9 Comparison of the mean of the squared estimation error $\| \hat{a} - a \|^2$ with ILE, BE and EM with the variation in $p$. The error bars indicate 1/2 standard deviations. The approximated CRB is presented by the black line. For all the simulations $N = 50$.

5-10 Marginal probability distribution of $y_i$ for one-dimensional trigonometric basis functions with $||a|| = 3.4528$ and $x_i = 0$.

5-11 Comparison of the mean of the squared estimation error $\| \hat{a} - a \|^2$ with ILE and EM as a function of $N$. The error bars indicate 1/2 standard deviations. The approximated CRB is presented by the black line. For all the simulations $\gamma = 1$.

5-12 Comparison of the mean of the squared estimation error $\| \hat{a} - a \|^2$ with ILE, JMAP and BE with the variation in $p$. The error bars indicate 1/2 standard deviations.
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Number of variables $</td>
<td>S</td>
</tr>
<tr>
<td>3.2</td>
<td>10-fold misclassification rate in %.</td>
<td>79</td>
</tr>
<tr>
<td>3.3</td>
<td>10-fold misclassification rate in %.</td>
<td>80</td>
</tr>
<tr>
<td>3.4</td>
<td>10-fold misclassification rate in %.</td>
<td>81</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

In this thesis, I have studied several aspects of mobile sensor systems. On the application side, I developed a car-mounted light scanning platform that collects data on urban street illumination and lamp inventory. On the theoretical side, I studied field/signal estimation under non-uniform arrangements of sensor nodes and addressed field estimation when observations are corrupted by both additive noise and uncertainty in the location of the samples. I begin by presenting background and applications of sensor networks. I then discuss mobility as a key element of sensor networks and discuss opportunities for mobile sensor systems. Thereafter, I highlight the challenges in developing and implementing such systems for natural and urban environment monitoring. I conclude the chapter with a summary of the key contributions and an outline of the rest of the thesis.

1.1 Networking the Physical World

The advancements in integrated circuits, microelectromechanical systems and wireless communications have led to the development of cheap, compact, accurate and reliable sensors. Today's sensor nodes like Mica Mote, Sun SPOT and even smartphones are capable of sensing location, acceleration, light intensity, temperature, pressure, relative humidity, air quality and chemical concentrations using built-in or add-on sensors [12, 9, 5, 124]. Researchers and practitioners have been looking into deploying
such sensor nodes to collect unprecedented amounts of data on natural and urban environments [12, 89, 123, 124].

Much of the earlier work in sensor networks has focused on the development of sensor nodes, communication and networking issues [12, 45, 68, 96, 124, 141]. Typical applications involve estimating a spatio-temporal field, identifying a “hot spot” which is a region of high values, or locating an event of interest. Recently there has been a growing interest in incorporating these questions in the overall design and implementation of sensor networks. Researchers have addressed both data acquisition and analysis techniques for event detection applications, such as surveillance and leak detection in water distribution pipes [41, 29, 49, 94, 32, 125]. Furthermore, various machine learning and signal estimation approaches have been explored to monitor a spatio-temporal field of interest such as temperature distribution, chemical concentration, radiation and crop yield [42, 146, 93, 89, 143, 116, 83].

1.2 Mobile Sensor Systems

Mobility in sensor networks has attracted much attention recently and can significantly enhance spatial coverage of sensor networks. On the one hand mobility in sensor networks might imply the use of data mules and robotic agents that can collect data from static sensors [23], [43]. On the other hand, due to their compact form factors, sensor nodes can be easily attached to mobile platforms such as robots, cars, buoys, humans and animals to perform wide area sensing [36], [45], [40]. The emergence of smartphones, embedded with sensors like GPS and accelerometers, have further enhanced the opportunities of ubiquitous mobile sensing [80], [56], [100]. In this thesis, we are interested in the second kind of mobility, namely mobile sensor systems viewed as mobile platforms equipped with sensing capabilities.

There has been a growing interest in infrastructure management and environment monitoring. Developed countries suffer from aging urban infrastructure and fear of failure while the developing countries are concerned about ensuring quality of recent constructions. Governments, businesses and local authorities often face grave
challenges during accidents like the Deepwater Horizon oil spill, Fukushima Daiichi nuclear disaster, the Tokyo Sarin gas attack and wildfires. The financial crisis of 2008 have made matters worse as many cities face severe budget shortfalls, several of which are teetering on the brink of bankruptcy [3]. Manual surveys are typically employed to collect data on environment and infrastructure but they suffer from low reliability, coverage and high cost. Another approach to collect data is crowdsourcing, where citizens participate in collecting and sharing data. This has emerged as an exciting opportunity but suffers from low participation due to lack of proper incentives and/or the need of expert knowledge [45], [56], [144].

Large-scale deployment of sensor nodes to collect data on infrastructures and environment is anticipated to address the aforementioned challenges by providing invaluable information to the decision makers. Traditionally, sensor networks comprise of static nodes that are often densely deployed to achieve sufficient coverage and connectivity. Static deployments are often unsuccessful in providing long-term data collection and monitoring because the sensors nodes are limited in power, computational capacity, communication range and memory and they are prone to failures [23], [43].

In [40], [42], [43] the authors define intentional and incidental mobility. Mobile platforms like robots and UAVs demonstrate intentional mobility, while platforms like taxis and humans are incidentally mobile. Mobile sensor systems where a stand-alone sensor node is attached to an incidentally mobile platform like cars, drifting buoys, buses and garbage trucks present an opportunity for more effective crowdsourcing. The mobility of such a platform ensures wide area coverage while the self contained, autonomous sensor node allows collection of scientifically accurate and reliable data with minimal user intervention. Transfer of data for analysis can be achieved by either using a wireless communication protocol on the platform or an off-line wired download once the mobile platform has completed its task. Furthermore, deployment cost with incidentally mobile platforms is minimal making such solutions financially attractive.
1.2.1 What are the main challenges?

Though an attractive concept, mobile sensor systems have significant challenges for developing effective natural and urban environment monitoring systems. The key challenges include:

- **Sensor Integration and Data Management:** Nowadays, sensor nodes are available from a number of manufactures and vendors in the form of transducer integrated circuits (IC), breakout boards, development kits and even standalone units [9]. A transducer IC provides basic electronics while breakout boards include additional electronics to regulate input and process the transduced signal into a specific analog or digital output format. Development kits allow integration of several transducer ICs and breakout boards and provide them with input/output ports and often means of data storage and transfer. Digital cameras are prime examples of standalone units that sense, process and store data. Often several sensor nodes need to be integrated to a mobile platform. A central challenge here is to handle different streams of sensor data, appropriately time-synchronize different nodes and log the data in a usable format. When wireless means are employed for data transfer to a cloud or a base station, issues like data rate and packet loss need to be addressed. Power management is another important challenge. Mobile platforms like cars can power sensors through cigarette charger lighter ports but battery life is critical for smartphones based sensing applications.

- **Sensor Deployment:** Path planning and motion control are important challenges that need to be addressed for the deployment of mobile platforms [117]. In recent years, researchers have routinely addressed the deployment of intentionally mobile sensors such as robotic sensors and autonomous underwater vehicles (AUVs) [120, 85, 146, 119, 142]. However, sampling with incidentally mobile sensors, where there is a poor control on the exact placement of sensor nodes, is an emerging problem [78], [42]. A fundamental question in sensor deployment is when and where sensors should take observations so that we maximize a sensing
metric while respecting the energy and motion constraints of a mobile sensor node. When using multiple nodes, coordination and collective motion need to be addressed. For incidentally mobile sensors, sampling at specific locations may not be feasible; here we aim to optimize the sensing and cost metric given we have less control on the exact position of the sensor nodes. For example, if air quality sensors are mounted on taxicabs to map urban emission data and since the speed of a taxicab is not constant, we have to address what would be an appropriate sampling scheme to obtain a good estimate of the urban emission field.

- **Field Estimation and “Hot Spot” Identification:** Sensors collect samples of a physical quantity that forms a space-time field over a region of interest. In many cases, we need to reconstruct the field and use it to predict how it will evolve or to know the locations of its peaks, or “hot spots”. For example, in the case of a chemical spill in a river or an ocean, we may want to map the concentration distribution and find the origin of the spill. The field estimation problem is typically addressed by assuming a specific mathematical model for the field and using the sensor measurements to estimate the unknown parameters of the model. Examples of models studied so far include band-limited functions [130], shift-invariant functions [130], [13], models based on proper orthogonal decomposition [102], [111], Markov random fields and Gaussian processes [34], [76]. Field estimation is closely linked with sensor deployment because the sensor arrangement has direct influence on the estimation of model parameters [78, 42, 76, 119]. Furthermore, the choice of the model is itself a fundamental question and often requires prior knowledge or further refinement. For an event detection problem, the goal is to categorize the observations to a particular class or hypothesis. The classification can be done through video analysis, signal processing, machine learning or statistical tests such as log likelihood ratio [129, 87, 106, 21]. Again, the sensor arrangement and the classification technique are fundamental questions here as they affect the quality of event
detection like accuracy and false positive rate. For example, if we want to use a
group of AUVs mounted with sensors to map chemical leak in a water body and
identify the leak source, we have to address what field models and deployment
trajectories would provide a good estimate.

1.3 Thesis Contributions

In this thesis, I have addressed several problems at the intersection of the challenges
described in Section 1.2. I base my research on a project in which we tried to map
the street lighting in urban areas using light sensors and video cameras.

- I have designed and prototyped a car-top sensor platform that collects data on
  street illumination and lamp inventory at a city-wide scale.

- I have developed data-mining algorithms to:
  - identify street lamps from video frames, estimate their heights and geotag
    them.
  - combine light sensor measurements with location data to create luminosity
    maps.
  - fuse car speed and accelerometer data with sparse GPS measurements to
    improve location estimates of the car.

- I have studied δ-dense sensor arrangement as a flexible, non-uniform sampling
  scheme with mobile nodes.

- I have proved sufficient conditions for reconstructing parametric signals modeled
  in finite basis through δ-dense sensor arrangement.

- I have developed a computationally efficient parametric signal estimation tech-
  nique when observations collected by mobile nodes are corrupted by both addi-
  tive sensor noise and sample location error.
1.4 Thesis Outline

The thesis is organized as follows. In Chapter 2, I present the design and prototyping of a car-mounted sensor platform for urban street light scanning. I then present the algorithms for creating luminosity maps and inventory of street lamps in Chapter 3 and discuss some fundamental issues in the analysis of data collected by mobile sensor systems. I address the problem of estimating fields when non-uniform sampling is performed by mobile sensors nodes in Chapter 4. In Chapter 5, I study field estimation in the presence of sample location errors. I conclude with a summary of results and contributions and outline future research directions in Chapter 6.
Chapter 2

A Car-Top Street Light Scanning Platform

Vehicle mounted or vehicle integrated mobile sensor systems have gained significant attention in the last few years with a potential to collect large scale, scientifically accurate urban data. Google Street View is one such example where images collected by a car driving on the streets are stitched together to create 360° panoramic views [18]. Though Google Street View has found some application in urban infrastructure monitoring [112], [20], they are not suited for street light scanning as the system and analysis is designed for daytime imaging. In this chapter, I present the design and development of a car mounted sensor platform for monitoring urban street lighting infrastructure. I first present background on the socio-economic importance of night-time street lighting. I discuss the challenges related to street lighting and optimization that infrastructure management companies and city governments are trying to address. Then, I describe an initial prototype and discuss its performance and the lessons learned from its testing. Thereafter, I present the design of the final prototype, the sensors selected and their integration into a car-top platform. I describe our data management and data logging system and address challenges related to developing a standalone system. Briefly, I discuss the performance of data collection during field experiments, postponing the detailed discussions on data analysis to Chapter 3.
2.1 Urban Street Lighting

There are nearly 40 million streetlights in the US alone, and they consume 31 TWh of energy per year [2]. Worldwide numbers will increase with urbanization as billions of people move to urban centers. International lighting standards require a certain threshold of street lighting ranging from 1 lux of light at the surface of residential suburbs to 50 lux at road intersections [1]. Oddly, the science of streetlight placement is relatively primitive today, and the means to monitor how much light reaches the street are very limited. The standards for measuring lighting are complicated, manual, and rarely implemented at a city-wide scale.

Furthermore, like much of the infrastructure in the developed world, lighting infrastructure has aged. Monitoring streetlights is a tedious manual task that relies on inspection and incident reports. A common practice is to have a city official drive scheduled routes and observe the status of street lamps. This approach lacks scalability and suffers from the lack of scientific reliability. Another approach involves local government blocking off a specific road and measuring light levels using a lux meter. Though this approach provides accurate and reliable data, it is labor intensive, not scalable and may cause significant inconvenience to its citizens. Researchers have also explored deployment of static light sensor nodes on streetlight poles for monitoring and control of lamps remotely [73], [69]. Though the static sensors measure the lighting levels accurately close to the lamp, the decision makers are more interested in measuring street level illumination. Furthermore, such static deployment suffers from the drawbacks discussed in Chapter 1, Section 1.2, such as, cost, maintenance and node failures.

Matters are not helped by the 2008 financial crisis, after which many cities have faced severe budget shortfalls and several of them are teetering on the brink of bankruptcy. Los Angeles, for example, spends $52M annually on light maintenance while the city faces an overall $72M budget shortfall [3]. To offload infrastructure costs, some cities are now outsourcing lighting activity to private companies on fixed-price contracts, but have inadequate methods to measure the service level delivered.
to citizens. The contractors themselves are considering replacing old lights with LED units, but are struggling to figure out what the inventory of lighting is, how many lux they deliver on the street level, and what their condition is. LED lights can be dimmable, which opens up another avenue: modulating lights to ensure that energy is minimized without compromising safety or security. This too requires better assessment and measurement tools.

There is a pressing need for a scalable way to measure and map street lighting. The information needs to be updated regularly enough that timely operational and maintenance decisions are enabled, but need not be real time. In short, imagine a form of “Google Street View” with a semi-live, updated view of lighting along city streets and its utility to both citizens and decision makers.

Ferrovial is a Spanish multinational company which is involved in the construction, maintenance and management of urban infrastructure and services [6]. In 2010, Ferrovial partnered with MIT to work on several research projects in the areas of intelligent cities, clean water and energy efficiency. During my thesis, I had the opportunity to work with Ferrovial engineers to develop a technology for measuring and optimizing street lighting. Ferrovial has contracts with several of the municipalities and the city governments in Spain and the UK and manages and maintains over 1 million streetlights. Through discussions with the Ferrovial management team and the city governments, I first tried to assess the requirements of street lighting measurements. Regularly updated luminosity maps, which are essentially an overlay of average street illumination on a map, can help the city officials to monitor the current state of service. Furthermore, appropriate analysis of these maps can help them to identify over-lit and under-lit areas and will open opportunities for saving energy or improving service. Ferrovial is also very interested in developing scalable techniques for creating databases of geotagged street lamps as well estimating their heights and type. This information can be extremely helpful when planning a repair or maintenance operation. Also, broken or damaged lamps can be identified by combining semi-live luminosity maps with lamp inventory. Another problem the cities want to address is to develop a realistic simulation model of the three-dimensional light field.
using accurate data on street lamp locations, luminaire characteristics and street geometry. Such a simulation environment can assist them in evaluating the effect of adding or removing light fixtures on street lighting.

2.2 Initial Prototype

After gaining an understanding of the requirements of the industry, I aimed to develop an initial prototype that will allow us to collect data and perform preliminary analysis. A major challenge in the design of mobile sensor systems is to find the right balance between sensors and hardware and the supporting analytics. If the hardware is too complex and expensive then the deployment of such systems may not be feasible. On the other hand if the data quality is too low then it may be infeasible to implement inference algorithms and extract useful information.

2.2.1 Sensor Platform

Design for deployment is a key aspect in engineering car-top sensor systems. Design for deployment requires one to address important questions such as:

- Are the fixtures for sensors appropriately designed such that the transducers are exposed to the environment (in the case of point sensors like light sensors) and the sensors have a good field of view (in the case of imaging systems)?

- How are the supporting microcontrollers, electronics and hardware assembled? Are they all placed on top of the car or some of them are placed inside the car? If some electronics are placed inside the car then how do they communicate and transfer data with the electronics placed on top of the car?

- How are the electronics mounted on top of the car? Can a mounting platform be designed that can hack into standard car mounts like a roof rack?

- How are the electronics powered? Can a cigarette charger provide enough power or is a supplemental battery pack needed?
• Is the data stored on a secondary storage device like an SD card or an external hard drive or transferred wirelessly to a server or to the Cloud? The choice will depend on the expected data rate from the sensor platform.

To summarize, design for deployment addresses the proper assembly and powering of sensors and electronics on a car such that relevant data is collected, stored and then transferred for future post-processing.

Given our goals, we are interested in measuring the amount of light incident on the streets and also to get information on the geometry of street lamps. Furthermore, we will need location information of the car as we are performing geospatial analysis. For our initial prototype I decided to select a Sun SPOT for measuring illuminance [10]. Sun SPOT is a Java programmable development toolkit which is quick to set up and has an embedded light sensor and accelerometer (Fig. 2-1a). The embedded light sensor has a resolution of 2 lux and a range from 0 lux to 1000 lux. The nodes are equipped with Li-ion rechargeable batteries and are capable of transferring data wirelessly to a basestation that can be connected to a laptop using IEEE 802.15.4 protocol. I selected an Aposonic 4 camera + 4 channel DVR set (Fig. 2-1b) for recording video data on light fixtures. To obtain location information of the car I used a Qstarz BT-Q818XT GPS (Fig. 2-1c) which has an accuracy of 3-15 m and can transfer data wirelessly over bluetooth. I used a Cyberpower AC mobile power cigarette charger to power the cameras during deployment (Fig. 2-1d).

Fig. 2-2a shows the assembly of the Sun SPOT node (light sensor) and the surveillance video cameras on an aluminum platform which was then mounted on a car roof rack. The light sensor measures normal incident luminous flux and hence it is sensitive to orientation. In order to understand the directional dependence of street illumination, we mounted the three sensors at different orientations. The center light sensor was mounted facing up while the left and the right light sensors were mounted facing +60° and -60° with respect to the vertical direction. Given our goal of imaging the street lighting inventory, we designed a semicircular fixture to mount the four surveillance cameras at an angular separation of 36° that allowed imaging the half plane above the car’s roof as shown in Fig. 2-2. We mounted the light sensor fixture
Figure 2-1: Sensors and electronics in the initial prototype. Courtesy: Google Images.

and the camera array on an aluminum platform which was then attached to a commercially available roof rack as shown in Fig. 2-2a. Fig. 2-2b shows the assembled sensor platform mounted on top of a car. During the field studies, we placed the GPS inside the car on top of the dashboard.

2.2.2 Data Collection System

During the course of this research, I worked with Abraham M. Rosenfeld to address data collection and management. We first wanted to identify a good methodology to log appropriately time synchronized data from the three Sun SPOTs, four video cameras and the GPS. We decided to use a laptop computer as the central data repository where each sensor pushes its respective data via a communication port at specified sampling frequencies [110].

Fig. 2-3 is a pictorial representation of the steps involved in implementing the data collection system. The Sun SPOT uses an IEEE 802.15.4 protocol to communicate
to the base station which is attached to the laptop via a USB port. We used a sampling frequency of 10 Hz in our implementation. The GPS uses Bluetooth for wireless communication with the laptop at a sampling frequency of 10 Hz. The cameras were connected to the DVR which was then connected to the laptop via an ethernet cable for real time data logging on the laptop. We developed separate data collection application for every sensor. The application essentially listens to a specific communication port and whenever data is available, it parses the data, attaches the time stamp of the local laptop and saves the result in an appropriate format in a .csv file. For the camera system, the application simply grabs the frame from every channel at a particular sampling frequency, uses the time stamp of the local laptop as the file name, for example, ‘15.36.58.331.jpeg’ and saves the image in a folder corresponding to the channel name. Every image is of the dimension 352 × 240 pixels. In our implementation we used a 10 Hz sampling frequency for every channel.

During our field studies, we found the total data rate to be of the order of 1.5 GB/hr. Though significant progress has been made in wireless communication and nowadays technologies like 4G LTE offer upload speed of around 4 GB/hr [4], wireless data transfer suffers from environmental effects, channel capacity and cost and is not yet a feasible option for transferring large amounts of data (a night-time survey of street lights in a city may need 5 hours of driving generating around 7.5 GB of data).
2.2.3 Lessons Learned

The initial prototype worked reliably and we were able to collect usable data during our deployment studies. It was a step in the right direction and helped us understand important issues in developing a standalone light scanning mobile platform. Fig. 2-4a,b show luminosity maps of few roads in Birmingham, UK and Torrejon, Spain respectively. These maps are created by overlaying the average of the three light sensor measurements on Google maps, using the location data obtained by the GPS. Our driving speeds were of the order of 30 - 35 mph or around 15 m/s. During analysis, we identified that a spatial resolution of $\sim 5$ m, or a sampling frequency of 3 Hz, is sufficient for creating visually clear luminosity maps and at the same time it provides enough data for studying variability in street lighting (street lamps are usually located at a separation of $\sim 20 - 30$ m). Fig. 2-5 shows the effect of orientation on the light sensor measurements. Since the street lamps are located at the edge of the roads, they may not be placed facing each other and since the data
is collected by a car driving on a specific lane, we expect different levels of light incident from different direction and hence a variation in the light collected by the sensors at different orientations. This also implies that solving the inverse problem of estimating street level illumination needs additional information about the geometry of light sources, for example, through cameras. Since our light sensor had a resolution of 2 lux, we were not able to create a finely resolved luminosity map, especially at low lighting levels ($\sim 0 - 10$ lux). We decided to use a higher resolution light sensor for the next prototype. We found the GPS to be fairly accurate but GPS suffers from outages especially in urban driving conditions. We resolved to augment our sensor platform with an inertial measurement unit (IMU) that provides accelerometer and gyroscope data and an on-board diagnostic (OBD) reader which provides car speed data to implement sensor fusion for improving estimates of car location.

![Luminosity maps](image)

(a) Birmingham, UK and (b) Torrejon, Spain.

The surveillance camera set was used to collect images of the street lamps. Our goal was to use these images to identify lamps, geotag them and estimate their heights using pattern recognition and computer vision techniques. Fig. 2-6 depicts two such example images. We observe that the images are saturated and blurred and not suitable for implementing lamp classification algorithms. The cameras have fixed ex-
Figure 2-5: Light sensor measurements by the three light sensors oriented in different directions as shown in Fig. 2-2.

Exposure settings of gain, aperture and shutter speed which is not suitable for night time imaging and resulted in the images having a high number of saturated pixels. Pixels with incident light flux above a certain threshold are saturated and cannot be used to make inferences about the brightness of the scene. There is a need to minimize the number of saturated pixels. From our initial analysis we found that capturing brightness gradient in the regions surrounding a lamp is critical for implementing pattern recognition techniques. Obtaining sharper images with higher details of brightness gradient is critical for implementing lamp classification and optical flow algorithms and we aimed to address them in our next version of the sensor platform.

Figure 2-6: Images from surveillance cameras providing geometric information of the street lamps.
2.3 Car-mounted Autonomous Street light scanning Platform (CASP)

I addressed the shortcomings discussed above in the design and development of the final prototype: CASP. A prime goal was to identify off-the-shelf sensors that provided higher accuracy and at the same time allowed for easy integration into a microcontroller. Since we wanted to develop a user-friendly standalone system, the microcontroller should be robust and easy to debug and operate. Furthermore, the camera system should allow for higher control on imaging parameters to provide better quality images. Lastly, an IMU and an OBD reader should be added to aid in improving car location and travelled distance estimates.

2.3.1 Sensor Selection and Integration

We used the TEMT6000 light sensor which has a resolution of 0.25 lux, a range of 0 – 250 lux and a sampling frequency of 10 – 50 Hz. For collecting video data, we used BC-EL630, which is a security camera with CCD sensors. We used a wide angle lens for a large field of view (~95.6 degrees). This particular camera was selected as it can operate effectively at low lighting conditions without the use of infrared and has a wide dynamic range that allows it to be used under varied lighting conditions. The system also has a power box that supplied 12 V DC to the cameras. During deployment, the power box was connected to a cigarette charger as before. The cameras came along with a DVR, which was used to record the videos during field experiments.

We used the 3D Robotics uBlox LEA6 GPS module which has an accuracy of around 2.5 m (circular error probability) and provides sampling frequency of 5 Hz. Furthermore, it has UART capabilities which allows easy integration with a microcontroller through a serial port for data logging. As an IMU, we used the UM6-LT Orientation Sensor from CH Robotics. It provides accelerometer and gyroscope measurements. The IMU samples the data internally at 1000 Hz and broadcast them.
at 20 – 300 Hz. We used an OBD-II UART from Sparkfun to interface with a car’s OBD-II bus providing car speed data. The module supports all major OBD-II standards, such as CAN and JBUS, and can be easily integrated with a microcontroller through its standard Rx/Tx ports. As a microcontroller, we chose Arduino Mega 2560 because of its speed, I/O capabilities, extensive open source documentation and code base and ease of use. We also used a standard 9 V battery pack for powering Arduino. The different sensors used in this prototype are shown in Fig. 2-7.

![Sensors and electronics](https://example.com/sensors.jpg)

(a) Light Sensor  (b) Camera with the wide angle lens  (c) GPS module  
(d) IMU  (e) OBD-II reader  (f) Arduino Mega 2560 (microcontroller)

Figure 2-7: Sensors and electronics in CASP. Courtesy: Google Images.

### 2.3.2 Car-top Platform

On the basis of the preliminary analysis, we decided that instead of mounting light sensors in different orientation, we would assemble them to only measure the vertical illumination. Furthermore, we decided to create an array of light sensors to measure spatial variability of illumination on a plane or the isolux contours, which when com-
bined with the lamp location information can help us to model the three-dimensional light field and estimate street level illumination. We designed a plexiglass board on which the light sensors were assembled in an $8 \times 2$ array as shown in Fig. 2-8a. The GPS, the IMU, the OBD-II sensor and the microcontroller were assembled on the plexiglass. As before, we fabricated aluminum supports to mount the cameras. In order to ensure coverage of the half-plane above the car roof, we mounted the cameras at angles $+30^\circ$ and $-30^\circ$ with respect to the vertical direction. Both the camera rig and the plexiglass were then attached to a roof rack which was then placed on top of a car. Fig. 2-8b shows the assembly of the sensor platform on top of a car. The DVR was placed inside the car and the BNC cables were passed through the window opening. Similarly, wires from the OBD-II reader were passed through the window opening to be connected to the car’s OBD port.

### 2.3.3 Data Logging and Management

Fig. 2-9 depicts the architecture of the data logging system [110]. We used the analog pins on Arduino Mega to read data from the light sensors and used the RX-TX pins to read the serial data coming from the IMU, OBD-II reader and GPS (digital). For the analog data, Arduino uses a 10-bit analog-to-digital converter to convert the voltage reading in the range $0 - 5$ V to a binary number. The other three sensors use an asynchronous serial protocol to communicate with the Arduino over the RX-TX ports. We used the documentation available for each of the three sensors to develop a computer program to parse the serial data coming from the sensors and extract the relevant sensor measurements. We used a microSD shield, an adapter that enables connecting a microSD card to the microcontroller, for direct logging of the sensor data as .txt files on the mounted microSD card. As seen in Fig. 2-9 the video data was recorded on the DVR.

We have data coming from different sensors and it is important to have a common frame of reference for their timestamps to enable multi-sensor data analytics. The digital sensors, microcontroller and DVR have internal clocks which may not be synchronized. We decided to use a laptop’s clock to provide a common time ref-
reference to all the sensors as shown in Fig. 2-9. At initialization we use a laptop to provide a reference time stamp to the DVR. Similarly, when the microcontroller is initialized, we send the current time of the laptop to the Arduino via USB through a program developed. Note that the two time stamps may have different values but are referenced with respect to the laptop’s clock. We then rely on the DVR’s and microcontroller’s internal clock to advance time stamps from the initial reference times. We use the Arduino’s time stamp when logging different sensors’ data. All sensor data on the Arduino are hence timestamped with respect to the laptop time just as the video on the DVR. This solution however requires a user to manually initialize both the Arduino and the DVR and then the data is collected autonomously during
The Arduino polls data from the different sensors sequentially and repeats the data collection loop throughout the field deployment as shown in Fig. 2-10. The data collection scheme is a polling system where the microcontroller sequentially looks for a new data packet from each sensor and if it is available, it reads it and transfers it to the microSD card. During testing, we identified that if the data packet is not available then moving to the next sensor reduces the idle time of the microcontroller and maximizes the overall sampling rate of the sensor system.

### 2.3.4 Field Studies

We tested our final prototype in Cambridge, MA USA, Malaga, Spain, Santander, Spain and Birmingham, UK. The prototype worked robustly and reliably. In our experiments we found that the GPS was logged at a sampling rate of ~ 1 Hz and the sampling rate had a standard deviation of ~ 25%. The other four sensors were logged at ~ 2.5 Hz and the rate had a standard deviation of < 20%. The total data rate for the Arduino microcontroller was ~ 1.71 MB/hr. The videos were recorded
at 25 fps and the data rate was $\sim 0.96$ GB/hr.

### 2.4 Conclusion

In this chapter, I discussed the development of CASP, a car-mounted autonomous street light scanning platform. The platform can be used for collecting data on street lighting infrastructure. After highlighting the importance of street light monitoring and the current challenges faced by the government and industry, I presented how a vehicle mounted sensor system can address these challenges and facilitate large-scale, reliable and accurate data collection on lighting infrastructure. I presented key considerations in developing such a sensor system and then discussed the design of the initial prototype, its testing and data collection. The lessons learned from field testing led us to identify important issues that need to be addressed for developing a standalone system capable of providing usable data.

Thereafter, I presented the design of the final prototype: sensor selection, assembly of the car-top system and data management. We tested the prototype in several cities and it was successful in collecting data reliably and robustly. I presented basic
analysis of the the collected data for example the data rate and sampling frequency, postponing the detailed discussion of the algorithms for analyzing street lighting data to Chapter 3.

I believe that mobile sensor systems have broad applications in managing urban and natural infrastructure, such as, street lighting, roads and bridges, building insulation, pollution plumes, chemical leaks and wild fires. I addressed critical challenges in system integration, such as, sensor selection, sensor assembly and data collection with the hope that the design methodology presented here will serve as a blueprint for developing future vehicle-mounted sensor systems.
Chapter 3

Analysis of Street Lighting Data

In this chapter I present the algorithms and the methods developed to analyze the data collected from the light scanning platform presented in Chapter 2. I first discuss a method to create luminosity maps by combining the light sensor measurements with the GPS measurements and display the results on an open sourced map such as Google Maps. I discuss how the car location and trajectory estimates can be improved by integrating the car speed data obtained by the OBD-II reader with the GPS data. I then present an algorithm for automating inventorying of street lamps by identifying lamps from the video data and combining that with the GPS data to geotag lamps. The lamp classification algorithm is a supervised learning method and I present the details of the classification studies. The results of lamp classification are then extended to develop a method for estimating the height of the lamps. I then introduce a framework for developing a three-dimensional light field model of street lighting and discuss how the model can be used to estimate street level illumination which is often required by the decision makers. I conclude the chapter with a summary of contributions and the problems addressed by the algorithms presented in this chapter. I motivate the content of the the next two chapters by highlighting some of the fundamental questions that emerge on the analysis of real world mobile sensor data. One needs to address these fundamental questions to understand the applicability and limits of mobile sensor systems.
3.1 Luminosity Maps

Our sensor platform collects illumination data through the light sensor array and the location data through GPS (Chapter 2). Overlaying the illumination data on a map will serve as a useful tool for the decision makers. I first developed a basic approach to make a grayscale illumination map as shown in Fig. 2-4. I take the average of the light sensor array measurements to compute the average illumination ($l_{avg}$). I then adopt a color scheme from black to gray to white as shown in Fig. 3-1. The slides are numbered from 1 to 5 and one of them is selected to be displayed on the map depending on the value of $l_{avg}$ as indicated in Fig. 3-1. We define $\phi$ as the latitude and $\lambda$ as the longitude. For every slide, $(\phi, \lambda)$ is obtained through the GPS sampled at 1 Hz and it specifies the center of the circle.

![Color scheme](image)

(a) Slide 1 ($0 \leq l_{avg} \leq 5$)  (b) Slide 2 ($5 < l_{avg} \leq 15$)  (c) Slide 3 ($15 < l_{avg} \leq 30$)

(d) Slide 4 ($30 < l_{avg} \leq 50$)  (e) Slide 5 ($50 < l_{avg}$)

Figure 3-1: Color scheme adopted to create the two-dimensional luminosity maps.

Fig. 3-2 shows another visualization method where vertical cylinders are used to indicate the lighting levels and the height of the cylinder is proportional to $l_{avg}$. Further, the visualization uses the jet colormap where low values of $l_{avg}$ are indicated by blue cylinders and high values are indicated by red. Like before, the location data is provided by the GPS and $(\phi, \lambda)$ specifies the center of the circle.
I developed both the visualization schemes in collaboration with Abraham M. Rosenfeld [110]. We wrote a library to combine light sensor measurements with the GPS data to generate .kml files which can be directly viewed in Google Earth or Google Maps. Chapter 2 Fig. 2-4 and Fig. 3-2 depict snapshots extracted from the .kml files displayed in Google Earth. As discussed in Chapter 2, Section 2.3.3, the different sensors on the sensor platform do not log data coincidentally, rather they are initialized with a common time stamp from the laptop. Hence, for example, there is a time lag between the light sensor data and the GPS data. To create luminosity maps, I created a common time grid and linearly interpolated $l_{avg}$, $\phi$ and $\lambda$ on it to obtain synchronized vectors of time, $l_{avg}$, $\phi$ and $\lambda$. Fig. 3-3 depicts the variation of average illumination ($l_{avg}$) with traveled distance for a particular road segment. An approximate periodic trend is observed in illumination which is indicative of a regular arrangement of street lamps at an interval of around 20 m. Though the above visualization schemes lack granularity, they provide useful information on average street illumination levels and correlate well with the street lamp location. The lower peaks may correspond to a damaged lamp and a semi-live update of the map in Fig. 3-3 can be used for deciding maintenance operations.

![Figure 3-2: Two examples of street lighting levels indicated by vertical cylinders.](image)

Several challenges need to be addressed in order to extend this work into a finer analysis of street lighting. For example, GPS data is often unreliable in urban driving
conditions and methods need to be developed to improve the location estimate before the light sensor measurements can be integrated with it for luminosity mapping. Furthermore, we would like to identify street lamps from the video data so that we can automate the development of a city-wide street lamp inventory. We can then integrate the location information of street lamps with the car-top light sensor measurements to make inferences about lamp luminosity and spatial distribution or contours of street illumination. In the rest of the chapter, I discuss the methods I have developed in this thesis to address the above problems.

3.2 Integrating OBD-II speed data with GPS measurements

GPS, which provides absolute geographic location, is typically used to obtain location information of driving vehicles and has an accuracy of a few meters [19]. Differential GPS can have an improved sub-meter accuracy [99]. GPS, in general, suffers from low reliability especially in urban driving conditions due to multipath effects and poor satellite signal [148]. Furthermore, as we discussed in Chapter 2 Section 2.3.4, we log the data from the GPS at a lower sampling rate as compared to the IMU and OBD-II scanner. A lower sampling rate limits the spatial resolution of location information. Hence in the context of our sensor platform, we need to develop robust methods for
using low frequency, noisy GPS data to accurately estimate vehicle trajectories.

Researchers have made significant progress in developing sensor fusion algorithms for combining GPS data with inertial measurement units (IMU). A typical approach involves using a vehicle motion model in combination with a Kalman filter to estimate states in the presence of noisy GPS and IMU data. Zhang et al. [148] compared the extended Kalman filter with the unscented Kalman filter for GPS/IMU integration. Carom et al. [28] included contextual information to define domains of valid sensor data and reject bad sensor measurements. Adaptive Kalman filtering [44] and factor graph based smoothing [71] have also been studied for vehicle trajectory estimation.

On-board-diagnostics (OBD) monitors engine health and vehicle components and allows services to be performed based on real-time and recorded events. Since 1996, all cars sold in the US were mandated to comply to OBD specifications [59]. An important vehicle motion parameter that can be obtained through OBD is speedometer data. Contrary to other vehicle network variables such as odometer readings, OBD data is accessible in a manufacturer-independent, standardized format. Furthermore, OBD scan tools have shown significant improvement in recent years and now have the capability of real time data logging through a serial port or wireless protocols such as bluetooth and wi-fi [7]. OBD scan tools have the potential for integrating speedometer data with the existing GPS hardware and even mobile phones to improve vehicle navigation solutions.

In this thesis, my objective is to study how the inclusion of OBD speed data can improve vehicle location estimation. I aim to robustly estimate driven distance and reconstruct a vehicle trajectory in the presence of (i) deliberate GPS downsampling and (ii) random GPS outage. To achieve this, I extend the GPS/IMU sensor fusion to include dead reckoning through the OBD speed data. This work was done in collaboration with Johannes Paefgen and the details of our approach can be found in [79].

Our work differs from prior research in two significant aspects:

1. we do not use map data for the estimation of driving parameters. The studies that leverage map data [122] are likely to achieve better estimation performance
than the presented approach, however, they require large storage and computational power making them a less practical option for low-cost embedded systems. From the point of post-processing, map matching has the potential to improve the results presented here.

2. we use accelerometers from an off-the-shelf IMU in combination with the OBD car speed data accessed via the standardized connector which makes our system straightforward to replicate independent of vehicle make or model.

### 3.2.1 Vehicle Trajectory Estimation

We first develop the equations of motion that form the basis for the Kalman filter-based sensor fusion. We model the car as a unicycle, a viable assumption when considering only linear vehicle dynamics [82] with the constraint that the vehicle moves along the trajectory with no slip. As shown in Fig. 3-4, $G$ is the frame fixed to the ground with respect to which we aim to estimate the position of the car $(X, Y)$. $B$ is the body frame attached to the vehicle and $\theta$ (heading) is the angle that $x_b$ makes with the $x$ axis.

![Simplified vehicle motion model in Cartesian coordinates](image)

Typically two drive controls are assumed, velocity, $v$, leading to forward motion and an angular velocity (rotation of the frame $B$), $\omega$, for turns. This yields the
following equations of motion:

\[ \begin{align*}
\dot{X} &= v \cos \theta, \\
\dot{Y} &= v \sin \theta, \\
\dot{\theta} &= \omega.
\end{align*} \tag{3.1} \]

Furthermore, we have the following no slip condition:

\[ \dot{Y} \cos \theta - \dot{X} \sin \theta = 0. \tag{3.2} \]

Let the acceleration measured in \( B \) be \( (a_{xb}, a_{yb}) \). Combining (3.1) and (3.2), we obtain the following equations:

\[ \begin{align*}
\dot{v} &= a_{xb}, \\
\dot{\theta} &= a_{yb} / v.
\end{align*} \tag{3.3} \]

As discussed before, we use three sensors in our study: GPS, IMU and OBD scan tool. Accelerometers on the IMU provide vehicle accelerations in the frame \( B \). Typically, low cost accelerometers, such as the one we use in our study, suffer from a significant scaling and bias error that can be modeled as:

\[ \begin{align*}
a_{xb} &= su_1 + b_1, \\
a_{yb} &= su_2 + b_2,
\end{align*} \tag{3.4} \]

where \( u_1 \) and \( u_2 \) are the accelerometer measurements along the axis \( x_b \) and \( y_b \) respectively, \( s \) is the accelerometer scaling factor and \( b_1 \) and \( b_2 \) are the accelerometer biases. We assume that the scaling factor \( s \) is the result of external parameters such as temperature fluctuations and may thus be assumed identical for both, lateral and longitudinal accelerations. Under this assumption, we can use OBD speed data to not only correct for accelerometer scaling in the direction \( x_b \), but also in \( y_b \). Furthermore, periods of vehicle stoppage as evident from OBD data allow for direct inference of
Accelerometer biases. GPS provides noisy latitude ($\phi$) and longitude ($\lambda$) estimate of the vehicle. Assuming the Earth to be spherical, we can convert ($\phi, \lambda$) to cartesian coordinates using the following equations:

$$X = R \cos(\lambda_m)(\phi - \phi_m),$$
$$Y = R(\phi - \phi_m),$$  \hspace{1cm} (3.5)

where $R$ is Earth’s radius, and $\phi_m = \sum_{i=1}^{N} \phi_i/N$ and $\lambda_m = \sum_{i=1}^{N} \lambda_i/N$ are the mean latitude and longitude averaged over a set of $N$ measurements. OBD scan tool measures vehicle speed along $x_b$ and is denoted by $v$.

**Discrete-Time State Space Model**

We represent the system by a state vector $\mathbf{x}(t) = (X, Y, d, v, \theta, b_1, b_2, s)$, where $d$ is an auxiliary state variable representing aggregated driven distance. Assuming bias and scaling errors as constant, we have the following continuous, non-linear state space model:

$$\mathbf{\dot{x}} = \begin{bmatrix} x_4 \cos x_5 \\ x_4 \sin x_5 \\ x_4 \\ x_8 u_1 + x_6 \\ (x_8 u_2 + x_7) / x_4 \\ 0_{3\times1} \end{bmatrix}.$$

Note that we include IMU readings as a system input, as they represent our best knowledge of the driver’s action. Furthermore, using IMU data as a measurement input would require an extension of (3.6) to include second-derivative terms.

Applying Euler’s Forward Method with step size $T$ to (3.6), we obtain the discrete
state update as:

\[ x(k + 1) = f(x(k), u(k)) \]

\[ = x(k) + T \begin{bmatrix} x_4(k) \cos x_5(k) \\
                      x_4(k) \sin x_5(k) \\
                      x_4(k) \\
                      x_8(k)u_1(k) + x_6(k) \\
                      x_8(k)u_2(k)+x_7(k) \\
                      0_{3 \times 1} \end{bmatrix} \] \hspace{1cm} (3.7)

Since the computation of changes in heading (\( \theta \)) becomes numerically unstable for velocities close to zero, we replace the corresponding state update equation with

\[ x_5(k + 1) = x_5(k) \text{ if } x_4(k) < v_{\text{min}} = 2 \text{ m/s}, \text{ i.e. assuming constant heading.} \]

Accelerometer readings \( u_1 \) and \( u_2 \) are affected by vehicle yaw and roll, i.e. rotations about the \( x_b \) and \( y_b \) axes. Such rotations are caused either by road inclination, or by load changes during cornering and vehicle acceleration. While in theory these dynamics could be explicitly modeled, we do not consider vehicle yaw and roll in our analysis. Doing so would require more sensing inputs and higher quality of measurements due to the increased number of states in the system. However, the simplified bias and scaling error model laid out above can account for changes in vehicle pose to some extent. When driving uphill, for instance, a constant gravity vector component of \( u_1 \) will be compensated for due to the discrepancy between \( u_1 \) and OBD readings.

Extended Kalman Filter with Hybrid Measurement Update

Since the dynamics of the system are non-linear, we implement a variant of the well-known discrete-time Extended Kalman filter (EKF) \cite{148}. For both, state update and measurement update, we assume zero-mean, additive noise that follows a multivariate Gaussian distribution. The EKF state estimate update then is

\[ \dot{x}(k + 1|k) = f(\dot{x}(k|k), u(k)). \] \hspace{1cm} (3.8)
The error covariance matrix $P(k|k)$ for state estimates $\hat{x}(k|k)$ is updated using the linearized state transition matrix:

$$F(k) = \frac{\partial f}{\partial \mathbf{x}} \bigg|_{\hat{x}(k|k), u(k)} = \begin{bmatrix}
\cos x_5 & -x_4 \sin x_5 & 0 & 0 & 0 \\
\sin x_5 & x_4 \cos x_5 & 0 & 0 & 0 \\
0_{5 \times 3} & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & u_1 \\
-x_6 u_2 + \frac{x_7^2}{x_4^2} & 0 & 0 & \frac{1}{x_4} & \frac{u_2}{x_4} \\
0_{3 \times 3} & 0_{3 \times 5}
\end{bmatrix},$$

(3.9)

evaluated at the current state estimate and IMU input vector $u$. In parametrizing the process noise covariance matrix $Q$, we chose comparatively high values for entries corresponding to steps involving updates of $u_1$ and $u_2$, while the bias and scaling error terms received low values as they vary only slowly over time. We assume independence among process noise and $Q$ to be a diagonal matrix.

We define the following measurement update vector

$$z(k) = h(\hat{x}(k|k-1)) = H\hat{x}(k|k-1)$$

(3.10)

which maps the system states to sensor observations. As stated before, our objective is to estimate the true vehicle trajectory with noisy and temporarily unavailable GPS measurements. Therefore, we use two different measurement vectors depending on the availability of GPS data:

$$\begin{align*}
\mathbf{z}(k) &= \begin{bmatrix} X \\ Y \\ v_x \end{bmatrix}, \\
\mathbf{z}(k) &= \begin{bmatrix} v_x \end{bmatrix}, \quad \text{with} \\
H &= \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}^T \\
\tilde{H} &= \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}^T
\end{align*}$$

(3.11)

56
In the absence of GPS measurements, our EKF implementation switches to $\hat{H}$ which renders the states $x_5$ (vehicle heading) and $x_7$ (lateral-direction IMU bias) unreachable. The measurement noise covariance matrix $R$, is parametrized according to specifications of the used sensors with the assumption of independence among measurement errors. If both GPS and OBD measurements are available, it is set to $R = \text{diag}(3^2, 3^2, 0.447^2)$, corresponding to a standard deviation of 3 m around the true position for GPS (according to the manufacturer specification).

We have the following set of update equations:

\[
\begin{align*}
\dot{x}(k+1|k) &= f(\hat{x}(k|k), u(k)) \\
P(k+1|k) &= F(k)P(k|k)F(k)^T + Q \\
\hat{y}(k+1) &= z(k+1) - h(\hat{x}(k+1|k)) \\
S(k+1) &= HP(k+1|k)H^T + R \\
K(k+1) &= P(k+1|k)H^T S(k+1)^{-1} \\
\dot{x}(k+1|k+1) &= \dot{x}(k+1|k) + K(k+1)\hat{y}(k+1) \\
P(k+1|k+1) &= (I - K(k+1)H)P(k+1|k).
\end{align*}
\] (3.12)

Note that we replace $z$ with $\hat{z}$ and $H$ with $\hat{H}$ in the absence of GPS measurements. We remark that this hybrid formulation is straightforward to adapt to further available measurements such as odometer, gyroscope, and magnetic field sensors by modifications analogous to (3.11).

### 3.2.2 Empirical Validation

**Procedure**

We used a Ford Focus hatchback for collecting data. Our sensor system included a low cost GPS (Qstarz BT-Q818XT), IMU (9 degree of freedom Razor IMU) and OBDLink SX scan tool [7]. Note that here we used the GPS from our initial prototype and we used an OBD scanner and an IMU similar to the ones in our final prototype which were tested before finalizing the IMU and the OBD scanner. We collected data
from a closed-loop route in Cambridge, MA with a total driving distance of around 3.35 km. All the sensors were integrated with a laptop to log data during driving. The GPS used Bluetooth to send data to the laptop. The IMU was connected to an Arduino Mega and attached to a roof rack placed on top of the car. The IMU was mounted with its x-axis aligned along $x_b$ and its y-axis along $y_b$. Through a USB cable, the Arduino and the OBD scan tool were connected to the laptop. We developed our own computer program to log data from the three sensors at 10 Hz.

To assess the impact of an intentionally reduced GPS sampling rate, we systematically downsampled the original GPS data collected at 10 Hz. The measurements vector hence includes unobserved GPS data and is addressed through the hybrid scheme discussed in Section 3.2.1. For each GPS sampling time, we execute an online EKF estimation implementing state updates at 10 Hz and store the resulting state trajectory.

We extended the analysis of deliberate GPS downsampling by emulating GPS unavailability. For every entry in the downsampled GPS-measurement vector, we determined its inclusion in the EKF estimation as the outcome of a Bernoulli trial with a certain probability of not observing GPS data (denoted by $p$). We chose $p = \{1/3, 1/2, 2/3\}$ and for each $p$ ran 50 simulations in order to obtain mean and standard deviation of the performance metrics to be discussed in the next subsection. We carried out all computations in Matlab.

**Performance Metrics**

We define two performance criteria for the proposed estimation technique. First, we computed the relative error in the driven distance estimate

$$
\epsilon_d = \frac{|\hat{x}_3(K) - x_3(K)|}{x_3(K)}
$$

(3.13)
where index $K$ denotes the final state. Secondly, we computed the root mean square error of position estimates

$$
\epsilon_r = \sqrt{\frac{1}{K} \sum_{k=1}^{K} [(\hat{x}_1(k) - x_1(k))^2 + (\hat{x}_2(k) - x_2(k))^2]}.
$$ (3.14)

In order to obtain the true states $x_1, x_2, x_3$ for error calculation, we compute a reference trajectory from the 10 Hz full resolution GPS data using our algorithm. These errors capture two important aspects of vehicle navigation solutions. The driven distance estimate is relevant to applications that are concerned with the extent of vehicle usage. The position-RMSE is relevant to our mobile sensor system applications where a precise reconstruction of the vehicle trajectory is needed to associate sensor measurements with geographic location. It is also a prerequisite to map matching techniques that fit the estimated positions into digital maps [108] which can be a possible future work.

As an estimate for traveled distance without the use of OBD data, we aggregate the distance increments between consecutive points of the downsampled GPS measurement vectors. For this distance estimate, we compute $\epsilon_d$ and compare it to the EKF generated error metric. When the speed data is unavailable, there is no obvious interpolation scheme between downsampled GPS measurements. The assumption of constant speed required for a simple linear interpolation to estimate locations at intermediate times does not appear justified. Hence we do not compute an estimate of $\epsilon_r$ when the OBD data is unavailable.

**Results**

First, we evaluate the evolution of scaling and bias estimates at 10Hz GPS sampling frequency as depicted in Fig. 3-5a. These states improve the accelerometer measurements to be included as an input in the EKF algorithm. Fig. 3-5b shows the integrated acceleration $u_1$ before and after application of the error model together with the observed OBD speed.

For the generation of sparse GPS measurement vectors, we select an exponentially
increasing series of integers from 1 to 500 as downsampling factors. For each factor, a new sample interval time $T_s$ is obtained for GPS measurements. We then use our hybrid EKF framework to estimate trajectories and evaluate performance metrics at 10Hz, switching between the measurement $\tilde{z}$ where GPS points are not available, and $z$ otherwise. Fig. 3-6a and 3-6b depict two examples of vehicle trajectory reconstructions at $T_s = 5$ s and 20 s, respectively, together with a $T_s = 0.1$ s reference trajectory in black. At a downsampling factor of 50 ($T_s = 5$ s) we achieve robust replication of the reference trajectory. At a downsampling factor of 200 ($T_s = 20$ s), a segment-wise distortion of the estimated trajectory is observed. These segments mostly exhibit errors in the vehicle heading estimate, while segment lengths show minor deviations.
trajectory reconstructions, $\epsilon_d$ is lower than 0.1% for $T_s < 1$ s and lower than 1% for $T_s < 50$ s. The aggregated distance increments exhibit a higher driven distance error. The U-shaped distribution of the errors can be attributed to GPS noise aggregation at low $T_s$ and the failure to capture sufficient path curvature details at high $T_s$. On the other hand, our hybrid EKF method provides a robust driven distance estimate that is largely unaffected by these sources of errors.

![Figure 3-7: Variation of $\epsilon_d$ with $T_s$](image)

Next, we evaluate the performance of the proposed algorithm under random omissions of GPS measurements. Fig. 3-8a shows that $\epsilon_r$ is acceptable up to $T_s = 1$ s, after which we observe a sharp increase in the RMSE of the vehicle position estimates. As expected, the error is higher for higher probabilities of GPS outage. Fig. 3-8b shows a similar trend, though the driven distance error is acceptable even for higher values of $T_s$. For driving scenarios where the GPS availability may vary across the route, this suggests an adaptive sampling strategy. Adjusting the GPS sampling rate in accordance with observed GPS outages will lead to higher accuracy in trajectory estimation.

We further reduce the base sampling rate of the Kalman filter (KF) estimation loop and study its effect on the performance metrics. This analysis involves IMU and OBD data at larger sampling times and the GPS data is again downsampled as before. Fig. 3-9 shows the performance metrics as contour plots in the GPS downsampling factor and the IMU/OBD/KF sampling time space. Counter to intuition, contour plots differ between the two errors. We observe the trajectory deviation error, $\epsilon_r$, to
increase with the GPS downsampling factor and the error increases faster for larger KF sampling time (Fig. 3-9a). The relative driven distance error, $\epsilon_d$, is robust to GPS downsampling for lower KF sampling times but at larger KF sampling times the error is higher for lower GPS downsampling and decreases with higher GPS downsampling (Fig. 3-9b). At larger KF sampling times, we have sparse measurements from all the sensors. The driven distance estimate through EKF is better when filtering is performed at higher GPS downsampling factor (the trajectory degenerates to a straight line) as compared to filtering at lower GPS downsampling factor which increases the error by causing the trajectory to fit through noisy GPS measurements.

Figure 3-8: Comparison of the performance metrics for different GPS outage probability: (a) $\epsilon_r$ and (b) $\epsilon_d$. The solid lines indicate the mean of the metric while the dashed lines indicate mean +/- $\frac{1}{2}$ standard deviation.

Figure 3-9: Contour plots of the performance metrics (a) $\epsilon_r$ (m) and (b) $\epsilon_d$ (%) with the variation in GPS downsampling factor and IMU/OBD/KF sampling time.
3.2.3 Summary

In this section of my thesis, I investigated the integration of OBD speed measurements into vehicle navigation solutions to handle sparse and/or missing GPS data. I evaluated an extended Kalman filter for the reconstruction of a vehicle trajectory and the driven distance in the presence of sparse GPS data. By including bias and scaling error of the IMU measurements in the state space model, the performance of the estimator was considerably improved. As expected, the inclusion of OBD data enabled consistent estimates of the driven distance even at GPS sampling rates below 0.02 Hz. A comparison with the aggregated GPS distance increments demonstrated that the estimated distance suffered neither from the integration of errors at high sampling frequencies, nor from the exclusion of path details at low sampling frequencies. The estimator was also able to maintain low deviations from a reference trajectory in terms of root mean squared position errors. I showed that for random GPS sample omissions of up to two thirds of the samples, the error remained below 4 m for GPS sampling rates up to 1 Hz. Furthermore, I demonstrated different trends in the trajectory and the driven distance estimation error with the variation in sampling time of the Kalman filter and the GPS downsampling factor.

For future work, we would like to apply our methodology on different datasets collected with varied sensors and driving conditions. Also modeling the influence of road surface characteristics and inclination on the equations of motion can help improve location estimates. Furthermore, the hybrid measurement update model can be easily adapted to other sensor measurements such as gyroscopes or magnetic field sensors. Similarly, while speed is a reliable OBD measurement across vehicle makes and models, other less-standardized OBD parameters such as throttle and steering application can also be assumed to significantly improve trajectory estimation. Unscented Kalman filter could be explored as an alternative to EKF for filtering the non-linear state space model. The implemented EKF may further be refined by a non-additive Gaussian noise model.

OBD speedometer suffers from sources of errors such as wheel slip, unknown
tire pressure and tire radius. Explicitly modeling these uncertainties and estimating them similar to the accelerometer scaling and bias can improve the performance of the filter. As discussed in Chapter 2 Section 2.3.4, the GPS is logged at a lower sampling rate ($\sim 1$ Hz) compared to the OBD-II reader and IMU ($\sim 2.5$ Hz). The methodology presented here can help in obtaining car location estimates on a finer grid for improving the spatial resolution of luminosity maps. Furthermore, Fig. 3-7 shows that the presented method provides a more accurate estimate of the distance traveled by the car which can directly improve the analysis shown in Fig. 3-3.

3.3 Street Lamp Identification

One of the goals of the Ferrovial light scanning project was to develop a framework for creating city-wide inventory of street lamps. This goal can be achieved by identifying street lamps in the video data collected by our light scanning platform and combining this video data with the car location data to appropriately geotag street lamps. Fig. 3-10 shows example images collected by our mobile light scanning platform. As one expects, night time images comprise either of dark pixels or saturated pixels. Furthermore, the images suffer from bloom or glow which produces fringes of light extending from bright objects into the dark regions of the image and reduces the contrast and sharpness (i.e., presence of edges). The aforementioned reasons make it hard to implement object identification algorithms.

![Figure 3-10: Example images collected by the mobile sensor platform.](image-url)
As seen in Fig. 3-10, the bright, saturated regions in the images may also correspond to non-street lamps such as lit windows and doors and we need to differentiate them from the street lamps. We minimize imaging headlights of other cars (except at intersections where the cars on the roads perpendicular to the driving direction may come into view) by ensuring that the field of view of the camera system is above the car roof. Essentially, the goal was to develop a two-class classification technique which can identify street lamps from other bright features in the background when given an image.

Machine learning has emerged as an important scientific discipline that seeks to develop algorithms and methods that can learn from the data. A more formal, widely quoted definition of machine learning is: A computer program is said to learn from experience $E$ with respect to some class of tasks $T$ and performance measure $P$, if its performance at tasks in $T$, as measured by $P$, improves with experience $E$ [98]. The core components of a machine learning task involve hypothesis representation and generalization to unseen data. Broadly speaking, learning algorithms can be divided into supervised and unsupervised framework. As the name suggests, a labeled training set is used to learn an input-output relationship in supervised learning while the latter uses an unlabeled training set to discover inherent structures or groups in the data [24].

Image classification has attracted significant attention in the computer vision community with applications in the areas of image retrieval, object recognition and scene classification. Image classification is essentially a machine learning task that aims to learn a map from the training set that can then be used to classify new images. Fig. 3-11 depicts the various steps involved in the classification of images. Here the goal is to correctly identify the category to which an image belongs to, for example, does the image have a car in it? There are several challenges to images classification: the same object can be imaged with different viewpoints, illumination, occlusion, scale, deformation and background clutter [48]. Significant advancements have been made in the classification of images to identify human faces [132, 139, 149, 133, 88] which has found applications in the areas of surveillance and social networking. Other ap-
plication areas that have seen significant attention are text recognition [16], [17], [46] and car, motorbike or aeroplane detection [115], [11], [131].

![Diagram of Image Classification Process]

**Figure 3-11: Aspects of Image Classification**

The first task involves identifying an appropriate representation of the object we aim to identify in the images. Essentially, we need to convert the image to a feature vector on which a classification model can be learned. The feature vector should have discriminative power and at the same time should be robust to changes in pose, illumination, scaling and background clutter. Researchers have explored various object representation and feature selection schemes such as edge descriptors, part and structure models, constellation models, shape features, histogram of oriented gradients, bag of visual words, etc. References [128], [48], [147] and the references therein provide a detailed discussion on the various feature selection techniques for image classification.

Learning approaches can be divided into generative and discriminative methods. In a generative model a joint distribution of the feature vector \( (x) \) and the class label \( (c) \) is learned as \( p(c, x) \) while in a discriminative model a conditional distribution \( p(c|x) \) or an input-output relationship \( c = f(x) \) is learned. Generative models (e.g. mixture models, naive Bayes, topic models) and discriminative models (e.g. support vector machines (SVM), nearest neighbors, conditional random fields and neural networks) have been used for image classification applications [128], [48].

The advancements in computer vision and image classification techniques have
encouraged their use in vision based environmental monitoring applications. In the context of this thesis, my goal is to study the applicability of machine learning based image classification techniques to detect street lamps from the collected video data which when combined with car location allows automated inventorying of street lamps. Much of the work in object recognition has focused on detecting faces, cars, chairs and other objects that have well defined edges and corners. The edge and corner sets form natural descriptors of the objects aiding in developing effective classification techniques. As seen in Fig. 3-10, relevant corners and edges that can aid in classifying street lamps are difficult to identify. As mentioned before, the pixels corresponding to street lamps are saturated and a direct first step towards identifying the regions of an image that may correspond to a street lamp is to apply a thresholding-based image segmentation. I first converted every color image $I_{N \times M \times 3}$ to a grayscale image $G_{N \times M}$ where every pixel has a value $G_{i,j} \in [0, 1]$. The grayscale image was then converted to a black and white image using thresholding segmentation where every pixel with $G_{i,j} \leq t$ was mapped to zero and every pixel with $G_{i,j} > t$ was mapped to 1. I chose the threshold to be $t = 0.975$ and denoted the binary image as $B_{N \times M}$. I further removed from the binary image all the bright objects that have less than 200 pixels as we are interested in street lamps which were experimentally observed to be larger than 200 pixels. Fig 3-12 shows three examples of the original image (first row) and the converted binary image (second row).

A bounding box was drawn around each bright region and the corresponding region in the original image ($I_{N \times M \times 3}$) was cropped and saved as a possible street lamp candidate. My goal was then to develop a classification technique that can detect street lamps from the cropped image set. Fig. 3-13 shows example lamp images and Fig. 3-14 shows examples of non-lamp bright objects in the scene that I aim to correctly differentiate from the lamps. I observed that there is a significant variation in shape, intensity distribution and background noise in the set of lamp images. When comparing Fig. 3-13 with Fig. 3-14, I observed the lamps are primarily oval in shape while the non-lamp objects are primarily rectangular in shape. Furthermore, I observed that there is a difference in the nature of the spatial distribution of intensity.
Figure 3-12: For the original images in the top row the bottom row contains the corresponding binary image.

gradients in the regions surrounding the saturated pixels between lamp and non-lamp objects. Note that identifying an appropriate edge or corner set representation of the lamp objects due to both the variation in shape as well the bloom effect is hard.

3.3.1 Feature Selection

Feature selection or representing the image in an appropriate feature space is a critical component of creating an object recognition algorithm. In this section I discuss the construction of the feature set. Note that Matlab was used for all computations.
The basic approach was to construct different categories of feature sets and then concatenate them to form a large feature vector. Dimension reduction techniques were then used to find a good subset of features from the large feature vector that have high discriminative power. The large feature vector was composed of the following categories:

- **Normalized Histogram of Grayscale Intensity**: I converted every cropped image to its grayscale version with pixel values in the range $[0, 1]$. I then calculated the histogram of the grayscale intensity with the following bin centers: $[0.05 : 0.1 : 0.95]^T$ and the bin size of 0.1 giving a 10 dimensional feature vector. Fig. 3-15 shows an example of different histograms computed for a lamp and a non-lamp image.

- **Size and Total Intensity**: 2 additional feature vectors included the number of pixels in the cropped images and the sum of the grayscale intensity.

Figure 3-13: Example images of lamps.
Figure 3-14: Example images of other bright objects in the scene.

Figure 3-15: Normalized histogram of grayscale intensity.

- **Shape Parameters**: I converted each cropped image to its corresponding black and white image by using thresholding segmentation with a threshold value of 0.975. Matlab allows direct computation of several shape parameters using the function “regionprops” and I used the following:

1. Number of Edges in the Convex Hull of the region: Matlab computes the Convex Hull of the black and white image which is a $p \times 2$ matrix that
specifies the smallest convex polygon that can contain the region. Each row of the matrix contains the \((x, y)\) coordinates of one vertex of the polygon. I took \(p\) as a feature vector which corresponds to the number of edges in the enclosing convex polygon of the region.

2. Eccentricity: Matlab computes an ellipse for the region which has the same second-moment as the region and calculates its eccentricity \((e)\). The eccentricity is the ratio of the distance between the foci of the ellipse and its major axis length and \(e \in [0, 1]\).

3. Euler Number: A scalar which specifies the difference between the number of objects in the image and the number of holes in the image. The rationale behind using the Euler number is that we expect lamp images to have no holes but other non-lamp objects may have holes within the bright regions. As shown in Fig. 3-16, the black and white image of a non-lamp object has black pixels within surrounding white pixels.

![Figure 3-16: Black and white binary image.](image)

(a) A lamp object  (b) A non-lamp object

4. Extent: The ratio of the total number of pixels in the region to the total number of pixels in the bounding box which is the smallest rectangle containing the bright/white pixels.

- **Histogram of Oriented Gradients (HOG):** Feature descriptors that count occurrences of gradient orientation in localized regions of an image. These descriptors have found good success in object recognition [35, 150, 113, 67, 38]. The premise for the HOG descriptors is that local object appearance and shape
can be described by the distribution of intensity gradients or edge directions. One way of obtaining these descriptors is to divide the image into small overlapping regions, called cells, and for each cell accumulate a weighted local one-dimensional histogram of gradient directions or edge orientations for the pixels within the cell. To compute the HOG descriptors I converted every cropped image to its grayscale version and then rescaled them to a standard template size of $50 \times 50$ using a bilinear transformation. I then chose to divide the image into $3 \times 3$ overlapping cells of pixel dimensions $24 \times 24$. I computed the $x$ and the $y$ derivative of the image intensity ($I_x$ and $I_y$) using the centered derivative mask $[1, 0, -1]$ and $[1, 0, -1]^T$ respectively. The gradient orientation was then calculated by $\theta = \tan^{-1}\left(\frac{I_y}{I_x}\right)$. The angular range of $(\pi, \pi]$ was divided into 18 discrete steps each of size $20^\circ$. The histogram for every cell was calculated by identifying all the pixels in the cell whose $\theta$ belong to a certain $20^\circ$ bin interval and assigning the magnitude of the gradient, i.e., $\sqrt{I_x^2 + I_y^2}$ as the contribution of the pixel for that bin. Hence for every cell, an 18 dimensional histogram vector ($h$) was created which was then contrast normalized by dividing the vector by $\|h\| + 0.01$, where $\|.|\|$ is the $\ell_2$ norm of the vector. Combining the histograms from all the cells results in a 162 dimensional feature vector that represents the HOG.

- **Pixel Grayscale Intensity Values:** I converted every cropped image into a standard $50 \times 50$ template using a bilinear transformation, converted them to grayscale and concatenated the grayscale pixel intensity values to form a 2500 dimensional feature vector.

After concatenating all the feature vectors obtained above, I formed a 2678 dimensional feature vector. Feature selection (aka variable reduction) is an important step towards developing a robust classifier. Feature selection has been studied extensively in the machine learning community and some of its key potential benefits include: a better understanding and visualization of the data, faster training and developing computationally efficient classifiers and reducing the curse of dimensionality and pro-
viding better prediction performance [65, 114, 105, 47, 104, 58]. The feature selection techniques can be broadly divided into three different categories:

1. **Wrappers:** The wrapper methodology involves using the prediction performance of a black box learning algorithm to assess the relative usefulness of subsets of variables. Typically, a greedy search strategy is employed to search the space of variable subsets and the cross-validation performance is optimized to guide the search. These methods are computationally expensive and are highly likely to get stuck at local optima. Stepwise regression is an example of a wrapper method.

2. **Filters:** These methods are employed as a preprocessing step and computed independently of the choice of the learning algorithm. Their popularity is attributed to their simplicity, computational and statistical scalability, speed and robustness to overfitting. Filters typically rank the individual features on the basis of their capability to differentiate the classes and then the top ranking features are selected in the learning model. Fisher score, Relief score and Information Gain are examples of filter methods for feature selection.

3. **Embedded Methods:** In this methodology, variable selection is performed by directly optimizing a two-part objective function with a goodness-of-fit term and a penalty for a large number of variables (regularization term). Such techniques work effectively for linear models, for example \( \ell_1 \) regularized linear SVM, but can be suboptimal for non-linear prediction models. Furthermore, these methods are computationally expensive when dealing with a large number of variables. Sparse regression (LASSO) and regularized trees are other examples of embedded methods.

The constructed image feature vector \( f_i \in \mathbb{R}^{2678} \). The data set comprised of 13483 cropped images out of which 1689 were images of lamps. The preceding ratio of the number of lamps to non-lamps was the natural distribution obtained when the original video frames were cropped. Out of the 11794 non-lamp images 3288 were
images of the texts added by the camera on the images (Fig. 3-17). On the basis of
the observation that the camera text is consistent, I decided to develop a three-class
classification technique where I first aim to classify the cropped image as a camera
text or the “others” (one-vs-all strategy for multi-class classification). If the image is
classified as the “others” I then employ a two-class classification technique to identify
if it is a lamp or a non-lamp. Essentially, I developed sequential binary classifiers to
achieve the task of identifying street lamps from the cropped images.

Figure 3-17: Example texts added by the camera on the images.

In this thesis, I employed filtering-based feature selection techniques to reduce the
dimension of the feature space because of their speed and computational efficiency.
As mentioned before, at every stage I solve a binary classification problem. Many
filter based feature selection techniques are supervised and use the class labels to
weigh the importance of individual variables. I use Fisher score and Relief score as
supervised filters and principal component analysis (PCA) as an unsupervised filter.

- The Fisher score of a feature for binary classification is defined as

\[
FS(f_i) = \frac{n_1(\mu_1^i - \mu_i)^2 + n_2(\mu_2^i - \mu_i)^2}{n_1(\sigma_1^i)^2 + n_2(\sigma_2^i)^2}
= \frac{1}{n_1} \left( \frac{(\mu_1^i - \mu_2^i)^2}{(\sigma_1^i)^2} + \frac{(\sigma_2^i)^2}{n_2} \right),
\]

where \( n_j \) is the number of samples belonging to class \( j \), \( n = n_1 + n_2 \), \( \mu_i \) is the
mean of the feature \( f_i \), \( \mu_j^i \) and \( \sigma_j^i \) are the mean and the standard deviation of
\( f_i \) in class \( j \). The larger the score, the higher the discriminating power of the
variable is. I calculated the Fisher score for every variable and ranked them in
a decreasing order. I then selected the subset whose weights are in the top \( p \% \) as the reduced feature set to be used in the learning algorithms.

- The Relief score of a feature is calculated by first randomly sampling \( m \) instances from the data and using the score:

\[
RS(f_i) = \frac{1}{2} \sum_{k=1}^{m} d(f^i_k - f^i_{NM(x_k)}) - d(f^i_k - f^i_{NH(x_k)}),
\]

where \( f^i_k \) denotes the value of the sample \( x_k \) on the feature \( f_i \), \( f^i_{NM(x_k)} \) and \( f^i_{NH(x_k)} \) denote the values of the nearest points to \( x_k \) on the feature \( f_i \) with the same and different class label respectively, and \( d(.) \) is a distance measure which I chose to be the \( \ell_2 \) norm. The larger the score, the higher the discriminating power of the variable is. Again, I calculated the Relief score for every variable and ranked them in a decreasing order. I then selected the subset whose weights are in the top \( q \% \) as the reduced feature set.

- Principal component analysis (PCA) is a feature transformation technique that employs an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components. They are defined in such a way that the first component accounts for the largest variance in the data and each succeeding component in turn has the highest variance possible under the constraint that it is orthogonal to (or uncorrelated with) the preceding components. PCA is effective in dimensionality reduction as the first few components can explain a large variance of the observation matrix. Given the observation matrix \( X \), PCA computes a score matrix \( S \) which represents the mean subtracted observation matrix in the latent space \( L \) through:

\[
S_{N \times M} = (X - X_{avg})_{N \times M}L_{M \times M}.
\]

If we chose a \( k \)-dimensional subspace to represent our observations, using the first \( k \) columns of \( L \) (\( L_k \)) allows the optimal representation of the observation.
in the \( k \)-dimensional subspace as:

\[
\hat{S}_k = (X - X_{\text{avg}})L_k.
\]  

(3.18)

Increasing \( k \) increases the percentage variance explained by the reduced subspace \((v_k)\) and we select \( k \) according to the criterion \( \frac{\text{var}_k}{\text{var}_M} \leq r \), where \( r \in [0, 1] \).

### 3.3.2 Learning and Optimizing a Classifier

As discussed in the previous section, I employed three different filtering techniques for dimensions reduction. Also the subsets selected by the three methods described in the previous section depend on the parameters \( p, q \) and \( r \) respectively. I used three different types of classifiers: Naive Bayes classifier, discriminant analysis and support vector machines.

- **Naive Bayes classifier** assumes that every feature \( f_i \) is conditionally independent of every other feature \( f_j \) for \( j \neq i \) given the class label \( y \) [86], [109], [27]. The conditional probability model of a class label is then defined as:

\[
p(y|f_1, f_2, \ldots, f_M) = p(y) \prod_{i=1}^{M} p(f_i|y).
\]  

(3.19)

The conditional distribution of a feature \((p(f_i|y))\) can be modeled as a Gaussian distribution or a nonparametric density estimation technique can be used that typically employs kernel smoothing.

- **Discriminant analysis** assumes that the conditional probability density functions \( p(f|y = 0) \) and \( p(f|y = 1) \) are both normally distributed with mean and covariance parameters \((\mu_0, \Sigma_0)\) and \((\mu_1, \Sigma_1)\) respectively [70], [97]. Under this assumption, the solution to the classification problem is the class that maximizes the conditional likelihood.

- **Support vector machines (SVMs)** constructs a hyperplane or a set of hyperplanes in the feature space (or the transformed feature space) which can be used for
classification. The hyperplane which has the largest distance to the nearest training data point of any class (or the margin) leads to a good separation of the classes and has a good generalization capability [25], [30], [101]. It is also termed as the maximum-margin classifier and can also perform non-linear classification through the kernel trick.

I divided the data set into a training set and a test set where the test set had around 10% of the samples from each of the three classes. I denote the set of observations in the training set as $f_T$ and the set of observations in the test set as $f_t$. The following details the steps involved in the development of the sequential binary classifiers:

- **Identifying Camera Text**: Given the training data set $f_T$, I first constructed class labels, assigning 1 to examples that were camera text and 0 to the ones that belonged to the “others” category. As discussed in Section 3.3.1, choosing different values of $p$, $q$ and $r$ leads to different sizes of the selected variable sets ($|S|$) as enumerated in Table 3.1.

<table>
<thead>
<tr>
<th>Fisher Score</th>
<th>Relief Score</th>
<th>PCA</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p$</td>
<td>$</td>
<td>S</td>
</tr>
<tr>
<td>0.85</td>
<td>286</td>
<td>0.85</td>
</tr>
<tr>
<td>0.75</td>
<td>31</td>
<td>0.75</td>
</tr>
<tr>
<td>0.50</td>
<td>13</td>
<td>0.50</td>
</tr>
<tr>
<td>0.25</td>
<td>6</td>
<td>0.25</td>
</tr>
<tr>
<td>0.10</td>
<td>2</td>
<td>0.10</td>
</tr>
</tbody>
</table>

Table 3.1: Number of variables $|S|$ in the selected variable subset.

Matlab has various implementations of each of the three classifiers and I studied performance with the different implementations. For the Naive Bayes classifier there are various options to model the conditional distribution of a feature $p(f_i|y)$: normal distribution, kernel smoothing density estimate, where the kernel can be normal, box, triangle or epanechnikov. Hence there are five different implementations of the Naive Bayes classifier.

Similarly, Matlab allows six different types of discriminant analysis classifier and each of them differs from the other in the way the covariance matrix and
its inverse is computed. The method ‘linear’ assumes $\Sigma_0 = \Sigma_1$, i.e., one covariance matrix for all the classes. The option ‘quadratic’ computes two different covariance matrices, one for each class. The methods ‘diagLinear’ and ‘diagQuadratic’ use only the diagonal of the ‘linear’ and the ‘quadratic’ covariance matrices respectively while the methods ‘pseudoLinear’ and ‘pseudoQuadratic’ use the pseudo inverse of the ‘linear’ and the ‘quadratic’ covariance matrices if necessary (when the covariance matrices are singular).

The concept of soft margin is typically used while developing an SVM classifier. In many cases the observations are not linearly separable and the soft margin classifier allows for certain observations to be within the margin or even misclassified while still maximizing the margin. This technique has been found to be robust to outliers. The hyper parameter $C$ which is also called the box constraint controls the degree of slackness in the SVM formulation [25]. SVMs can also learn a non-linear classifier using the kernel trick. The different kernel functions available in Matlab that I used are: linear, quadratic, polynomial of degree 3 and Gaussian radial basis function (rbf) with a scaling factor $\gamma$.

To find the best method and the parameters of a classifier, I employed 10-fold cross-validation as the criterion. For a given method and a set of parameters, every classifier was trained on each of the 9-folds of the training data and then tested on the remaining 1-fold of the training observations to compute the misclassification rate. For the Fisher score based variable selection method, with $p = 0.75$, Table 3.2 enumerates the 10-fold misclassification rate for the different classifiers. For SVMs, the optimal choice of $C$ and $\gamma$ (in the case of rbf kernel) was done by doing a grid search on the discrete set $C = [0.005, 0.01, 0.1, 1, 10, 50]^T$ and $\gamma = [0.005, 0.01, 0.1, 1, 10, 50]^T$ and choosing the parameter that gave the minimum 10-fold cross-validated misclassification rate. We observed that the SVMs clearly outperforms the Naive Bayes and the discriminant analysis classifiers.

From the grid search I found that the best values of $C$ are 0.1 for the linear
Fisher Score based feature selection, $p = 0.75$

<table>
<thead>
<tr>
<th>Gaussian distribution</th>
<th>Kernel smoothing density estimate</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Gaussian kernel</td>
<td>Box kernel</td>
<td>Triangle kernel</td>
<td>Epanechnikov kernel</td>
</tr>
<tr>
<td>2.6997</td>
<td>1.4611</td>
<td>2.1657</td>
<td>1.8319</td>
<td>1.8987</td>
</tr>
</tbody>
</table>

Discriminant Analysis

<table>
<thead>
<tr>
<th>Linear</th>
<th>Quadratic</th>
<th>Diagonal</th>
<th>Diagonal</th>
<th>Pseudo</th>
<th>Pseudo</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Linear</td>
<td>Quadratic</td>
<td>Linear</td>
<td>Quadratic</td>
</tr>
<tr>
<td>1.4537</td>
<td>2.7516</td>
<td>1.1422</td>
<td>2.6997</td>
<td>1.4537</td>
<td>2.7516</td>
</tr>
</tbody>
</table>

Support Vector Machines

<table>
<thead>
<tr>
<th>Linear</th>
<th>Quadratic</th>
<th>Polynomial (order 3)</th>
<th>Radial Basis Function</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.0235</td>
<td>0.9419</td>
<td>1.2089</td>
<td>0.8233</td>
</tr>
</tbody>
</table>

Table 3.2: 10-fold misclassification rate in %.

kernel, 0.005 for the quadratic and the polynomial kernel and 50 for the radial basis function kernel and $\gamma = 10$ gave the minimum cross-validation error. Fig. 3-18 plots the variation of the 10-fold misclassification rate and the size of the selected variable set with $p$. Typically, a “U-shaped” curve is expected of the misclassification rate with the size of the feature set, where a high number of features lead to overfitting and higher cross-validation error while very low number of features miss relevant features that have good discriminative power. My simulations show a similar trend and comparing with the size of the selected variable set (Fig. 3-18b), I chose $p = 0.75$ as a good choice of the parameter for Fisher score based feature selection.

I then tested the SVM classifiers on the test set (ft) and found that the minimum misclassification rate of 0.2849% was achieved by both the linear and the radial basis function kernel. Owing to its simplicity, I chose the linear kernel SVM over the radial basis function kernel SVM with $C = 0.1$.

For the Relief score based feature selection, Table 3.3 enumerates the 10-fold misclassification rate with the different classifiers and $q = 0.75$. The table depicts the SVMs whose best parameters were selected from the grid search. I conducted an analysis similar to Fig. 3-18 and found that $q = 0.75$ is a good
Figure 3-18: (a) Variation of 10-fold misclassification rate with $p$ and (b) Variation of size of the variable set selected by the Fisher score ranking with $p$.

choice for Relief score based feature selection. I again observed that the SVMs outperform the other classifiers and when they were compared on the test set the linear kernel SVM gave the minimum misclassification rate of 0.2849%.

<table>
<thead>
<tr>
<th>Relief Score based feature selection, $q = 0.75$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naive Bayes Classifier</td>
</tr>
<tr>
<td>Gaussian distribution</td>
</tr>
<tr>
<td>Kernel smoothing density estimate</td>
</tr>
<tr>
<td>Gaussian kernel</td>
</tr>
<tr>
<td>Box kernel</td>
</tr>
<tr>
<td>Triangle kernel</td>
</tr>
<tr>
<td>Epanechnikov kernel</td>
</tr>
<tr>
<td>1.7949</td>
</tr>
<tr>
<td>1.4759</td>
</tr>
<tr>
<td>1.3573</td>
</tr>
<tr>
<td>1.3498</td>
</tr>
<tr>
<td>1.3573</td>
</tr>
<tr>
<td>Discriminant Analysis</td>
</tr>
<tr>
<td>Linear</td>
</tr>
<tr>
<td>Quadratic</td>
</tr>
<tr>
<td>Diagonal</td>
</tr>
<tr>
<td>Diagonal</td>
</tr>
<tr>
<td>Pseudo</td>
</tr>
<tr>
<td>Pseudo</td>
</tr>
<tr>
<td>Linear</td>
</tr>
<tr>
<td>Quadratic</td>
</tr>
<tr>
<td>Linear</td>
</tr>
<tr>
<td>Quadratic</td>
</tr>
<tr>
<td>Linear</td>
</tr>
<tr>
<td>Linear</td>
</tr>
<tr>
<td>1.5872</td>
</tr>
<tr>
<td>4.0570</td>
</tr>
<tr>
<td>1.4537</td>
</tr>
<tr>
<td>1.7949</td>
</tr>
<tr>
<td>1.5872</td>
</tr>
<tr>
<td>4.0570</td>
</tr>
<tr>
<td>Support Vector Machines</td>
</tr>
<tr>
<td>Linear</td>
</tr>
<tr>
<td>Quadratic</td>
</tr>
<tr>
<td>Polynomial (order 3)</td>
</tr>
<tr>
<td>Radial Basis Function</td>
</tr>
<tr>
<td>1.0013</td>
</tr>
<tr>
<td>0.9938</td>
</tr>
<tr>
<td>0.9938</td>
</tr>
<tr>
<td>0.9048</td>
</tr>
<tr>
<td>0.9048</td>
</tr>
</tbody>
</table>
| Table 3.3: 10-fold misclassification rate in %.

The 10-fold misclassification rate with PCA based feature selection is depicted in Table 3.4 for $r = 0.75$. Similar to the analysis presented before, I found $r = 0.75$ to be a good choice. On the test set I found that the linear kernel SVM gave the minimum misclassification rate of 0.2849%.

From the above discussion, the best linear SVM selected by either of the three feature selection methods gives 0.2849% misclassification rate on the test set.
Principal Component Analysis based feature selection, \( r = 0.75 \)

<table>
<thead>
<tr>
<th>Gaussian distribution</th>
<th>Naive Bayes Classifier</th>
<th>Kernel smoothing density estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Gaussian kernel</td>
</tr>
<tr>
<td>7.0014</td>
<td>5.5700</td>
<td>6.2078</td>
</tr>
</tbody>
</table>

**Discriminant Analysis**

<table>
<thead>
<tr>
<th>Linear</th>
<th>Quadratic</th>
<th>Diagonal Linear</th>
<th>Diagonal Quadratic</th>
<th>Pseudo Linear</th>
<th>Pseudo Quadratic</th>
</tr>
</thead>
</table>

**Support Vector Machines**

<table>
<thead>
<tr>
<th>Linear</th>
<th>Quadratic</th>
<th>Polynomial (order 3)</th>
<th>Radial Basis Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.4537</td>
<td>2.9296</td>
<td>2.9148</td>
<td>1.1348</td>
</tr>
</tbody>
</table>

Table 3.4: 10-fold misclassification rate in %.

I selected all three of them as the final set of classifiers for identifying camera texts annotated on the images. The final prediction was made on the basis of a simple majority vote across the three classifiers. Note that in all the three cases the best value of the box margin parameter, \( C \), was found to be 0.1.

- **Identifying Street Lamps**: In this section I discuss the development of a two-class classifier to identify lamps and non-lamps from the set “others”. I followed a process similar to the one discussed above. Firstly, I identified the best parameters for the different feature selection methods by comparing the 10-fold misclassification rate and then for the best parameter I selected the classifier that gave the minimum misclassification error on the test set as the final classifier. Though, in this case, I observed that the PCA based feature selection did not work as the first principal component explained around 99.6% of the total variance and when I used this component to train the classifiers, I observed high misclassification rates (~ 15%).

For the Fisher score based feature selection, I found that \( p = 0.75 \) was a good choice and it led to a variable set of size 27. When the classifiers were used on the test set, the minimum misclassification rate of 2.793% was achieved by the quadratic SVM while the minimum false positive rate of 1.223% was achieved.
by the rbf SVM. Note that the misclassification rate with rbf SVM was 8.379% while the quadratic SVM had a false positive rate of 5.228%. I selected the quadratic SVM as the final choice of the classifier with $C = 0.01$.

I found $q = 0.85$ with $|S| = 15$ to be a good choice for the Relief score based feature selection. On the test set, the minimum misclassification rate of 3.352% was achieved by the polynomial SVM while the minimum false positive rate of 1.446% was achieved by the rbf SVM. Also, the misclassification rate with rbf SVM was 5.028% while the polynomial SVM had a false positive rate of 5.339%. I selected the polynomial SVM as the final choice of the classifier with $C = 0.01$.

I included the quadratic SVM with the Fisher score based feature selection and the polynomial SVM with the Relief score based feature selection as the final set of classifiers. I adopted a hybrid classification scheme where an image was classified as a lamp if both the classifiers predicted it to be a lamp otherwise it was classified as a non-lamp. This approach was successful in reducing the false positives and I obtained $\sim 95\%$ true positive rate and $\sim 3.5\%$ false positive rate with the hybrid approach. Fig. 3-19 shows the steps involved in creating the final hybrid classifier.

### 3.3.3 Summary

In this section, I discussed the development of a street lamp classification technique that can aid in automating inventorying of street lighting infrastructure. I posed the lamp identification as a machine learning problem on the video data collected by the car-top light scanning platform developed in this thesis. I first created a training set of lamp and non-lamp images by appropriate cropping of the video frames. I discussed the challenges in employing traditional image representation techniques that use edge and corner sets for the problem at hand and then developed a feature construction technique. Thereafter, I employed filtering techniques like Fisher score, Relief score and PCA for dimension reduction. I developed a two-step classification process, where
I first identified the image as a camera text or the “others” which was then classified as a lamp or a non-lamp. I presented the details of the feature subset selection and classifier optimization. The final hybrid classifier performed well on the test set with good accuracy and low false positive rate. Comparing the time stamps of the video frames that have street lamps with the GPS time stamps leads to direct geotagging of street lights.

As digital cameras are becoming cheap and pervasive, using images for infrastructure management has attracted significant attention. As described in this thesis, several important challenges related to object identification in an image need to be addressed. As a future work, I would like to explore a more sophisticated machine learning technique like deep learning that has the potential to automatically learn high level features from the image set and perform improved classification [81]. I would also like to make the data publicly available so that other researchers can explore novel image classification techniques for street lamp identification. In the next section, I will discuss estimation of another important parameter of the lighting infrastructure, height of the lamp post, which was aided by the results on lamp classification.
3.4 Lamp Post Height Estimation

Another important goal of this thesis was to estimate the height of the street lamps to aid in developing a proper inventory of the street lamps. Apart from the geographic location of the street lamps, the knowledge of the height of the lamp posts will help improve maintenance and replacement tasks. Fig. 3-20 is a visual representation of the imaging performed by the car-mounted camera system. Here we make an assumption that the car travelled parallel to the edge of the road, \textit{i.e.}, along the \( x \) axis with a speed \( U \). In this section, I present a perspective projection and optical flow based street lamp height estimation algorithm using the two-dimensional images collected by the car-mounted camera system.

![Figure 3-20: Geometry of the car-mounted imaging system.](image)

3.4.1 Perspective Projection

Perspective projection describes the mapping of a three-dimensional point in space to a two-dimensional coordinate on the image plane [66], [33]. As shown in Fig. 3-21, \( P(X, Y, Z) \) is the coordinate of the scene point in space and \( p(x, y, f) \) is the coordinate on the image plane and \( f \) is the camera’s focal length. Assuming camera to be an ideal pinhole and using properties of similar triangles, we have the following set of
Here the $Z$ axis is aligned with the optical axis and the $X - Y$ plane is considered parallel to the image plane.

![Perspective projection of a camera imaging system](image)

Figure 3-21: Perspective projection of a camera imaging system [33].

We go back to the geometry of the car-mounted imaging system described in Fig. 3-20 and define the following set of axes described in Fig. 3-22. $X - Y - Z$ is the frame of reference attached to the ground with the $Z$-axis pointing vertically up. The frame $X' - Y' - Z'$ is attached to the camera with the $Y'$ axis aligned with the optical axis and at an angle $\theta = 60^\circ$ with respect to $Y$ axis. The point $P$ is represented by the coordinated $(X', Y', Z')$ and its image $p$ by $(x, f, z)$. Note that the axis $X$ and $X'$ are coincident. Adapting (3.20) to the current system of axes, we get:

\[
\frac{x}{f} = \frac{X'}{Y'} \quad \Rightarrow \quad Y' = \frac{X' f}{x}
\]

and

\[
\frac{z}{f} = \frac{Z'}{Y'} \quad \Rightarrow \quad Z' = \frac{z Y'}{f}.
\]
The frame $X' - Y' - Z'$ can be obtained from the frame $X - Y - Z$ by rotating it about the $X$-axis counterclockwise by an angle $\theta = 60^\circ$. We now have the following coordinate transformation equations:

$$
\begin{bmatrix}
  Y \\
  Z
\end{bmatrix} =
\begin{bmatrix}
  \cos(\theta) & -\sin(\theta) \\
  \sin(\theta) & \cos(\theta)
\end{bmatrix}
\begin{bmatrix}
  Y' \\
  Z'
\end{bmatrix}.
$$

(3.22)

Since we are interested in the height or the $Z$ coordinate of the street lamp, we have the following equation for the $Z$ coordinate of the street lamp:

$$
Z = \sin(\theta)Y' + \cos(\theta)Z'
= \left(\sin(\theta) + \frac{z}{f}\cos(\theta)\right)Y'
= \left(\sin(\theta) + \frac{z}{f}\cos(\theta)\right)\frac{X'f}{x}.
$$

(3.23)

We have further assumed that the ground reference frame is attached to the roof of the car. To obtain the height of the lamp post with respect to the ground level we need to add the height of the car ($C_h$) to our estimate of $Z$, i.e.

$$
h_{\text{lamp}} = C_h + \left(\sin(\theta) + \frac{z}{f}\cos(\theta)\right)\frac{X'f}{x}.
$$

(3.24)
3.4.2 Optical Flow

Optical flow is defined as the apparent motion of the brightness pattern on an image as the camera moves with respect to an object or vice versa [66]. Referring to our imaging geometry of Fig. 3-22, let \( I(x, z, t) \) be the irradiance at a time \( t \) at the image point \((x, z)\). If \( u(x, z) \) and \( w(x, z) \) are the \( x \) and the \( z \) components of the optical flow vectors at that image point and if \( \partial t \) is a small time interval, we make the following assumption:

\[
I(x, z, t) = I(x + \partial t, z + \partial t, t + \partial t).
\] (3.25)

Using the Taylor series expansion of (3.25) and ignoring the higher order terms, we obtain the following standard optical flow equation:

\[
\frac{\partial I}{\partial x} \frac{dx}{dt} + \frac{\partial I}{\partial z} \frac{dz}{dt} + \frac{\partial I}{\partial t} = 0,
\]

\[
\Rightarrow I_x u(x, z) + I_z w(x, z) + I_t = 0,
\] (3.26)

where \( u = \frac{dx}{dt}, \ w = \frac{dz}{dt}, \ I_x = \frac{\partial I}{\partial x}, \ I_z = \frac{\partial I}{\partial z} \) and \( I_t = \frac{\partial I}{\partial t} \). For a digital image, (3.26) need to be solved at every pixel location \((i, j)\). In the discrete version, the spatial derivatives are replaced by a finite difference approximation and (3.26) can be written as

\[
I_x u_{i,j} + I_z w_{i,j} + I_t = 0,
\] (3.27)

where we use a finite difference approximation to compute \( I_x = \left( \frac{\partial I}{\partial x} \right)_{i,j}, \ I_z = \left( \frac{\partial I}{\partial y} \right)_{i,j} \) and \( I_t = \left( \frac{\partial I}{\partial t} \right)_{i,j} \).

Matlab employs two different methods to compute optical flow. The first method, Horn-Schunck, assumes the optical flow to be smooth and computes an estimate of the velocity field by minimizing the following equation:

\[
\sum_{i} \sum_{j} \left( I_x u_{i,j} + I_z w_{i,j} + I_t \right)^2 + \lambda \left( \left( \frac{\partial u}{\partial x} \right)^2 + \left( \frac{\partial u}{\partial z} \right)^2 + \left( \frac{\partial w}{\partial x} \right)^2 + \left( \frac{\partial w}{\partial z} \right)^2 \right),
\] (3.28)

where again a finite difference scheme is used to calculate the partial derivatives at the pixel location \((i, j)\). The second technique, Lucas-Kanade, divides the original
image into smaller sections and assumes a constant velocity in each section. It then
solves a weighted least squares problem in each section by minimizing the following
objective:

\[
\sum_{(i,j) \in \Omega} W^2 (I_x u + I_z w + I_t)^2,
\]

(3.29)

where \( W \) is a window function that emphasizes the constraint at the center of each
section, \( \Omega \) is the extent of the section and \((u, w)\) are the constant optical flow veloc-
ities obtained by minimizing (3.29). Fig. 3-23 shows an example image and the \( x \) and
\( y \) intensity gradient of the image. We observe that the magnitude of the intensity
gradient is primarily zero throughout the image and the pixels with higher magni-
tude are confined in a small region surrounding the lamps and other bright regions.
This observation prompts a modification of the Lucas-Kanade method where we first
identify the region of the image where a street lamp is located using the lamp clas-
sification technique and use that region to estimate the image velocity of the street
lamp.

Figure 3-23: (a) Example grayscale image frame, (b) Magnitude of \( x \)-direction in-
tensity gradient and (c) Magnitude of \( y \)-direction intensity gradient.
3.4.3 Algorithm for Height Estimation

The height of the lamp post as given by (3.24) is in terms of the unobserved coordinate \(X'\). In this section I will show how the concept of optical flow can be used to obtain an equation of the lamp height in terms of known quantities and observations. We can write the first part of (3.21) as

\[
X' = \frac{xy'}{f}.
\]  

(3.30)

I assume that as the car moves along \(X'\) axis, \(Y'\) or the distance of the lamp from the camera remains constant. Differentiating the left and the right side of (3.30) with respect to time gives

\[
U = \frac{u(x, z)Y'}{f} \Rightarrow Y' = \frac{uf}{u(x, z)},
\]  

(3.31)

where \(U\) is the speed of the car along the \(X\) axis. Substituting (3.31) in (3.23) we get

\[
h_{\text{lamp}} = C_h + \left( \sin(\theta) + \frac{z}{f} \cos(\theta) \right) \frac{uf}{u(x, z)},
\]  

(3.32)

where \(z\) is the image pixel corresponding to the street lamp and \(u(x, z)\) is the \(x\) component of the optical flow corresponding to the lamp pixel.

As discussed in Section 3.4.2, we find the optical flow velocity for lamp pixels by first identifying the street lamp using lamp classification and then use the Lucas-Kanade formulation by assuming constant \((u, w)\) over the region corresponding to the street lamp. Assuming \(W = 1\) in (3.29), we obtain the following estimate of \(u\):

\[
u = \frac{\sum (\sum_{(i,j) \in \Omega_l} I_{t_y}^2) (\sum_{(i,j) \in \Omega_l} I_{x,t} I_{t}) - (\sum_{(i,j) \in \Omega_l} I_{x,t} I_{y}) (\sum_{(i,j) \in \Omega_l} I_{y,t})}{\left(\sum_{(i,j) \in \Omega_l} I_{x}^2\right) \left(\sum_{(i,j) \in \Omega_l} I_{y}^2\right) - \left(\sum_{(i,j) \in \Omega_l} I_{x} I_{y}\right)^2},
\]  

(3.33)

where \(\Omega_l\) is the region of the image that is identified as a street lamp. The lamp identification algorithm further provided the coordinates of the centroid \((x_l, z_l)\) of the
bounding box corresponding to a lamp object. The z-coordinate of the centroid, $z_l$, was used in height estimate. The car speed, $U$, was measured by the OBD-II reader. We then have the following estimate of the height of the street lamp:

$$h_{\text{lamp}} = C_h + \left( \sin(\theta) + \frac{z_l}{f} \cos(\theta) \right) \frac{U f}{u}. \quad (3.34)$$

For the simulation, I used $C_h = 2 \text{ m}$, $\theta = 60^\circ$, $f = 2.8 \times 10^{-3} \text{ m}$ and the size of an image pixel to be $3.522 \times 10^{-4} \text{ m}$ (which was used to calculate $z_l$). From the simulation, I estimated the height of the Cobrahead street lamps at the intersection of Vassar street and Main street, Cambridge MA to be $\sim 12.9 \text{ m}$. Cobrahead street lamps are typically installed between $12-14 \text{ m}$.

### 3.4.4 Summary

I presented here an algorithm that can be used for estimating the height of street lamps. The method was developed by combining concepts on imaging geometry and optical flow. I used the results from lamp classification to improve estimates of optical flow. For the application of street light inventorying, we are interested in identifying tall, medium height and short lamps. Hence an error bar of few meters is acceptable and the algorithm presented here is a sufficient approach.

An accurate geospatial virtual database on urban infrastructure can significantly improve addressing maintenance and replacement needs. In this chapter, I presented methods that address analysis of the street lighting data collected by the mobile sensor platform. As a next step towards creating a virtual representation of urban street lighting, I would like to develop a three-dimensional light field model. It should be possible to combine the location and height of a lamp with the illumination measurements by the car-top light sensors array to create an estimate of the three-dimensional light field model created by a lamp. In the following section, I will introduce a framework that can combine the measurements and analysis presented before to form a three-dimensional light field model.
3.5 A Three-dimensional Light Field Model

Fig. 3-24 shows a model of the light field created by a lamp and $I_Z(X, Y)$ is the normal illuminance on the $X-Y$ plane at a certain height $Z$ from the surface. Note that here we are doing far-field modeling at distances significantly larger than the dimension of the source. We assume that $I_Z(X, Y)$ can be decomposed into the following functional form:

$$I_Z(X, Y) = I_0(Z)\Psi(X, Y, Z), \quad (3.35)$$

where $I_0(Z)$ models the variance of illumination intensity with height and $\Psi(X, Y, Z)$ models the illumination distribution on the $X-Y$ plane at a height $Z$. An inverse square law can be used to model $I_0(Z)$ as

$$I_0(Z) = \frac{I_0}{(z_1 - Z)^2}, \quad (3.36)$$

where $z_1$ is the height of the street lamp.

![Figure 3-24: A model of lamp light field.](image)

The cone of the illumination is determined by the angle $\alpha$ which governs how the contours of illumination vary with height. We model the spatial distribution of
illumination as a Gaussian distribution, i.e.,

\[ \Psi(X, Y, Z) = \frac{1}{2\pi \sqrt{|\Sigma_Z|}} \exp \left\{ -\frac{1}{2} (x - \mu)^T \Sigma_Z^{-1} (x - \mu) \right\}, \]  

(3.37)

where \( x = [X, Y]^T \), \( \mu \) is the \((X, Y)\) coordinate of the center of the lamp and \( \Sigma_Z \) is the covariance matrix governing illumination distribution. We can estimate \( \mu \) from the video and vehicle’s location data. The covariance matrix can be represented as

\[ \Sigma_Z = C_Z \begin{bmatrix} a^2 & \rho ab \\ \rho ab & b^2 \end{bmatrix} = C_Z \Sigma_0, \]  

(3.38)

where \( a \) and \( b \) are the standard deviations in \( X \) and \( Y \) direction respectively, \( \rho \) is the correlation coefficient and \( C_Z \) is a height dependent coefficient. The major and minor axes, \( l_1 \) and \( l_2 \), are obtained by finding the Eigenvalues of the covariance matrix and we assume that the major and minor axes scale linearly with height, i.e., \( l_1 = (z_l - Z) \gamma_1 \) and \( l_2 = (z_l - Z) \gamma_2 \) where \( \gamma_1 \) and \( \gamma_2 \) are constants. Knowing that \( C_Z = (z_l - Z) \) yields

\[ \Sigma_Z = (z_l - Z) \begin{bmatrix} a^2 & \rho ab \\ \rho ab & b^2 \end{bmatrix}. \]  

(3.39)

We now have the following model for illumination measurements:

\[ I_Z(x) = \frac{I_0}{2\pi \sqrt{|\Sigma_0|(z_l - Z)^3}} \exp \left\{ -\frac{1}{2(z_l - Z)} (x - \mu)^T \Sigma_0^{-1} (x - \mu) \right\}. \]  

(3.40)

We can obtain an estimate of \( x, \mu \) and \( Z \) from the video data, vehicle’s location and height. \( I_0 \) and \( \Sigma_0 \) can then be estimated from the light sensor array measurements using techniques like maximum likelihood estimators [37]. Once the parameters of the model are estimated, we can use the three-dimensional light field model to estimate illumination distribution at ground level which can then aid the decision makers to compare the current illumination levels with the standards.

92
3.6 Conclusion

In Chapter 3, I presented algorithms for analysis of the collected data that address management of street lighting infrastructure. There are several challenges in making inferences from noisy sensor measurements and false positives. I presented luminosity maps which is an overlay of average street illumination on a map and its use in both visualization and identification of maintenance needs. I developed an extended Kalman filter to improve vehicle location estimates in the presence of noisy and sparse GPS data. This has direct relevance to the use of vehicle based urban sensing as GPS suffers from outages and often has a sampling rate lower than desired. I presented how integrating car speed data obtained via off-the-shelf OBD sensor can address this issue and improve trajectory and travelled distance estimates.

I then addressed the problem of inventorying street lamps using video data. I developed an image classifier that can automatically identify street lamps in the video frames. Combing this with the GPS data can help geotag street lamps. I also developed an algorithm for estimating the height of the lamp posts which can further enhance the inventory database and help the decision makers in their maintenance operations. We envision a future where a regularly updated "Street View" of street lighting can help the decision makers to not only optimize maintenance and replacement needs but also identify under-lit and over-lit areas for improving service and saving energy. A three-dimensional model of light field generated by street lamps, like the one I presented in this thesis, can help drive the vision.

Chapter 2 and 3 studied the use of a mobile sensor system for addressing a practical problem in infrastructure management. The analysis of the data collected by real world deployment of the mobile sensor platform has opened up several fundamental questions that need to be addressed to understand the limits and applicability of mobile sensor systems. Specifically, in the context of field estimation and "hot spot" identification several fundamental questions arise. For example, where should we take samples given practical constraints on mobility, what would be the effect of uncertainty in the locations of mobile sensors and how should we model and estimate
a field from the observations taken by noisy mobile sensors. In the following two chapters, I study two problems related to mobile sensor based sampling and signal estimation: nonuniform sampling and errors in sample location.
Chapter 4

Arrangements and Deployment of Mobile Sensor Nodes

As discussed in Chapter 1 Section 1.2.1, a major application of distributed sensing is to reconstruct spatio-temporal physical fields, to predict how they will evolve and to determine the locations of their “hot spots” as shown in Fig. 4-1. A fundamental question in this regard is when and where the sensors should take samples to obtain a good estimate of the field. I call a geometric configuration of sampling locations (and times) a sensor arrangement and refer to this problem as a sensor arrangement problem. At a high level, a standard approach in the literature to address this problem is to assume some underlying model for the field and find a sensor arrangement that helps to best estimate unknown parameters of the model. Any approach needs to address three issues: sensor noise, modeling errors and sensor mobility. Sensor noise stems from errors in measurements whereas modeling errors corresponds to the difference between the actual field and the (unknown) model. Previous work has extensively addressed the first two issues by developing robust estimation methods to address sensor noise and modeling error. Examples of models studied so far include band-limited functions [130], shift-invariant functions [130], [13], models based on proper orthogonal decomposition [102], [111], Markov random fields and Gaussian processes [34], [76]. Methods for finding best sensor arrangements search for locations that minimize some error metric associated with the field estimation. Example error
metrics include mean squared error [39], weighted squared error [13], variance [42], entropy [92] and mutual information [51], [76].

Figure 4-1: Field estimation and “hot spot” identification.

The third issue in sampling concerns sensor mobility and has become very relevant with recent work in mobile sensor systems. I view sensor mobility as of two kinds, intentional and incidental [40]. In intentional mobility, a sensor is mounted on a mobile platform that moves to a specified location under its control (e.g. sensors attached to robots and UAVs). In incidental mobility, a sensor is mounted on a platform that moves on its own accord (e.g. sensors attached to cars and smartphones carried by humans). In this thesis, I address the sensor arrangement problem with a particular focus on incidentally mobile sensors. Unlike intentionally mobile sensors, specifying a particular sensor arrangement (e.g. uniform arrangement) for incidentally mobile sensors is of no use. There is a lack of control on the exact placement of incidentally mobile sensor nodes. Instead, I take the approach of finding a class of sensor arrangements that are compatible with the mobility of incidentally mobile sensors and lead to a stable reconstruction of the field.

As discussed earlier, a common approach to addressing the sensor arrangement problem is to assume a model for the underlying field and find a sensor arrangement that yields the best estimate of the field. Researchers from different domains including signal processing, computational mechanics, and statistics have investigated the sensor arrangement problem. In signal processing, one of the classic results in field reconstruction is the Shannon sampling theorem, which states that a band-limited field can be reconstructed using uniformly placed samples above the Nyquist rate, i.e. twice the highest frequency of the field [118], [130]. Furthermore uniform arrange-
ments of sensors not only enable faster reconstructions via fast Fourier transform (FFT), but are also robust to sensor noise in that they lead to the minimum mean-squared error of reconstruction [61]. In [39], the authors show that there exists a general class of non-uniform sensor arrangements that yield these exact same properties as uniform arrangements. Since Shannons original result, several researchers have addressed uniform and non-uniform sampling of band-limited and shift-invariant signal spaces. Non-uniform sampling does not necessarily require a higher number of samples but the reconstruction problem is difficult and more sensitive to noise [13], [50], [126]. References [13, 63, 62, 50, 126] consider δ-dense sampling schemes and their stability in the context of trigonometric polynomial fields and shift-invariant fields with applications in image processing and geo-physics. They obtain explicit bounds on δ for the case of trigonometric polynomials [63], [62]. However they consider only implicit bounds for shift-invariant functions [13].

In the computational mechanics domain, researchers have considered the optimal sensor placement problem for the reconstruction of convective-diffusive fields modeled using proper orthogonal decompositions [102], [14], [15]. In statistics, the sensor arrangement problem is the same as the optimal design of experiments [107] and researchers have used the Bayesian inference framework. In most cases, the emphasis has been on finding a particular optimal or near-optimal sensor arrangement for the problem at hand. With recent research in wireless sensor networks (WSNs), which has enabled pervasive sensing using mobile sensors, non-uniform sampling is not only practical but often the only option [55]. Most papers in this domain have addressed the sensor arrangement problem for intentionally mobile sensors. The exceptions are [42] and [40] where the authors propose a variety of error tolerant sensor arrangements for sampling trigonometric polynomial fields that guarantee that the mean-squared error of reconstruction is less than some error tolerance.

In this thesis, I studied non-uniform sensor arrangements for stable signal estimation. A part of this work has been published in the American Control Conference, 2012 [78]. I considered a field model that is a linear combination of known basis functions. In a typical forward parameter estimation problem, a linear estimator finds the
estimates of unknown coefficients of the basis functions that minimize the weighted least square error. In the presence of sensor noise and local variations, depending on the sampling locations the linear estimator may be ill-conditioned. I formulated the sensor arrangement problem as one of finding a class of sensor arrangements that lead to well-conditioned linear estimators and guarantee a stable reconstruction of the field. Motivated by the ideas in signal processing literature [13], [63] and [62], I considered a family of δ-dense sensor arrangements in which there is no square hole in the sampling domain bigger than 2δ. I derived sufficient conditions under which a signal can be reconstructed from δ-dense sampling. Furthermore, the results are valid for multidimensional signals and are thus applicable to a wide range of spatio-temporal field monitoring. I found an explicit bound on the condition number of the sampling operator in terms of δ, which improves when the field is known to have a sparse representation in the basis set. δ-dense sensor arrangements are geometrically intuitive and allow for easy encoding of simple sampling rules in incidentally as well as intentionally mobile sensors. I present simulation results on the condition number of direct linear estimators of the field and show that δ-dense sensor arrangements are not only flexible but also robust estimators with small condition numbers. I present discussions and examples on obtaining basis functions for physical fields. For a one-dimensional chemical diffusion field in a heterogeneous medium, I derived basis functions using proper orthogonal decomposition and present simulation results on the mean squared estimation error through δ-dense sampling in the presence of additive Gaussian noise.
4.1 Stable δ-Dense Sensor Arrangements for Sampling Physical fields

Let \( \{\phi_k(x)\}_{k=1}^{M} \) be an orthonormal set of basis functions defined on the \( D \)-dimensional domain \( Q = [0, 1]^D \), meaning

\[
\langle \phi_k(x), \phi_l(x) \rangle = \int_Q \phi_k(x)\phi_l(x) \, dx = \begin{cases} 1, & \text{if } k = l; \\ 0, & \text{if } k \neq l. \end{cases}
\] (4.1)

I assume the unknown scalar field to be in the span of \( \{\phi_k(x)\}_{k=1}^{M} \):

\[ f(x) = \sum_{k=1}^{M} a_k \phi_k(x). \] (4.2)

I denote \( \mathcal{R} \) as the set of real numbers and \( \mathcal{L}^2(Q) \) as the set of square integrable functions on \( Q \). The norm of a function is defined as \( ||f(x)|| = \sqrt{\langle f(x), f(x) \rangle} \). The vector inner product is defined as \( \langle x, y \rangle = x^T y = \sum_{i=1}^{D} x_i y_i \). The norm of a vector is defined as \( ||x|| = \sqrt{\langle x, x \rangle} \). Note that Gram-Schmidt orthogonalization can be used to create an orthonormal set of basis functions from any initial general set of basis functions [78].

Suppose samples \( \{y_i\}_{i=1}^{N} \) of the field were taken at \( N \) distinct sampling or sensor locations \( \{x_i\}_{i=1}^{N} \in Q \), as shown for a two-dimensional case in Fig. 4-2. I assume the measurement and modeling error to be additive independent, identically distributed Gaussian noise \( (\eta_i) \). I then have the following observation model: for each \( i = 1, 2, \ldots, N \),

\[ y_i = f(x_i) + \eta_i. \] (4.3)

I define \( X = \{x_i\}_{i=1}^{N} \) as a set of sampling points in time and/or space and refer to it as a sensor arrangement. Under the aforementioned setting the field estimation problem boils down to estimating the unknown coefficients \( a_k \)'s from the data set \( (x_i, y_i) \). In
matrix and vector notation, the observations can be described in the following form:
\[ y = \Phi a + \eta, \]  
where \( y = [y_1, y_2, \ldots, y_N]^T \), \( a = [a_1, a_2, \ldots, a_M]^T \), \( \eta = [\eta_1, \eta_2, \ldots, \eta_N]^T \) and
\[
\Phi = \begin{bmatrix}
\phi_1(x_1) & \cdots & \phi_M(x_1) \\
\phi_1(x_2) & \cdots & \phi_M(x_2) \\
\vdots & \ddots & \vdots \\
\phi_1(x_N) & \cdots & \phi_M(x_N)
\end{bmatrix}.
\]

![Figure 4-2: A δ-dense sensor arrangement in 2D with the sampling locations at \( x_i \). Each square partition \( S_i \) is of size \( \delta \times \delta \) and has one sampling point.](image)

The weighted least square (WLS) estimator of \( a \), which minimizes the weighted residual \( \sum_{i=1}^{N} w_i (y_i - f(x_i))^2 \), is:
\[
\hat{a} = (\Phi^T W \Phi)^{-1} \Phi^T W y = B^{-1} \Phi^T W y,
\]  
where \( B = \Phi^T W \Phi \) and \( w_i \)'s form the main diagonal of the weight matrix \( W \). Weights can be chosen in proportion to the separation between points, for example half the nearest neighbor distance of a sampling point [74]. Through the rest of the chapter I use half the nearest neighbor distance of \( x_i \) as a choice of \( w_i \) for our weight matrix.
Note that with \( w_i = 1 \) the regular least square (LS) estimator is obtained as:

\[
\hat{a} = (\Phi^T \Phi)^{-1} \Phi^T y = B_2^{-1} \Phi^T y,
\]

(4.6)

where \( B_2 = \Phi^T \Phi \). The field can now be reconstructed using:

\[
\hat{f}(x) = \sum_{k=1}^{M} \hat{a}_k \phi_k(x).
\]

(4.7)

### 4.1.1 Geometry of \( \delta \)-Dense Sensor Arrangements

In this section I define the geometry of \( \delta \)-dense sensor arrangements. As shown in Fig. 4-2, I consider partitioning of a closed and bounded two-dimensional domain \( Q \) into squares \( S_i \) of size \( \delta \times \delta \). This partitioning can be generalized to higher dimensions by considering partitions of the domain through hypercubes of size \( \delta^D \) in a \( D \) dimensional domain. The total number of square partitions \( N_s = \frac{1}{\delta^2} \). Every square partition \( S_i \) has one sampling location (Fig. 4-2). I call such a sensor arrangement \( X \) \( \delta \)-dense.

### 4.1.2 The Sensor Arrangement Problem

The estimators defined in (4.5) and (4.6) are functions of the sensor arrangement \( X \). As discussed before the measurements \( y_i \)'s can be noisy because of sensor noise and modeling error. An arrangement \( X \) is stable if the condition number of the matrix \( B \) (or \( B_2 \)), \( \kappa(B) \) (or \( \kappa(B_2) \)) is close to 1. This mitigates the effect of noise on the estimate of \( a \) and leads to higher numerical stability of the estimators defined in (4.5) and (4.6). Formally I define the sensor arrangement problem as finding a stable arrangement of \( X \). I also define parameter estimation through (4.5) or (4.6) to be a stable parameter estimation if \( \kappa(B) \) or \( \kappa(B_2) \) is close to 1.

Furthermore, \( X \) is a \( \delta \)-dense sensor arrangement which is a probabilistic sensor arrangement governed by the parameter \( \delta \). Essentially, I would like to find conditions on \( \delta \) such that every \( X \) generated by it leads to a stable parameter estimation.
4.1.3 Sufficient Conditions for Stable \( \delta \)-Dense Sensor Arrangements

In this section I derive sufficient conditions for \( \delta \)-dense sensor arrangements to lead to a stable sampling operator of functions represented by (4.2). For a \( \delta \)-dense sensor arrangement, I define the distance of every sampling location to its nearest neighbor as \( 2z_i \) and define the weight matrix \( W \) as:

\[
W = \begin{bmatrix}
  z_1 & 0 & \cdots & 0 \\
  0 & z_2 & \cdots & 0 \\
  \vdots & \vdots & \ddots & \vdots \\
  0 & 0 & \cdots & z_N 
\end{bmatrix}.
\] (4.8)

I define \( V_\Phi = \text{span}\{\phi_k(x)\}_{k=1}^M \) as the normed vector space generated by the orthonormal basis functions. Note that \( V_\Phi \) is a finite dimensional vector space. The gradient of a function is defined as \( \nabla f(x) = \left[ \frac{\partial f(x)}{\partial x_1}, \frac{\partial f(x)}{\partial x_2}, \ldots, \frac{\partial f(x)}{\partial x_D} \right]^T \). I define \( \mathcal{A} \) as the sampling operator similar to the one defined by Gröchenig [63], [62]. The measurements from a sensor arrangement \( X \) can be thought of as an operation \( \mathcal{A} \) on the function \( f(x) \). I can write

\[
\mathcal{A}(f(x)) = \mathcal{P} \left( \sum_{i=1}^N f(x_i) \chi_i(x) \right),
\] (4.9)

where \( \mathcal{P} \) is the orthogonal projection on \( V_\Phi \), i.e., \( \mathcal{P}(g(x)) = \sum_{k=1}^M (g(x), \phi_k(x)) \phi_k(x) \), \( x_i \)s represent the sampling locations and \( \chi_i(x) \)s are the characteristic functions with mutually disjoint support, i.e.

\[
\chi_i(x) = \begin{cases} 
1, & \text{if } x \in S_i; \\
0, & \text{o.w.}
\end{cases}
\]

and

\[
\int_Q \left( \sum_{i=1}^{N_S} \chi_i(x) \right) = 1.
\] (4.10)

The operation \( \mathcal{A} \) can be considered as sampling the signal at locations \( x_i \)s, making
a piecewise constant approximation of it and then projecting the piecewise constant approximation on $V_\Phi$.

The following theorems are the main result of this section:

**Theorem 4.1.** \( \forall f(x) \in V_\Phi \) and \( D \geq 2 \), let \( X \) be a \( \delta \)-dense sensor arrangement with 
\[ \delta < \frac{1}{\sqrt{D \sum_{k=1}^{M} c_k^2}} \]\text{ where } \( c_k = \max_{x \in Q} ||\nabla \phi_k(x)|| \), then \( X \) is a stable \( \delta \)-dense sensor arrangement.

For the special case of one-dimensional signals (Fig. 4-3) I have the following result:

**Theorem 4.2.** For \( D = 1 \), \( \forall f(x) \in V_\Phi \), let \( X \) be a \( \delta \)-dense sensor arrangement with 
\[ \delta < \frac{\pi}{\sqrt{2 \sum_{k=1}^{M} b_k^2}} \]\text{ where } \( b_k^2 = \int_{0}^{1} \left( \frac{d \phi_k(x)}{dx} \right)^2 dx \), then \( X \) is a stable \( \delta \)-dense sensor.

![Figure 4-3: A sensor arrangement in 1D with the sampling locations at \( x_i \) and partitions of size \( \delta \) denoted by \( S_i \).](image)

In order to prove the theorem, I make use of the following lemmas.

**Lemma 4.1.** Let \( A \) be a bounded operator on a Banach space \( B \). If there exists a \( \gamma \in [0, 1) \) such that \( \|f(x) - A(f(x))\| \leq \gamma\|f(x)\| \forall f(x) \in B \) then \( A \) is invertible and the inverse is bounded. I define such an operator as a stable sampling operator for \( B \) [62], [63].

**Lemma 4.2.** Cauchy-Schwarz inequality: \( \forall \alpha_k, \beta_k \in \mathbb{R} \):

\[
\left( \sum_{k=1}^{M} \alpha_k \beta_k \right)^2 \leq \sum_{k=1}^{M} \alpha_k^2 \sum_{k=1}^{M} \beta_k^2 \tag{4.11}
\]

**Lemma 4.3.** Wirtinger’s inequality for one-dimensional functions [62], [63]: If \( f(x), f'(x) = \frac{df(x)}{dx} \in L^2(a, b) \) and either \( f(a) = 0 \) or \( f(b) = 0 \) then

\[
\int_{a}^{b} |f(x)|^2 dx \leq \frac{4}{\pi^2} (b - a)^2 \int_{a}^{b} |f'(x)|^2 dx, \tag{4.12}
\]

103
where \( \mathcal{L}^2(a, b) \) is the space of square integrable functions defined on the closed interval 
\([a, b]\).

Proof of Theorem 4.1. For a sensor arrangement \( X \), I have:

\[
\|f(x) - Af(x)\|^2 = \left\| \sum_{i=1}^{N} f(x_i) \chi_i(x) \right\|^2 = \left\| \sum_{i=1}^{N} \left( f(x) - f(x_i) \right) \chi_i(x) \right\|^2 \\
\leq \left\| \sum_{i=1}^{N} \left( f(x) - f(x_i) \right) \chi_i(x) \right\|^2 \leq \sum_{i=1}^{N} \left\| \left( f(x) - f(x_i) \right) \chi_i(x) \right\|^2 \\
= \sum_{i=1}^{N} \int_{Q} \|f(x) - f(x_i)\|^2 \chi_i(x) dx. \tag{4.13}
\]

For a \( \delta \)-dense sensor arrangement, \( S_i \) is convex and using multivariate mean value theorem I can write

\[
f(x) - f(x_i) = (x - x_i)^T \nabla f(c), \tag{4.14}
\]

\( \forall x \in S_i \) and for some \( c \) on the line connecting \( x \) and \( x_i \). Since \( S_i \) is a convex domain, I further have

\[
\left| f(x) - f(x_i) \right| \leq \|x - x_i\| \|\nabla f(c)\|, \tag{4.15}
\]

where \( c \in S_i \). Taking the maximum of the right hand side gives

\[
\left| f(x) - f(x_i) \right| \leq \max_{x \in S_i} \|x - x_i\| \max_{x \in S_i} \|\nabla f(x)\|. \tag{4.16}
\]

Due to the geometry of the \( \delta \)-dense sensor arrangement the maximum distance between two points in a \( D \) dimensional hypercube of size \( \delta^D \) is \( \sqrt{D}\delta \). Also using Cauchy-Schwarz inequality I get:

\[
\|\nabla f(x)\| \leq \sqrt{\sum_{k=1}^{M} a_k^2} \sqrt{\sum_{k=1}^{M} \|\nabla \phi_k(x)\|^2}. \tag{4.17}
\]
Hence I can write, \( \forall \mathbf{x} \in S_i \)

\[
|f(\mathbf{x}) - f(\mathbf{x}_i)| \leq \sqrt{D} \delta \left( \sum_{k=1}^{M} a_k^2 \right) \left( \sum_{k=1}^{M} \max_{\mathbf{x} \in Q} ||\nabla \phi_k(\mathbf{x})||^2 \right)
\]

\[
\Rightarrow |f(\mathbf{x}) - f(\mathbf{x}_i)| \leq \sqrt{D} \delta \left( \sum_{k=1}^{M} a_k^2 \right) \left( \sum_{k=1}^{M} c_k^2 \right). \quad (4.18)
\]

Since \( f(\mathbf{x}) \in V_\phi \), \( ||f(\mathbf{x})||^2 = \sum_{k=1}^{M} a_k^2 \). Using (4.13), (4.18), \( N = N_S \), the fact that \( \chi_i(\mathbf{x}) \)'s have disjoint support and (4.10), I can write

\[
||f(\mathbf{x}) - Af(\mathbf{x})||^2 \leq \delta^2 D \left( \sum_{k=1}^{M} a_k^2 \right) \left( \sum_{k=1}^{M} c_k^2 \right) \int_{Q} \left( \sum_{i=1}^{N} \chi_i(\mathbf{x}) \right) d\mathbf{x}
\]

\[
= \delta^2 D \left( \sum_{k=1}^{M} a_k^2 \right) \left( \sum_{k=1}^{M} c_k^2 \right) = \delta^2 D \left( \sum_{k=1}^{M} a_k^2 \right) ||f(\mathbf{x})||^2. \quad (4.19)
\]

Now using Lemma 4.1, the sampling operator which is dependent on the sampling arrangement is stable for \( \delta < \frac{1}{\sqrt{D} \sum_{k=1}^{M} c_k^2} \). Hence \( \delta^* = \frac{1}{\sqrt{D} \sum_{k=1}^{M} c_k^2} \) is the critical value of \( \delta \) below which any \( \delta \)-dense sensor arrangement is stable. \( \square \)

**Proof of Theorem 4.2.** For a \( \delta \)-dense sensor arrangement \( X \) in 1D (Fig. 4-3), I have

\[
||f(\mathbf{x}) - Af(\mathbf{x})||^2 = \left| f(\mathbf{x}) - \mathcal{P} \left( \sum_{i=1}^{N} f(x_i) \chi_i(\mathbf{x}) \right) \right|^2 = \left| \mathcal{P} \left( \sum_{i=1}^{N} \left( f(x) - f(x_i) \right) \chi_i(\mathbf{x}) \right) \right|^2
\]

\[
\leq \left( \sum_{i=1}^{N} \left( f(x) - f(x_i) \right) \chi_i(\mathbf{x}) \right)^2 = \int_{0}^{1} \left( \sum_{i=1}^{N} \left( f(x) - f(x_i) \right) \chi_i(\mathbf{x}) \right)^2 dx = \sum_{i=1}^{N} \int_{z_i}^{z_{i+1}} (f(x) - f(x_i))^2 dx. \quad (4.20)
\]

Now I can write

\[
\int_{z_i}^{z_{i+1}} (f(x) - f(x_i))^2 dx = \int_{z_i}^{x_i} (f(x) - f(x_i))^2 dx + \int_{x_i}^{z_{i+1}} (f(x) - f(x_i))^2 dx. \quad (4.21)
\]
Using Lemma 4.3

\[ \int_{z_{i}}^{z_{i+1}} (f(x) - f(x_i))^2 dx + \int_{x_{i}}^{x_{i+1}} (f(x) - f(x_i))^2 dx \leq \frac{4\delta^2}{\pi^2} \int_{z_{i}}^{z_{i+1}} f'(x)^2 dx. \] (4.22)

Using (4.20) and (4.22) I have

\[ \|f(x) - Af(x)\|^2 \leq \frac{4\delta^2}{\pi^2} \int_{0}^{1} f'(x)^2 dx = \frac{4\delta^2}{\pi^2} \|f'(x)\|^2. \] (4.23)

Now, \( f'(x)^2 = \left( \sum_{k=1}^{M} a_k \phi_k'(x) \right)^2, b_k^2 = \int_{0}^{1} \phi_k'(x)^2 dx \) and using Lemma 4.2, I get

\[ \|f'(x)\|^2 = \int_{0}^{1} \left( \sum_{k=1}^{M} a_k \phi_k'(x) \right)^2 dx \leq \int_{0}^{1} \sum_{k=1}^{M} a_k^2 \sum_{k=1}^{M} \phi_k'(x)^2 dx \]
\[ = \sum_{k=1}^{M} a_k^2 \sum_{k=1}^{M} b_k^2. \] (4.24)

Using (4.23) and (4.24) I have

\[ \|f(x) - Af(x)\|^2 \leq \frac{4\delta^2}{\pi^2} \sum_{k=1}^{M} a_k^2 \sum_{k=1}^{M} b_k^2 = \frac{4\delta^2}{\pi^2} \left( \sum_{k=1}^{M} b_k^2 \right) \|f(x)\|. \] (4.25)

Now using Lemma 4.1, the sampling operator for 1D signals is stable for \( \delta < \pi/ \left( 2 \sqrt{\sum_{k=1}^{M} b_k^2} \right) \).

Hence \( \delta^* = \pi/ \left( 2 \sqrt{\sum_{k=1}^{M} b_k^2} \right) \) is the critical value of \( \delta \) below which any \( \delta \)-dense sensor arrangement is stable.

Note that the bounds in (4.19) and (4.25) present worst-case analyses. In many cases few dominating coefficient \( a_k \)s may be present and hence the bound will be much lower. For example, if the basis function with the maximum \( c_k \) (for 1D: \( b_k \)) has very low \( a_k \) or only a few dominating coefficients \( a_k \) are present in the case of sparse fields, the resulting bound will be lower. Nonetheless the theoretical bounds provide insights into the factors controlling reconstruction of signals through \( \delta \)-dense sampling and a good starting point to select \( \delta \).

Further the operator norm which is defined as \( \|A\|_{op} = \sup_{f(x) \in B} \|Af(x)\|/\|f(x)\| \)
has the following range:

\[ 1 - \delta \sqrt{D \sum_{k=1}^{M} c_k^2} \leq \|A\|_{op} \leq 1 + \delta \sqrt{D \sum_{k=1}^{M} c_k^2}, \]  
\[ (4.26) \]

or for the case of one-dimensional signals

\[ 1 - \frac{2\delta \sqrt{\sum_{k=1}^{M} b_k^2}}{\pi} \leq \|A\|_{op} \leq 1 + \frac{2\delta \sqrt{\sum_{k=1}^{M} b_k^2}}{\pi}. \]  
\[ (4.27) \]

Let \( \kappa(A) \) denote the upper bound on the condition number of the operator \( A \). The following relationship holds for \( D \geq 2 \) and using (4.26):

\[ \kappa(A) \leq \frac{1 + \delta \sqrt{D \sum_{k=1}^{M} c_k^2}}{1 - \delta \sqrt{D \sum_{k=1}^{M} c_k^2}} \]  
\[ (4.28) \]

Since I am considering the sensing model as described in (4.2) and a finite dimensional vector space of functions \( V_\Phi \), estimating the unknown coefficient vector is sufficient for reconstructing the scalar field. I claim that the conditions derived above are sufficient conditions leading to stable sampling and parameter estimation in the presence of noise. Obtaining parameters involve inverting the sampling operator that can be achieved either directly [42] or an iterative approach [63]. Here I consider a direct approach using the weighted least square estimator (4.5) and least square estimator (4.6). The condition numbers \( \kappa(B) \) and \( \kappa(B_2) \) of the matrices \( B \) and \( B_2 \) respectively affect the stability of the estimators. High condition numbers amplify the effect of the noise on parameter estimation.

For a sensor arrangement requiring sampling at specific locations mobile sensing platforms would need accurate and sophisticated position controllers making it expensive and energy inefficient. On the other hand \( \delta \)-dense arrangements open up the possibility of using simpler, energy efficient position controllers for mobile sampling. The only constraint that the \( \delta \)-dense arrangements impose is that every square partition of size \( \delta \) has one sample in it. For instance, in case of sampling air quality
using sensors mounted on taxicabs, δ-dense sensor arrangement simply means that one needs to take a sample once at least in a span of δ distance. Further, while dropping sensors from UAVs, a δ-dense arrangement needs to be ensured.

4.1.4 Numerical Simulations Showing the Stability of Linear Estimators in δ-Dense Sensor Arrangements

I now present the stability (condition number) of the linear estimator discussed in Section 4.1. I consider two classes of basis functions in 2D, polynomial and cosine basis functions. Consider the following set of basis functions:

\[
\psi_k(x_1, x_2) = x_1^p x_2^q, \quad (4.29)
\]

where \( p = q = \{0, 1, 2, 3\} \), leading to 16 basis functions in the set. I use the aforementioned set to create a set of 16 orthonormal basis functions \( \{\phi_k(x_1, x_2)\}_{k=0}^{15} \) in two dimensions using Gram-Schmidt orthogonalization [78]. I follow the same procedure with cosine basis functions:

\[
\psi_k(x_1, x_2) = \cos(p x_1) \cos(q x_2), \quad (4.30)
\]

to create another set of orthonormal basis functions.

For a given δ I simulate 1000 randomly generated δ-dense arrangements (Section 4.1.1) and calculate the mean and the standard deviation of \( \kappa(B) \) and \( \kappa(B_2) \). Fig. 4-4 shows the variation of the mean condition numbers with δ for the two sets of basis functions. The error bars indicate the corresponding \( \frac{1}{2} \) standard deviations. For polynomial basis functions \( \delta^* = 0.0036 \) and for cosine basis functions \( \delta^* = 0.0014 \). The theoretical bounds are very conservative and numerically I find that \( \delta \leq 0.025 \) gives a mean condition number \( \leq 1.4 \) with standard deviation \( \leq 0.15 \) for both polynomial and cosine basis functions. Furthermore, \( \kappa(B_2) \leq \kappa(B) \) implying that in the context of δ-dense arrangements generated using my method, the least square estimator is more robust to additive noise as compared to the weighted least square estimator. Though
δ-dense arrangements are probabilistic in nature, the mean condition numbers are small with acceptable variance for low values of δ, providing a suitable framework for sampling parametric signals.

Figure 4-4: Variation of the mean of condition number of $B$ and $B_2$ with δ for 2D a) orthonormal polynomial basis functions; b) orthonormal cosine basis functions. For δ ≤ 0.025 I obtain small condition numbers implying higher numerical stability and robustness to noise. The error bars indicate $\frac{1}{2}$ standard deviations.

4.2 Obtaining Basis Functions for Physical Fields

Several choices of basis functions such as trigonometric polynomials, band limited functions, shift-invariant functions have been explored in the literature. As discussed in [102, 111, 14, 15], proper orthogonal decomposition (POD) has emerged as a powerful methodology to obtain reduced order models of complex phenomena and have shown success in the domain of fluid modeling, nonlinear dynamics and computational science. In scenarios where the signal is generated by a physical phenomenon governed by a set of differential equations, POD is a promising technique for appropriate selection of basis functions. Furthermore, these basis functions need not be bandlimited in the Fourier sense and hence open up new challenges in stable sampling and in the reconstruction of fields with generalized basis functions. For the sake of completeness I present a mechanism to derive PODs.

Suppose $f(\mathbf{x}, \mu)$ is a parametric scalar field that is obtained as a result of the solution of a parametric governing differential equation such as chemical diffusion.
The parameter \( \mu \) has physical relevance. For example, it could be the position of release of pollutants affecting the diffusion field. As described in [31], I can write the M-term approximation as:

\[
f(x, \mu) \approx \sum_{k=1}^{M} a_k(\mu)\phi_k(x). \tag{4.31}
\]

The main question is how to optimally choose \( \phi_k(x) \). Although PODs are used for continuous fields, here I present their discrete version on a fine resolution grid allowing the use of matrix operations. The function can now be represented as a matrix \( A_{P \times Q} \) where each column \( \mathbf{a}_q \) represents \( f(x, \mu) \) on a fine resolution grid with \( P \) points for a particular choice of \( \mu \). I can now write the singular value decomposition of \( A \) as

\[
A_{P \times Q} = U_{P \times P} \Sigma_{P \times Q} V_{Q \times Q}^T = U W^T = \sum_{k=1}^{P} u_k w_k^T, \tag{4.32}
\]

where \( U \) comprises of the left singular vectors of \( A \), \( V \) comprises of the right singular vectors of \( A \), \( \Sigma \) is the matrix of singular values \( (\sigma_1 \geq \sigma_2 \ldots \geq \sigma_Q \geq 0) \), \( W = V \Sigma^T \), \( u_k \)'s form the columns of \( U \) and \( w_k \)'s the columns of \( W \). Eqn. (4.32) is the discrete version of (4.31). POD basis functions should contain as much information or energy as possible. Mathematically, obtaining a POD basis function involves solving the following optimization problem:

\[
\hat{x} = \arg \max_{\mathbf{x}} \sum_{q=1}^{Q} (\langle \mathbf{a}_q, \mathbf{x} \rangle)^2 \quad \text{s.t.} \quad \|\mathbf{x}\| = 1. \tag{4.33}
\]

Using Lagrange multiplier \( \lambda \), the solution to the above constrained optimization problem is given by

\[
AA^*\mathbf{x} = \lambda \mathbf{x}. \tag{4.34}
\]

From the singular value decomposition of \( A \), I get that \( u_1 \) is the solution to the above problem with \( \lambda = \sigma_1^2 \) and the resulting value of the objective function in (4.33) is \( \sigma_1^2 \).
Continuing with the above procedure, I find that \( u_i \), for \( i = 1 \ldots Q \) solves

\[
\mathbf{x} = \arg \max \sum_{q=1}^{Q} (a_q, \mathbf{x})^2 \quad \text{s.t.} \quad \|\mathbf{x}\| = 1 \quad \text{and} \quad \langle \mathbf{x}_j, \mathbf{x} \rangle = 0 \quad \text{for} \quad j = 1 \ldots i - 1.
\]

Since the singular values are arranged in decreasing order, I can choose the first \( M \) PODs \( (u_k) \) which are the first \( M \) most significant modes as a choice for \( \phi_k(\mathbf{x}) \)s on the discrete grid. From the perspective of a real world deployment of mobile sensors, a prior simulation with a large number of different parameters would help in choosing the most significant basis functions.

### 4.2.1 Example Field Modeling through POD and Field Estimation through \( \delta \)-Dense Sensor Arrangement

I now consider an example of constructing basis functions through POD. Consider the following one-dimensional model of unsteady chemical diffusion in a heterogeneous medium with a source term [72]:

\[
\frac{\partial C(x_1, t)}{\partial t} = \frac{\partial}{\partial x_1} \left( D(x_1) \frac{\partial C(x_1, t)}{\partial x_1} \right) + S(\mu),
\]

where \( C(x_1, t) \) is the chemical concentration field, \( D(x_1) \) is the diffusion coefficient and \( S(\mu) \) is the source term.

I consider the field at a particular time instant \( T_0 \), when it changes very slowly in time and is close to the steady state. Such steady chemical fields can be used to model the steady release of gas or oil in a quiescent environment, like oil leak from underwater pipes in a lake. I further assume that the source term is a narrow Gaussian impulse of known amplitude and variance but of unknown mean position \( \mu \in [0, 1] \), which acts for all times. The particular choice of the source term models the release of chemicals at an unknown location in the domain.

I develop a finite difference scheme to numerically solve the non-linear partial differential equation (4.36). For \( x_1 \in [0, 1] \) and \( t \in [0, T_0] \), I use the index \( i \) for the
spatial coordinate \( x \), and the index \( j \) for time \( t \). I discretize the space-time grid using \( \Delta x = 1/(P - 1) \), \( i = 1 \ldots P \) and \( \Delta t = T_0/(N - 1) \), \( j = 1 \ldots N \). I use an implicit scheme for time differencing and a central difference for the spatial coordinate. I assume Dirichlet boundary conditions, i.e., \( C_{1,j} = \beta_0 \) and \( C_{P,j} = \beta_1 \) for \( j = 1 \ldots N \) and the initial condition as \( C_{i,1} = 0 \) for \( i = 2 \ldots P - 1 \). Equation (4.36) can be written as

\[
\frac{\partial C(x_1, t)}{\partial t} = \frac{\partial D(x_1)}{\partial x_1} \frac{\partial C(x_1, t)}{\partial x_1} + D(x_1) \frac{\partial^2 C(x_1, t)}{\partial x_1^2} + S(\mu). \tag{4.37}
\]

I then have the following finite difference scheme of second order:

\[
\frac{C_{i,j+1} - C_{i,j}}{\Delta t} = \frac{D_{i+1} - D_{i-1}}{2\Delta x_1} C_{i+1,j+1} - C_{i-1,j+1} + D_i \left( \frac{C_{i-1,j+1} - 2C_{i,j+1} + C_{i+1,j+1}}{\Delta x_1^2} \right) + S_i
\]

\[
\Rightarrow C_{i,j+1} - C_{i,j} = \alpha \left( \frac{D_{i+1} - D_{i-1}}{4} + D_i \right) C_{i-1,j+1} - 2\alpha D_i C_{i,j+1}
\]

\[
+ \alpha \left( \frac{D_{i+1} - D_{i-1}}{4} + D_i \right) C_{i+1,j+1} + \Delta t S_i, \tag{4.38}
\]

where \( \alpha = \frac{\Delta t}{\Delta x_1^2} \). Rearranging (4.38) gives

\[
A c_{j+1} = b_j, \tag{4.39}
\]

where \( c_{j+1} = [C_1,j+1, C_2,j+1, \ldots, C_{P,j+1}] \), \( b_j = [\beta_0, \Delta t S_2 + C_2,j, \ldots, \Delta t S_P + C_{P-1,j}, \beta_1] \) and

\[
A_{i,1} = A_{P,P} = 1, \quad A_{i-1,i} = \alpha \left( \frac{D_{i+1} - D_{i-1}}{4} - D_i \right),
\]

\[
A_{i,i} = 1 + 2\alpha D_i, \quad A_{i+1,i} = -\alpha \left( \frac{D_{i+1} - D_{i-1}}{4} + D_i \right) \quad \text{for} \quad i = 2 \ldots P - 1
\]

and for all other entries \( A_{i,j} = 0 \). \tag{4.40}

For the simulations, I choose \( \Delta x_1 = 1/1000 \) and \( \Delta t = 1 \) s. I solve (4.39) iteratively and record the solution at \( N = 5001 \) (\( T_0 = 5000 \) s) as the steady state chemical concentration. I choose \( S(\mu) = \exp(-200(x_1 - \mu)^2) \) and \( D(x_1) = 0.001 + \)
Figure 4-5: (a) Variation of diffusivity, source term and chemical concentration profile in the domain; (b) First 5 POD basis functions and the corresponding singular values (\[|\sigma_i/10|\]) generated for the model in (4.36); (c) Decay of singular values with the POD basis function number.

\[0.01 \exp(-100(x_1 - 0.3)^2) + 0.005 \exp(-20(x_1 - 0.75)^2).\]

I carry out simulation with different values of \(\mu\) between 0 and 1 in increments of 0.01 and use the resulting fields to generate the PODs. Fig. 4-5a shows the variation of \(D(x_1)\), an example source term \(S(\mu)\) and the resulting concentration field \(C(x_1, T_0)\) for \(\mu = 0.28\). The particular profile of \(D(x_1)\) was chosen to model a domain where certain regions have higher diffusion coefficients leading to faster diffusion of chemical species. I assemble the solutions in a matrix \(A\) and perform the analysis in (4.32) after centering each column to find the PODs. I then chose the constant 1 and the first 10 PODs as the set of 11 basis functions to represent \(C(x_1, T_0)\). For the sake of illustration, I show the first 5 PODs with the corresponding singular values in Fig. 4-5b. Fig. 4-5c shows that the singular values decay rapidly highlighting the suitability of choosing few modes.
for representing solutions to the parametrized partial differential equation in (4.36).

Figure 4-6: (a) Variation of the mean of condition number of $B$ and $B_2$ with $\delta$ for the PODs ($\alpha = 0.01$); (b) MSE in reconstructing the test function through $\delta$-dense sampling with different magnitudes of additive noise ($\alpha$). I report error in direct reconstruction through weighted least squares (4.5) in blue and least squares (4.6) in red. The error bars indicate $1/2$ standard deviations.

Figure 4-7: Comparison of the MSE with uniform sampling and $\delta$-dense sampling. Note that the weighted least squares (WLS) and least squares (LS) estimator are identical for uniform sampling. $N = 1/\delta$. The error bars indicate $1/2$ standard deviations.

I evaluate the performance of $\delta$-dense sensor arrangements in reconstructing $C(x_1, T_0)$ for a particular value of $\mu$ different from the ones used for learning the POD basis functions. Gaussian noise, $\eta \sim \mathcal{N}(0; \alpha \max |C(x_1, T_0)|)$ was added to the measurement vector for each choice of $\alpha$ from the set $\{0.01, 0.03, 0.05\}$ with $\max |C(x_1, T_0)| = 14.7323$ and $\|C(x_1, T_0)\| = 10.6058$. Using the method described in Section 4.1.3 I find $\delta^* = 0.0205$ for the set of POD basis functions. I perform simulations with
different values of $\delta$ and for each $\delta$ I generate 1000 arrangements to compute the mean and the standard deviation of the condition numbers and the mean squared estimation errors (Fig 4-6). Similar to Fig. 4-4, Fig 4-6 shows that for low $\delta$ both the mean condition numbers are small with reasonable variances. Furthermore, the mean squared error (MSE) in the field estimation is $\leq 6.5\%$ even at high levels of noise ($\alpha = 0.05$) for $\delta \leq 0.05$. As shown in Fig. 4-7, the MSE with $\delta$-dense compares well with uniform sampling which highlights its suitability for mobile spatial sampling.

![Figure 4-8: Comparison of the MSE with three different choices of basis function sets. POD refers to the POD basis function set, Trig refers to the trigonometric basis function set and Poly refers to the polynomial basis function set. For all the simulations $\alpha = 0.01$ and the least squares method (4.6) was used.](image)

In order to demonstrate the efficacy of the PODs, I further compare the above result with the mean squared estimation error when the field is modeled in a different basis set. I choose $\psi_0(x_1) = 1$, $\psi_k(x_1) = \sin(2\pi k x_1)$ for $k = 1, \ldots, 5$, and $\psi_k(x_1) = \cos(2\pi(k - 5)x_1)$ for $k = 6, \ldots, 10$. I create a set of orthonormal basis functions from $\psi_k(x_1)$s and call it the set of trigonometric basis functions. Similarly, I create a set of orthonormal polynomial basis functions using $\psi_k(x_1) = x_1^k$ for $k = 0, \ldots, 9$. Fig. 4-8 shows that the POD basis set outperforms the other two in terms of the mean squared estimation error.

In the above example, I assumed the location of the Gaussian source to be unknown while the shape of the Gaussian distribution and the diffusion coefficient are assumed to be known. I created the POD basis set by choosing different values of
Figure 4-9: Comparison of the MSE with the three basis function sets when there is an approximate knowledge of model parameters. (a) $D'(x_1) = D(x_1) + 0.01 \sin(5x_1)$; (b) $D'(x_1) = D(x_1) + 0.01 \cos(10x_1)$; (c) $S'(\mu) = \exp(-500(x_1 - \mu)^2)$; (d) $S(\mu) = \exp(-20(x_1 - \mu)^2)$.

$\mu$ and generating solutions to (4.36), hoping that the solutions will capture enough information about the solution space and the basis set be effective in representing future test signals. In many scenarios there is limited information about the system, for example, the diffusion coefficient $D(x_1)$ may not be completely known. One way to address this is to sample $D(x_1)$ from an assumed distribution and extend the framework discussed before, though this may increase the overall computational complexity. On the other hand, one can use the $D(x_1)$ to the best of our knowledge and use the framework presented before, hoping that the PODs generated will still have a good approximation capability. For example, if the actual diffusion coefficient or the shape of the Gaussian source is different compared to the ones used for generating the PODs, the basis set may not be optimal and I would like to assess the applicability of the PODs generated by the assumed model in representing the field. Fig. 4-9 shows
the comparison of the three basis sets in estimating a test chemical concentration field that was generated with a $D(x_1)$ or a $S(\mu)$ different from the ones used for generating the PODs. The POD basis set gives a lower MSE when compared to the other two basis sets even in the presence of model uncertainty which highlights the utility of using precomputed simulation results on an approximate model to generate a basis set for the test field.

4.3 Conclusion

In this chapter, I addressed the sensor arrangement problem: when and where sensors should take samples to obtain a stable reconstruction of a physical field. I modeled the field as a linear combination of known basis functions and used linear estimators for the reconstruction of the field. I considered a family of $\delta$-dense sensor arrangements and obtained sufficient conditions ($\delta < \delta^*$) for $\delta$-dense arrangements to lead to a stable estimator. My results are valid for multidimensional signals making the $\delta$-dense sampling framework suitable for a broad class of signals. The bound on $\delta$ improves when the field is sparse. Using numerical simulations, I showed that $\delta$-dense arrangements yield stable reconstructions of the field in the presence of sensor noise and modeling error. I also highlighted that $\delta$-dense arrangements yield sufficient flexibility in terms of sampling using incidentally mobile sensors. I presented a proper orthogonal decomposition based framework for obtaining basis functions to model fields that are solutions to a parametric differential equation and compared its performance with naive basis sets.

For future work, I would like to explore the field sparsity further and improve bounds on $\delta$. In addition, I would like to consider explicitly time varying dynamic fields and explore classes of stable adaptive sensor arrangements. In the next chapter, I address the effect of sample location errors on signal estimation which has direct relevance to mobile spatial sampling as it is difficult and often impossible to obtain the exact location of a mobile agent.
Chapter 5

Signal Estimation from Samples with Location Errors

Researchers have studied and developed various localization schemes for mobile platforms. GPS, which can be used to obtain position information of mobile sensing platforms in outdoor environments like taxicabs, has an accuracy of a few meters [19], [145]. Commercially available differential GPS can further achieve sub-meter accuracy [99]. Localization of autonomous robots has been studied extensively in the literature [52, 53, 64, 84, 127]. Though significant advances have been made in the localization of autonomous robots, location errors exist and their effect on the reconstruction of a spatio-temporal field monitored by a system of mobile agents has not received much attention. As the interest in real-world deployment of mobile WSNs is increasing, the aforementioned problem is expected to be of utmost practical interest. Tolerating higher sample location errors would in turn allow the use of simpler and less expensive localization/positioning systems, increasing the feasibility of real-world deployment of mobile sensor nodes.

Location errors in sampling have been studied in the signal processing community as random time jitter in analog-to-digital converters (ADC) [22, 90, 103, 137, 136]. Typically the power consumption of an ADC goes up with the speed and the accuracy of the clock signals [137]. Hence tolerating higher time jitter through post processing presents an opportunity for lowering the power of acquisition subsystems.
In [22, 90, 103, 95] the authors characterize error due to time jitter and study the estimation error of bandlimited signals in the presence of jitter. In [22] the author develops a linear interpolator for processing bandlimited signals with correlated and uncorrelated time jitter. In [137] the authors implement a nonlinear maximum likelihood estimator through an expectation-maximization (EM) algorithm to mitigate the effect of sample location error in the reconstruction of bandlimited signals. Likewise, Bayesian approaches through Gibbs sampling and slice sampling have been explored to non-linearly post-process the jitter affected samples, showing success in reducing the effect of time jitter [136]. Nonlinear estimators like the polynomial least squares estimator have also been explored in the literature [135]. Though nonlinear estimators exhibit improved performance over linear estimators, in general they suffer from high computational cost and limit the possibility of developing real-time embedded applications.

In this thesis, I have developed a computationally efficient estimator to mitigate the effect of sample location errors in reconstructing signals that lie in a finite-dimensional subspace. This work has been published in the IEEE Transactions on Signal Processing [77]. The parameter estimation problem is formulated as obtaining a maximum likelihood estimate given the observations and an observation model. Using the linear term from a Taylor series expansion, I derive an approximation to the nonlinear observation model. I then adopt an iterative approach to develop a computationally efficient estimator, the iterative linear estimator (ILE), which captures the first order effect of sample location errors on signal estimation. I highlight practical issues in computing the numerical approximation of the complete likelihood function and show the suitability of Monte Carlo integration over Gauss-Hermite quadrature for our problem. Using Monte Carlo integration I derive a numerical scheme for finding the Cramér-Rao lower bound (CRB) and an EM algorithm to implement the maximum likelihood estimation.

I present simulations to show the efficacy of the proposed algorithm in improving the estimation error at low computational cost. When compared with a baseline linear estimator, ILE gives a lower estimation error and the improvement increases with the
variance of the sample location error but decreases with the number of sampling points. I present statistical properties of our estimator, such as contribution of bias versus variance and the mean computation time. The mean computational time with ILE for the simulated one-dimensional signals is of the order of $10^{-2}$ s which increases to 1 s for the simulated two-dimensional signals. For a one-dimensional signal, I show that for low additive noise and a reasonable level of oversampling ILE outperforms EM (in terms of estimation error) while at higher noise and higher number of samples EM algorithm gives a lower estimation error. I also show that the performance of EM worsens with decreasing additive noise due to high numerical errors associated with computing integrals that involve narrow Gaussian distributions, while ILE does not suffer from such problems. Furthermore, implementation of ILE is around five orders of magnitude faster when compared to EM, making it an appealing candidate for real-time embedded applications.

My approach is related to the Taylor-series based vehicle position estimation [54], where the known functional relationships between the unknown vehicle position and the known observation stations are linearized to obtain a linear system of equations which is subsequently solved iteratively until convergence. The key difference between my work and [54] is that in our case the Taylor-series approximation does not lead to a linear system of equations as both the function and the location errors are unknown.

### 5.1 Notation

Let $\{\phi_k(x)\}_{k=1}^{M}$ be an orthonormal set of continuous and differentiable functions defined on the $D$-dimensional domain $Q \triangleq [-1, 1]^D$, meaning

$$
\langle \phi_k(x), \phi_l(x) \rangle = \int_{Q} \phi_k(x)\phi_l(x) \, dx = \begin{cases} 1, & \text{if } k = l; \\ 0, & \text{if } k \neq l. \end{cases}
$$

(5.1)
I consider signals in the span of \( \{ \phi_k(x) \}_{k=1}^{M} \):

\[
f(x) = \sum_{k=1}^{M} a_k \phi_k(x). \tag{5.2}
\]

I denote the \( L_2 \) norm of the signals as \( \| f(x) \| = \sqrt{\langle f(x), f(x) \rangle} \). For a vector in \( D \) dimensional space I use the regular \( \ell_2 \) norm as \( \| x \| = \sqrt{\sum_{j=1}^{D} x_j^2} \). The determinant of a matrix \( M \) is denoted as \( |M| \).

The signal estimation problem can be defined as obtaining an estimate of \( \{ a_k \}_{k=1}^{M} \) by appropriate sampling of the signal and employing an estimation algorithm. Note that here I do not restrict our analysis to bandlimited functions, making my framework suitable for a wide range of spatio-temporal signals frequently encountered in the real-world application of pervasive sensing. I assume uniform arrangements of the nominal/expected sampling locations with equal spacing between the samples in every dimension. I assume the location errors to be independent, identically distributed (i.i.d.) Gaussian noise and the measurement/modeling error to be additive i.i.d. Gaussian noise. I then have the following observation model: for each \( i = 1, 2, \ldots, N \),

\[
y_i = f(x_i + z_i) + \eta_i, \tag{5.3}
\]

where \( x_i \) is the nominal sensor location, \( z_i \sim \mathcal{N}(0, \sigma_z^2 I) \) is the location error (with per-component standard deviation \( \sigma_z \)), \( \eta_i \sim \mathcal{N}(0, \sigma^2) \) is the additive noise (with standard deviation \( \sigma \)) and \( y_i \) is the observation. The noise matrix \( \{ z_i^T \}_{i=1}^{N} \) and the noise vector \( \{ \eta_i \}_{i=1}^{N} \) are independent. In theory, the Gaussian distribution assumption of location error may lead the sampling points to be outside \( Q \). In simulations, I ensure the points are in \( Q \) by considering location errors to be a fraction of the spacing between the nominal sampling points. The assumption of Gaussian distribution enables an analytical treatment of the problem and subsequent derivation of the estimator.

In matrix and vector notation the observations can be described in the following semilinear form:

\[
y = \Phi(X, Z)a + \eta, \tag{5.4}
\]

122
where \( \mathbf{y} = [y_1, y_2, \ldots, y_N]^T \), \( \mathbf{a} = [a_1, a_2, \ldots, a_M]^T \), \( \pmb{\eta} = [\eta_1, \eta_2, \ldots, \eta_N]^T \), \( X = \{x_i | i = 1 \ldots N\} \),

\[
Z = \{z_i | i = 1 \ldots N\} \text{ and } \Phi(X, Z) =
\begin{bmatrix}
\phi_1(x_1 + z_1) & \cdots & \phi_M(x_1 + z_1) \\
\phi_1(x_2 + z_2) & \cdots & \phi_M(x_2 + z_2) \\
\vdots & \ddots & \vdots \\
\phi_1(x_N + z_N) & \cdots & \phi_M(x_N + z_N)
\end{bmatrix}
\]

For convenience I denote the \( j^{th} \) row of \( \Phi(X, Z) \) as \( \phi_j(z_j)^T = [\phi_1(x_j + z_j), \ldots, \phi_M(x_j + z_j)] \). From the observation model one can clearly see that the effect of \( z_i \) on \( y_i \) is non-linear and non-trivial to characterize. I denote the probability density function (pdf) of \( \mathbf{x} \) by \( p(\mathbf{x}) \), the pdf of \( \mathbf{x} \) parametrized by a nonrandom vector \( \mathbf{a} \) by \( p(\mathbf{x}; \mathbf{a}) \), and the pdf of \( \mathbf{x} \) conditioned on a random variable \( \mathbf{y} \) by \( p(\mathbf{x}|\mathbf{y}) \). I define expectations similarly. I denote the multivariate normal distribution of a \( D \) dimensional vector \( \mathbf{x} \) by

\[
\mathcal{N}(\mathbf{x}; \mu, \Sigma) = |(2\pi)^D \Sigma|^{-1/2} \exp\left\{-\frac{1}{2} (\mathbf{x} - \mu)^T \Sigma^{-1} (\mathbf{x} - \mu) \right\}, \quad (5.5)
\]

where \( \mu \) is the mean vector and \( \Sigma \) is the covariance matrix.

### 5.2 Parameter Estimation

In parameter estimation, we seek to find an estimate \( \hat{\mathbf{a}} \) of \( \mathbf{a} \) which minimizes a certain cost function. Typically the cost function is chosen to be the MSE:

\[
C(\hat{\mathbf{a}}(\mathbf{y}); \mathbf{a}) = E_{\mathbf{y}}[||\hat{\mathbf{a}} - \mathbf{a}||^2]. \quad (5.6)
\]

Using the estimate of the parameters we reconstruct the signal using

\[
\hat{f}(\mathbf{x}) = \sum_{k=1}^{M} \hat{a}_k \phi_k(\mathbf{x}). \quad (5.7)
\]

As discussed in [138], it is impossible to compute the minimum MSE (MMSE) estimator without prior knowledge about \( \mathbf{a} \). From an application perspective it is important to derive an estimator based only on the observation model and the actual
observations. The maximum likelihood (ML) estimator is one such estimator which uses the observations and the observation model to find an estimate that maximizes the likelihood function \( l(a; y) \triangleq p(y; a) \). The likelihood function corresponding to the observation model in (5.3) can be written as

\[
l(a; y) = \int p(y|z; a)p(z)dz = \int \mathcal{N}(y; \Phi(X, Z)a, \sigma^2 I) \prod_{i=1}^{N} \mathcal{N}(z_i; 0, \sigma^2_z I)dz_i. \tag{5.8}
\]

We now have the following definition of the ML estimator:

\[
\hat{a}_{ML} = \arg \max_{a} l(a; y). \tag{5.9}
\]

Maximizing (5.8) is not straightforward and does not lead to an analytical solution. In [137] the EM algorithm is employed to obtain the ML estimate. EM is computationally expensive and may not be suitable for real-time embedded applications. In this thesis, I present an approximation to the observation model and the subsequent development of ILE and discuss its performance.

### 5.3 Iterative Linear Estimator (ILE)

Using the Taylor series expansion and keeping only the first order term, I obtain the following linearization of the observation model:

\[
y_i = f(x_i) + z_i^T \nabla f(x_i) + \eta_i, \tag{5.10}
\]

where \( \nabla f(x_i) \) is the gradient of the function calculated at \( x_i \). Note that for one-dimensional signals the linearized observation model takes the following simplified form

\[
y_i = f(x_i) + z_i f'(x_i) + \eta_i. \tag{5.11}
\]
Since the sum of two independent Gaussian random variables is Gaussian, we can write the marginal pdf of $y_1$ as

$$p(y_1; a) = \frac{1}{\sqrt{2\pi(\sigma^2 + ||\nabla f(x_i)||^2\sigma^2)}} \exp \left\{ -\frac{(y_1 - f(x_i))^2}{2(\sigma^2 + ||\nabla f(x_i)||^2\sigma^2)} \right\}. \quad (5.12)$$

For one-dimensional signals we replace $||\nabla f(x_i)||^2$ with $f'(x_i)^2$. Note that higher order Taylor expansion, for example $y_i = f(x_i) + z_i f'(x_i) + \frac{1}{2} z_i^2 f''(x_i) + \eta_i$, will lead to a complicated pdf of $y_i$ (square of Gaussian random variable is not Gaussian) and hence it will be difficult to obtain an analytical expression for the marginal pdf of $y_i$. Since all the observations are independent we have $p(y; a) = \prod_{i=1}^{N} p(y_i; a)$ and hence the approximated likelihood function $\hat{l}(a; y)$ can be written as

$$\hat{l}(a; y) = \frac{1}{\sqrt{(2\pi)^N \prod_{i=1}^{N}(\sigma^2 + ||\nabla f(x_i)||^2\sigma^2)}} \exp \left\{ -\sum_{i=1}^{N} \frac{(y_i - f(x_i))^2}{2(\sigma^2 + ||\nabla f(x_i)||^2\sigma^2)} \right\}. \quad (5.13)$$

My aim is to find an estimate $\hat{a}$ that maximizes $\hat{l}(a; y)$. Note that $f(x) = \sum_{k=1}^{M} a_k \phi_k(x)$ and $\nabla f(x) = \sum_{k=1}^{M} a_k \nabla \phi_k(x)$. The derivative of $\hat{l}(a; y)$ with respect to $a$ will still result in a nonlinear set of equations making the estimation problem difficult.

I now present an iterative strategy to obtain an estimate of $a$ using the linearized likelihood function, (5.13). In the absence of sample location error the resulting observation model is linear with additive i.i.d. Gaussian noise, i.e. $y = \Phi(X, 0)a + \eta$. The minimum variance unbiased estimator (MVUE) with the above observation model is

$$a_b = (\Phi(X, 0)^T \Phi(X, 0))^{-1} \Phi(X, 0)^T y. \quad (5.14)$$

I call the estimator defined in (5.14) the blind estimator (BE). Using $a_b$ I can obtain an initial estimate of the unknown signal as $f_b(x)$ using (5.7) and its gradient $\nabla f_b(x)$. 125
Using the approximation for the gradient I write (5.13) as

\[ \hat{l}(\mathbf{a}; \mathbf{y}) = \frac{1}{\sqrt{(2\pi)^N \prod_{i=1}^{N} (\sigma^2 + \| \nabla f_b(x_i) \|^2 \sigma_z^2)}} \exp \left\{ - \sum_{i=1}^{N} \frac{(y_i - f(x_i))^2}{2(\sigma^2 + \| \nabla f_b(x_i) \|^2 \sigma_z^2)} \right\}. \]  

(5.15)

I take the derivative of the approximated likelihood function (5.15) to obtain the following linear systems of equations:

\[ \frac{\partial \hat{l}(\mathbf{a}; \mathbf{y})}{\partial a_j} = 0 \]

\[ \Rightarrow \sum_{i=1}^{N} \frac{y_i \phi_j(x_i)}{\sigma^2 + \| \nabla f_b(x_i) \|^2 \sigma_z^2} = \sum_{i=1}^{N} a_k \sum_{i=1}^{N} \frac{\phi_k(x_i) \phi_j(x_i)}{\sigma^2 + \| \nabla f_b(x_i) \|^2 \sigma_z^2} \]  

(5.16)

for \( j = 1 \ldots M \).

I define

\[ \alpha_i = \sigma^2 + \| \nabla f_b(x_i) \|^2 \sigma_z^2; \]  

(5.17)

and

\[ C = \text{diag}(1/\alpha_1, 1/\alpha_2, \ldots, 1/\alpha_N). \]  

(5.18)

Solving the above system of linear equations I obtain ILE as follows:

\[ \mathbf{a}_{ILE} = (\Phi(X, 0)^T C \Phi(X, 0))^{-1} \Phi(X, 0)^T C \mathbf{y}. \]  

(5.19)

I implement the estimator in (5.19) iteratively by using the estimate \( \mathbf{a}^{ILE}_{t} \) obtained above to calculate \( \nabla f_b(x_i) \) using \( \nabla f(x) = \sum_{k=1}^{M} a_{ILE,k}^t \nabla \phi_k(x) \), where \( t \) is the current iteration index. \( \alpha_i \)s are subsequently recalculated which are then used to obtain a new estimate of \( \mathbf{a} \) using (5.19) as \( \mathbf{a}^{(t+1)}_{ILE} \). The procedure is repeated until convergence, i.e., \( \| \mathbf{a}^{(t+1)}_{ILE} - \mathbf{a}^{t}_{ILE} \| \leq \epsilon \), where \( \epsilon \) is the tolerance for convergence, or a maximum number of iterations have been reached. Algorithm 5.1 summarizes the steps involved, and the performance of the estimator is presented in Section 5.5.
Algorithm 5.1 Iterative Linear Estimator (ILE) for signal reconstruction under measurement and sampling location errors

Require: \( y, X, \{\phi_k(x)\}^M_{k=1}, \epsilon \) (convergence tolerance), \( P \) (maximum number of iterations)
Use (5.14) to find \( a_b \)
Find \( f^1_b(x) \) using (5.7) and compute \( \nabla f^1_b(x) \)
\( t \leftarrow 0 \)
\( \hat{a}^t = a_b \)
repeat
\( t \leftarrow t + 1 \)
Compute \( \alpha_i = \sigma^2 + \| \nabla f^t_b(x_i) \|^2 \sigma_x^2 \quad \forall i = 1 \ldots N \)
Create \( C^t = \text{diag}(1/\alpha_1, 1/\alpha_2, \ldots, 1/\alpha_N) \)
Compute \( \hat{a}^t \) from (5.19)
Using \( \hat{a}^t \) find \( f^{t+1}_b(x) \) using (5.7) and compute \( \nabla f^{t+1}_b(x) \)
until \( ||\hat{a}^t - \hat{a}^{t-1}|| \leq \epsilon \) or \( t \geq P \)
return \( \hat{a}^t \)

5.4 Cramér-Rao Lower Bound and Expectation-Maximization Algorithm

Since for a general choice of basis functions, no closed form solution exists to the likelihood function given by (5.8), a numerical method is needed to approximate (5.8). This numerical approximation is a precursor for either obtaining the CRB or implementing an EM approximation to the ML estimator (5.9) as discussed in [137]. In this section I first discuss techniques and issues involved in obtaining an approximation to (5.8). Second, on the basis of our numerical approximation scheme I present an approximation to the CRB and an EM algorithm for ML estimation. I present the formulation and simulation results for one-dimensional signals and highlight the challenges associated with higher dimensional signals.
5.4.1 Numerical Integration

Since the observations and the location errors are independent, (5.8) can be written as

\[
 l(a; y) = \prod_{i=1}^{N} \int_{z_i} \mathcal{N}(y_i; \phi_i(z_i)^T a, \sigma^2) \mathcal{N}(z_i; 0, \sigma^2 I) dz_i = \prod_{i=1}^{N} p(y_i; a). \quad (5.20)
\]

Numerical approximation of the marginal pdf of the observations \( y_i, p(y_i; a) \), involves estimating the \( D \) dimensional integral

\[
 p(y_i; a) = \int_{z_{i,1}}^{\cdots} \cdots \int_{z_{i,D}} \mathcal{N}(y_i; \phi_i(z_i)^T a, \sigma^2) \mathcal{N}(z_{i,1}; 0, \sigma^2) dz_{i,1} \cdots \mathcal{N}(z_{i,D}; 0, \sigma^2) dz_{i,D},
\]

where \( z_i = [z_{i,1}, z_{i,2}, \ldots, z_{i,D}]^T \) is the sample location error at the sampling point \( x_i \). Clearly the complexity of estimation grows with the dimension of the signal. For a one-dimensional signal the marginal has a simpler expression:

\[
 p(y_i; a) = \int_{z_i} \mathcal{N}(y_i; \phi_i(z_i)^T a, \sigma^2) \mathcal{N}(z_i; 0, \sigma^2) dz_i = \mathbb{E}_{z_i} [\mathcal{N}(y_i; \phi_i(z_i)^T a, \sigma^2)]. \quad (5.22)
\]

There are two major approaches to obtain a numerical approximation to the marginal pdf (5.22). The first method, called quadrature integration, approximates the weighted integral with a weighted sum [60], i.e.,

\[
 p(y_i; a) \approx \sum_{j=1}^{J} p(y_i|z_{i,j}, a) w_j, \quad (5.23)
\]

where \( z_{i,j} \)'s are the abscissas, \( w_j \)'s are the weights, \( J \) is the number of points for the quadrature integration and \( p(y_i|z_{i,j}, a) = \mathcal{N}(y_i; \phi_i(z_{i,j})^T a, \sigma^2) \). Since \( z_i \) is normally distributed, the appropriate choice of abscissas and weights are derived through the Gauss-Hermite quadrature (GHQ) rule [60]. Using a Cartesian product of the one-dimensional quadrature grid for evaluating multidimensional integrals leads to an exponential increase in the computational complexity (number of function evaluations \( \sim J^D \)). Though use of sparse grids provides improved performance, the complexity...
still increases significantly with the dimension [57].

The second method, Monte Carlo integration (MCI) approximates the integration by generating $S$ samples of $z_i$ from its distribution and using the samples $(z_{i,s})$ to approximate the integral as

$$p(y_i; a) \approx \frac{1}{S} \sum_{s=1}^{S} p(y_i|z_{i,s}, a). \quad (5.24)$$

It is known that the approximation error is proportional to $\text{var}(f)/\sqrt{S}$, where $\text{var}(f)$ is the variance of the function to be integrated. Though the error is independent of the dimension of the problem [134], the variance ($\text{var}(f)$) can grow exponentially with the dimension, requiring an exponential increase in the number of samples for achieving a desired accuracy. Use of quasi-Monte Carlo integration can further improve the performance for higher dimensional integrals [134].

Eqn. 5.25 shows an example where the variance grows exponentially with the dimension, requiring an exponential increase in the number of samples for achieving a desired accuracy.

$$E[\sin(x) \sin(y)]$$

$$= \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \sin(x) \sin(y) \mathcal{N}([x, y]^T; [0, 0]^T, \sigma^2 I) dxdy$$

$$= 0$$

$$\text{var}[\sin(x) \sin(y)]$$

$$= \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \sin^2(x) \sin^2(y) \mathcal{N}([x, y]^T; [0, 0]^T, \sigma^2 I) dxdy$$

$$= \left( \int_{-\infty}^{+\infty} \sin^2(x) \mathcal{N}(x; 0, \sigma^2) dx \right)^2. \quad (5.25)$$

I now present an example of comparing the performance of GHQ and MCI in computing the numerical approximation of the marginal pdf (5.22) for a one-dimensional signal. I choose $1, \sin(kx)$ and $\cos(kx)$ as the basis functions for a specific frequency,
and create an orthonormal set \( \{\phi_1(x), \phi_2(x), \phi_3(x)\} \) from them [78]. I choose \( \mathbf{a} = [0.5, 1, -1]^T \) and \( f(x) = 0.5\phi_1(x) + \phi_2(x) - \phi_3(x) \). For a specific sampling location, say \( x_i = 0 \) and \( k = \{1, 5, 10\} \) I evaluate the performance of GHQ and MCI. I first generate \( 10^5 \) samples of \( y_i \) according to (5.3) and generate histograms to obtain an empirical estimate of the pdf of \( y_i \). I use an open source code [26] to implement GHQ (5.23) with \( J = 128 \). For MCI, I first generate \( 10^4 \) samples of \( z_i \) by sampling from a zero mean normal distribution and given standard deviation \( (\sigma_z) \) and estimate the pdf by (5.24). Fig. 5-1(a), (b) and (c) show that as the frequency of the signal increases the MCI outperforms GHQ. Similarly, Fig. 5-1 (a) and (d) show that for the same frequency, \( k = 1 \), as the sample location error \( (\sigma_z) \) increases, MCI provides a better approximation of the pdf compared to GHQ. In the rest of the thesis I employ MCI for approximating marginal pdfs and expectations.

Figure 5-1: Comparison of the histogram of the sample \( y_i \) (Hist) with numerical approximation of the pdf through Gauss-Hermite quadrature (GHQ) and Monte Carlo integration (MCI); \( \sigma = 0.053 \).
5.4.2 Cramér-Rao Lower Bound and its Numerical Approximation

The CRB provides a lower bound on the variance of any unbiased estimator. The CRB(a) is defined as the trace of the inverse of the Fisher information matrix $I_y(a)$. Under certain regularity conditions [75], $I_y(a)$ satisfies the following:

$$I_y(a) \triangleq E \left[ \left( \frac{\partial \log l(a; y)}{\partial a} \right) \left( \frac{\partial \log l(a; y)}{\partial a} \right)^T \right] = -E \left[ \frac{\partial^2 \log l(a; y)}{\partial a \partial a^T} \right], \quad (5.26)$$

where the expectation is taken with respect to $p(y; a)$ (5.20). Using (5.20), $\log l(a; y) = \sum_{i=1}^{N} \log p(y_i; a)$, hence one can write (5.26) as

$$I_y(a) = \sum_{i=1}^{N} E \left[ \left( \frac{\partial \log p(y_i; a)}{\partial a} \right) \left( \frac{\partial \log p(y_i; a)}{\partial a} \right)^T \right]. \quad (5.27)$$

Now $\frac{\partial \log p(y_i; a)}{\partial a} = \frac{1}{p(y_i; a)} \frac{\partial p(y_i; a)}{\partial a}$ where I compute $p(y_i; a)$ from (5.24).

I can write

$$\frac{\partial p(y_i; a)}{\partial a} \approx \frac{1}{S} \sum_{s=1}^{S} \left\{ \frac{(y_i - \phi_1(z_{i,s})^T a) \phi_1(z_{i,s})}{\sigma^2} N(y_i; \phi_1(z_{i,s})^T a, \sigma^2) \right\} \quad (5.28)$$

$$\Rightarrow \frac{\partial \log p(y_i; a)}{\partial a} \approx \frac{1}{S} \sum_{s=1}^{S} \frac{(y_i - \phi_1(z_{i,s})^T a) \phi_1(z_{i,s})}{\sigma^2} N(y_i; \phi_1(z_{i,s})^T a, \sigma^2)$$

$$\triangleq d(y_i). \quad (5.29)$$

Eqn. (5.27), (5.29) and MCI gives

$$I_y(a) \approx \sum_{i=1}^{N} E \left[ d(y_i) d(y_i)^T \right] \approx \frac{1}{S} \sum_{i=1}^{N} \sum_{s=1}^{S} d(y_{i,s}) d(y_{i,s})^T, \quad (5.30)$$

131
where \( y_{i,s} \) are generated from \( p(y_{i}; a) \).

Hence for any unbiased estimator \( \hat{a} \) we have

\[
\text{var}(\hat{a}) \geq CRB(a) = \text{trace}(I_y(a)^{-1}). \tag{5.31}
\]

### 5.4.3 An Expectation-Maximization Algorithm for Maximum Likelihood Estimation

EM is a classical approach for approximating ML estimators in the presence of incomplete or unobserved data [37]. The observation vector \( y \) in (5.8) depends on the unobserved location errors, \( z \). Instead of implementing (5.9), EM maximizes the following lower bound of the likelihood function at the current estimate \( a^{(t-1)} \):

\[
Q(a; a^{(t-1)}) = E[log p(y, z; a)|y; a^{(t-1)}]. \tag{5.32}
\]

Under certain regularity conditions the EM algorithm is guaranteed to converge to a local maximum of the likelihood function (5.8) [137], [37]. Using (5.8) we can write

\[
\log p(y, z; a) = -\frac{1}{2\sigma^2} \|y - \Phi(X, Z)a\|^2 - \frac{1}{2\sigma_z^2} \|z\|^2 + K, \tag{5.33}
\]

where \( K \) is a constant term not relevant to the optimization problem. Substituting (5.33) into (5.32) gives

\[
Q(a; a^{(t-1)}) = -\frac{1}{2\sigma^2} \|y\|^2 - 2y^T E[\Phi(X, Z)|y; a^{(t-1)}] a + a^T E[\Phi(X, Z)^T \Phi(X, Z)|y; a^{(t-1)}] a \\
- \frac{1}{2\sigma_z^2} E[\|z\|^2|y; a^{(t-1)}] + K. \tag{5.34}
\]

Since (5.34) is quadratic in \( a \), taking its derivative with respect to \( a \) and equating it to 0 gives the following system of linear equations:

\[
E[\Phi(X, Z)^T \Phi(X, Z)|y; a^{(t-1)}] a = E[\Phi(X, Z)|y; a^{(t-1)}]^T y. \tag{5.35}
\]
Hence the EM estimate of $a$ at $t$ is

$$\hat{a}^t = E \left[ \Phi(X, Z)^T \Phi(X, Z) | y; a^{(t-1)} \right]^{-1} E \left[ \Phi(X, Z) | y; a^{(t-1)} \right]^T y. \quad (5.36)$$

As discussed in [137], we can separate the expectations in (5.35) into univariate expectations:

$$E \left[ \Phi(X, Z)^T \Phi(X, Z) | y; a^{(t-1)} \right] = \sum_{i=1}^{N} E \left[ \phi_i(z_i) \phi_i(z_i)^T | y; a^{(t-1)} \right], \quad (5.37)$$

$$E \left[ \Phi(X, Z) | y; a^{(t-1)} \right] y = \sum_{i=1}^{N} E \left[ \phi_i(z_i) | y; a^{(t-1)} \right] y_i. \quad (5.38)$$

The expectations are with respect to $p(z | y; a^{(t-1)}) = \prod_{i=1}^{N} p(z_i | y_i; a^{(t-1)})$. Hence each univariate expectation is calculated with respect to

$$p(z_i | y_i; a^{(t-1)}) = p(y_i | z_i; a^{(t-1)}) p(z_i) / p(y_i; a^{(t-1)}) \quad (5.39)$$

where $p(y_i | z_i; a^{(t-1)}) = \mathcal{N}(y_i; \phi_i(z_i)^T a^{(t-1)}, \sigma^2)$. Given $a^{(t-1)}$ we compute $p(y_i; a^{(t-1)})$ from (5.24). Each univariate expectation is of the form

$$\int_{z_i} g(z_i) \mathcal{N}(y_i; \phi_i(z_i)^T a^{(t-1)}, \sigma^2) \mathcal{N}(z_i; 0, \sigma_z^2) dz_i, \quad (5.40)$$

which can be approximated by MCI. We have the following approximations to the expectations in (5.37) and (5.38):

$$E \left[ \Phi(X, Z)^T \Phi(X, Z) | y; a^{(t-1)} \right] \approx \sum_{i=1}^{N} \frac{1}{Sp(y_i; a^{(t-1)})} \sum_{s=1}^{S} \phi_i(z_{i,s}) \phi_i(z_{i,s})^T p(y_i | z_{i,s}; a^{(t-1)}), \quad (5.41)$$

$$E \left[ \Phi(X, Z) | y; a^{(t-1)} \right] y \approx \sum_{i=1}^{N} \frac{y_i}{Sp(y_i; a^{(t-1)})} \sum_{s=1}^{S} \phi_i(z_{i,s}) p(y_i | z_{i,s}; a^{(t-1)}). \quad (5.42)$$
Equation (5.36) is then employed to obtain the current estimate \( \mathbf{a}^t \). The aforementioned steps are repeated until convergence, i.e., \( \| \mathbf{a}^t - \mathbf{a}^{(t-1)} \| \leq \epsilon \) or a maximum number of iterations have been reached. Algorithm 5.2 summarizes the steps involved in the EM implementation of ML estimation.

### Algorithm 5.2 Expectation-Maximization for maximum likelihood estimation of signal coefficients under measurement and sampling location errors

**Require:** \( y, X, \{\phi_k(x)\}_{k=1}^M \), \( \epsilon \) (convergence tolerance), \( P \) (maximum number of iterations)

Use (5.14) to find \( \mathbf{a}_b \)

\[ t \leftarrow 0 \]

\[ \hat{\mathbf{a}}^t = \mathbf{a}_b \]

**repeat**

\[ t \leftarrow t + 1 \]

Compute \( p(y_i; \hat{\mathbf{a}}^{(t-1)}) \) by (5.24)

Approximate \( E [\Phi(X, Z)^T \Phi(X, Z) | y; \hat{\mathbf{a}}^{(t-1)}] \) by (5.41)

Approximate \( E [\Phi(X, Z) | y; \hat{\mathbf{a}}^{(t-1)}]^T y \) by (5.42)

Find \( \hat{\mathbf{a}}^t \) using (5.36)

**until** \( \| \hat{\mathbf{a}}^t - \hat{\mathbf{a}}^{(t-1)} \| \leq \epsilon \) or \( t \geq P \)

**return** \( \hat{\mathbf{a}}^t \)

### 5.5 Numerical Simulations

In this section I present simulation results on the performance of ILE. ILE is not an unbiased estimator, so I study its bias and variance apart from the MSE. I perform simulations with both one- and two-dimensional basis functions and present performance in terms of the MSE and computational time. I also present a detailed argument for the stability of ILE. I then present comparisons of the ILE with the CRB for unbiased estimators and the EM approximation of the ML estimator for one-dimensional basis functions highlighting the computational advantage of my algorithm. Note that Matlab was used for all the simulations and the parallel processing toolbox was used when multiple simulations were needed for a given set of parameters.
5.5.1 Performance of ILE

I implement numerical simulations with two classes of basis functions. Equation (5.14) presents the baseline blind linear estimator against which I compare the performance of ILE. In all the simulations I generate $x_i$s on a uniform grid. For one-dimensional signals I place $N$ samples at an equidistant spacing of $\Delta x = 2/N$ and $x_i = (i - 0.5)\Delta x$ for $i = 1, \ldots, N$. Similarly for two-dimensional simulations I have samples at $[(i - 0.5)\Delta x, (j - 0.5)\Delta y]^T$ for $i = 1, \ldots, N$, $j = 1, \ldots, N$, $\Delta x = \Delta y = 2/N$ and the total number of samples $= N^2$.

I now describe the parameters used for simulation studies. As mentioned before, the measurement errors are assumed to be i.i.d. Gaussian variables with known variance. Here I assume the variance to be a certain fraction of the maximum value of the function, i.e. $\sigma = (\gamma/100) \max |f(x)|$. Similarly I assume the variance of the position error to be a fraction of the spacing between the sampling points, i.e. $\sigma_z = (p/100)\Delta x$. Though the sample location error was assumed to have a Gaussian distribution, in simulations I used the following hard thresholding on the values of $z_i$s: $z_i = \text{sign}(z_i) \min\{|z_i|, \Delta x/2\}$. This ensures that the sample location, say,
Figure 5-3: Comparison of the mean of the squared estimation error $||\hat{a} - a||^2$ with ILE, BE and KLE for polynomial basis functions. The error bars indicate 1/2 standard deviations. (a) Comparison of the performance with the variation in $p$. Solid line corresponds to $\gamma = 1$, dashed line $\gamma = 3$, dotted line $\gamma = 5$ and for all the simulations $N = 100$; (b) Comparison of the performance of ILE against BE with the variation in $N$. The percentage sample location error ($p$) is indicated on the left of each set of plots and $\gamma = 1$.

$x_i + z_i$ is lower than $x_{i+1} + z_{i+1}$ and I have an ordered set of samples. Hence, in simulations, a higher $\sigma_z$ will lead to a distribution close to a uniform distribution on $[-\Delta x/2, \Delta x/2]$.

For one-dimensional simulations I first have $\psi_0(x) = 1$, $\psi_k(x) = \sin(kx)$ for $k = 1, \ldots, 5$, and $\psi_k(x) = \cos((k - 5)x)$ for $k = 6, \ldots, 10$. I use Gram-Schmidt orthonormalization [78] to generate 11 orthonormal basis functions $\{\phi_k(x)\}_{k=0}^{10}$ and call it the set of trigonometric basis functions. Similarly, I create an orthonormal set of 10 polynomial basis functions from $\psi_k(x) = x^k$ for $k = 0, \ldots, 9$. For each combination of the parameters, $p$, $\gamma$ and $N$, 1000 simulations were carried out in a parallel fashion using Matlab’s ‘parfor’. For trigonometric basis functions $||a|| = 2.917$, for polynomial basis functions $||a|| = 4.283$ and I choose $\epsilon = 10^{-4}, P = 200$. I compute the mean and the standard deviation of the squared estimation error ($||\hat{a} - a||^2$). I compare the squared estimation error with ILE, the blind estimator (5.14) and the following least square estimator with known sample location error:

$$\hat{a}_{KLE} = (\Phi(X, Z)^T \Phi(X, Z))^{-1} \Phi(X, Z)^Ty.$$  \hspace{1cm} (5.43)
Fig. 5-2 presents the performance of the ILE for trigonometric basis functions and Fig. 5-3 for polynomial basis functions. Fig. 5-2a and Fig. 5-3a show that as the sample location error, \( p \), is increased the ILE performs better compared to the *blind* estimator. The estimation error with ILE is comparable to BE for low values of \( p(\sigma_z) \), but with increase in \( p \), the difference between the MSE with ILE and BE increases. As expected, Fig. 5-2b and Fig. 5-3b show that the estimation error decreases with increase in \( N \). Interestingly, the difference between estimation error with ILE and BE is higher for lower \( N \) and at higher \( N \) the difference between them is negligible. As expected, the estimate from (5.43) does not depend on sample location error (\( \sigma_z \)).

The following is the relationship of the MSE in terms of bias and variance:

\[
E [\|\hat{a} - a\|^2] = \sum_{k=1}^{M} E [(\hat{a}_k - a_k)^2]
\]

\[
= \sum_{k=1}^{M} (E [(\hat{a}_k - E[\hat{a}_k])^2] + (E[\hat{a}_k] - a_k)^2)
\]

\[
= \sum_{k=1}^{M} (\text{var}(\hat{a}_k) + \text{bias}_k^2)
\]

\[
= \text{variance} + \text{bias}^2.
\] (5.44)

I use the 1000 simulations for each parameter set and compute the bias and the variance of ILE. I define \( \Lambda = \text{bias}/\sqrt{\text{variance}} \). Fig. 5-4 plots the variation of \( \Lambda \) for trigonometric basis functions with the variation in \( p \) for various combinations of \( \gamma \) and \( N \). The contribution of bias to the overall estimation error increases with \( p \) and the effect is more pronounced at higher \( N \) and at lower \( \gamma \). Recall that \( \sigma = (\gamma/100) \max |f(x)| \) and hence higher \( \gamma \) implies higher additive noise. A similar variation of \( \Lambda \) is observed with polynomial basis functions.

For polynomial basis functions, Fig. 5-5 shows the variation of the mean computational time for my algorithm with the variation in simulation parameters (\( p, \gamma \) and \( N \)). The mean computational times increase with \( p \) and are similar for different values of \( \gamma \). Furthermore, an interesting trend is observed with \( N = 50 \), where there is a sharp increase in computational time with \( p \). A similar trend is observed with
trigonometric basis functions. Note that the computational times are of the order of $10^{-2}$ s. Typically the number of iterations for convergence is around 5 except for $N = 50$ where at high $p$ the number of iterations required is around 30.

Similarly, I create a set of two-dimensional orthonormal polynomial basis functions from the following set of basis functions:

$$
\psi = \{1, x_1, x_2, x_1 x_2, x_1^2, x_2^2, x_1 x_2, x_1 x_2^2, x_1^2 x_2, x_1^3 x_2^3\}. \quad (5.45)
$$

I have $||a|| = 2.398$. Fig. 5-6 shows a trend similar to the one-dimensional case where the ILE outperforms BE with increase in $p(\sigma_z)$ and the difference between ILE and BE decreases with $N$. The mean computational times are of the order of 1 s and show trends similar to the one-dimensional signals. Though the computational time increased by two orders of magnitude, our method is still suitable for estimating a slowly varying higher dimensional field.

### 5.5.2 Convergence and Stability of ILE

In this section I show the numerical stability of ILE, which ensures that the estimate does not diverge with iterations. I first assume that the $N \times M$ matrix $\Phi(X, 0)$ is of

![Figure 5-4: Comparison of $\Lambda$ with the variation in $p$.](image-url)
Figure 5-5: Variation of the mean of the computational time for ILE with % sample location error ($p$) for different % additive noise ($\gamma$) and the number of sampling points ($N$).

Figure 5-6: Comparison of the mean of the squared estimation error $||\hat{a} - a||^2$ with ILE, BE and KLE for two-dimensional polynomial basis functions. The error bars indicate 1/2 standard deviations. (a) Comparison of the performance with the variation in $p$. Solid line corresponds to $\gamma = 1$, dashed line $\gamma = 3$, dotted line $\gamma = 5$ and for all the simulations $N = 25$; (b) Comparison of the performance of ILE against BE with the variation in $N$. The percentage sample location error ($p$) is indicated on the left of each set of plots and $\gamma = 1$.

rank $M$. This is a reasonable assumption given I have orthonormal basis functions, I sample at distinct locations, and $N \geq M$. Note that if the aforementioned assumption does not hold, then the blind estimator (5.14) does not exist and ILE terminates at the initialization (Algorithm 5.1).

I denote the singular values of $\Phi(X, 0)$ by $\{s_k\}_{k=1}^M$ with $s_1 \geq s_2 \geq \ldots \geq s_M > 0$. At the current iteration, $t$, I have $C^t$ given by (5.18). I obtain $\nabla f^t_i(x_i)$ from the previous
estimate \( \hat{a}^{(t-1)} \) by \( \nabla f_b(x_i) = \sum_{k=1}^{M} \hat{a}_k^{(t-1)} \nabla \phi_k(x_i) \). I define \( \beta_k = \max_{x \in Q} \| \nabla \phi_k(x) \| \). I have \( \| \beta \| = \sum_{k=1}^{M} \beta_k^2 \) and \( \| \beta \| < \infty \) since the basis functions are continuous and differentiable and the domain \( Q \) is closed and bounded. Cauchy-Schwarz inequality gives the following relationship:

\[
\| \nabla f_b(x) \| = \sqrt{\sum_{j=1}^{D} \left( \sum_{k=1}^{M} \hat{a}_k^{(t-1)} \frac{\partial \phi_k(x)}{\partial x_j} \right)^2} \leq \sqrt{\sum_{j=1}^{D} \left( \sum_{k=1}^{M} (\hat{a}_k^{(t-1)})^2 \sum_{k=1}^{M} \left( \frac{\partial \phi_k(x)}{\partial x_j} \right)^2 \right)} = \| \hat{a}^{(t-1)} \| \sqrt{\sum_{k=1}^{M} \sum_{j=1}^{D} \left( \frac{\partial \phi_k(x)}{\partial x_j} \right)^2} = \| \hat{a}^{(t-1)} \| \sqrt{\sum_{k=1}^{M} \| \nabla \phi_k(x) \|^2} \leq \| \hat{a}^{(t-1)} \| \| \beta \|. \quad (5.46)
\]

Hence for \( i = 1, \ldots, N \), \( \| \nabla f_b(x_i) \| < \| \hat{a}^{(t-1)} \| \| \beta \| \). According to (5.14), I initialize ILE with \( a^0 = a_b \) and have:

\[
\| \hat{a}^0 \| = \| (\Phi(X,0)^T \Phi(X,0))^{-1} \Phi^T(X,0)y \| \leq \| (\Phi(X,0)^T \Phi(X,0))^{-1} \| \| \Phi^T(X,0)y \| = \frac{1}{s^2} s_1 \| y \| < \infty. \quad (5.47)
\]

For the first iteration of ILE \( (t = 1) \), every diagonal entry of \( C^1 \) satisfies:

\[
C^1_{i,i} = \frac{1}{\sigma^2 + \| \nabla f_b(x_i) \|^2 \sigma_x^2} \implies \frac{1}{\sigma^2 + \| \hat{a}^0 \| \| \beta \| \sigma_x^2} \leq C^1_{i,i} \leq \frac{1}{\sigma^2}. \quad (5.48)
\]

The following relationships is obtained by using (5.47) in (5.48):

\[
0 < \frac{s^2_M}{\sigma^2 s^2_M + s_1 \| y \| \| \beta \| \sigma_x^2} \leq C^1_{i,i} \leq \frac{1}{\sigma^2}. \quad (5.49)
\]

Note that the above lower bound of \( C^1_{i,i} \) holds true for any finite norm initialization of ILE, \( i.e., \| \hat{a}^0 \| < \infty \). Since \( C^1 \) is a diagonal matrix with all its diagonal entries greater than zero, \( C^1 \) is positive definite, \( i.e., C^1 \succ 0 \). I now make use of the following
Lemma 5.1. Let $A$ be an $m \times n$ matrix of full row rank, and let $D_+$ denote the set of positive definite diagonal $n \times n$ matrices. Then,

$$
\sup_{D \in D_+} \| (ADA^T)^{-1} AD \| = \max_{J \in \mathcal{J}(A)} \| A_J^{-1} \|,
$$

where $\mathcal{J}(A)$ is the collection of sets of column indices associated with nonsingular $m \times m$ submatrices $(A_J)$ of $A$.

After the first iteration of ILE, $\| \hat{a}^1 \|$ satisfies the following:

$$
\hat{a}^1 = (\Phi(X, 0)^T C^1 \Phi(X, 0))^{-1} \Phi(X, 0)^T C^1 y \\
\implies \| \hat{a}^1 \| \leq \| (\Phi(X, 0)^T C^1 \Phi(X, 0))^{-1} \Phi(X, 0)^T C^1 \| \| y \| \\
\leq \max_{J \in \mathcal{J}(\Phi(X, 0))} \| \Phi(X, 0)_J^{-1} \| \| y \| = G \| y \| < \infty,
$$

(5.50)

where $\mathcal{J}(\Phi(X, 0))$ is the collection of sets of row indices associated with nonsingular $M \times M$ submatrices of $\Phi(X, 0)$ and $G = \max_{J \in \mathcal{J}(\Phi(X, 0))} \| \Phi(X, 0)_J^{-1} \|$ and is finite.

Since $\| \hat{a}^1 \|$ is finite, from (5.46), (5.48), (5.50) and $t = 2$, we have $\| \nabla f_{k}^2(x_i) \| \leq G \| y \| \| \beta \|$, $C^2 > 0$ and hence $\| \hat{a}^2 \| \leq G \| y \|$. Extending the argument in (5.48) and (5.50) to future iterations, I can claim that $C^t > 0$ and $\| \hat{a}^t \| \leq G \| y \| \forall t$. Hence, ILE is a stable estimator and the estimate is guaranteed to have a finite norm.

ILE may however converge to a different $\hat{a}$ with a different initialization. Numerically, I studied the variance in the convergence of ILE by random initialization of $a$ from a zero mean Gaussian distribution with a standard deviation of 100. Fig. 5-7 shows the standard deviation of the converged $\| \hat{a} \|$ with 1000 different random initializations for different values of $p$, $\gamma$ and $N = \{100, 200\}$. We observe that the standard deviation is acceptable and is within 6% of $\| a \|$ even at high levels of noise ($\gamma = 10, p = 100$).
5.5.3 Comparison of ILE with CRB and EM algorithm

I perform simulations with the one-dimensional trigonometric basis functions as described before to compare the performance of ILE with the CRB and the EM approximation of the ML estimator. The CRB gives the minimum achievable variance of an unbiased estimator. On the other hand EM is a state of the art approach to implement ML estimation for problems with missing data [37].

Since EM involves approximation of integrals through MCI (Section 5.4.3), computational times are high. As discussed in Section 5.4.1, the number of computations and hence computational time may scale up exponentially with the dimension of the signal. Here I only present comparison with a one-dimensional signal and I perform 250 simulations (as opposed to 1000) for each combination of the simulation parameters for both ILE and EM. Fig. 5-8 shows the variation of the moving average of the squared estimation error with the iteration number for seven randomly chosen simulation settings. In all the cases it is observed that the moving average reaches close to its final value after around 250 simulations implying that 250 iterations are sufficient to quantify the mean performance of ILE.

Fig. 5-9 compares the squared estimation error with ILE, BE (5.14) and EM (Section 5.4.3) with the variation in $p$. For all the simulations, I choose trigonometric basis functions as described in 5.5.1 and $\|a\| = 5.823$. I further compare the squared estimation errors with the CRB (Section 5.4.2) which is the minimum
Figure 5-8: Variation of the moving average of the squared estimation error with the iteration number for seven randomly chosen simulation settings with the one-dimensional polynomial basis functions.

Figure 5-9: Comparison of the mean of the squared estimation error $||\hat{a} - a||^2$ with ILE, BE and EM with the variation in $p$. The error bars indicate 1/2 standard deviations. The approximated CRB is presented by the black line. For all the simulations $N = 50$.

achievable variance of any unbiased estimator. For low additive noise ($\gamma = 1$), ILE outperforms EM while at higher noise ($\gamma = 5$), the EM algorithm gives a lower estimation error. Interestingly, Fig. 5-9a shows that as $p$ increases, EM performs worse when compared to BE. The EM algorithm uses the marginal and conditional probabilities of the observations $y_i$s to iteratively calculate an estimate of $a$. The conditional probability $p(y_i|z_i; a^{(t-1)}) = \mathcal{N}(y_i; \phi_i(z_i)^T a^{(t-1)}, \sigma^2)$ is a function of the noise level ($\sigma = (\gamma/100) \max |f(x)|$). The normally distributed conditional probability $p(y_i|z_i; a^{(t-1)})$ gets narrower and eventually assumes a Kronecker $\delta$ function as $\gamma \to 0$. This causes higher numerical errors when conditional expectations are calculated (Section 5.4.3). Fig. 5-10 shows the marginal distribution of $y_i$ calculated...
through MCI (5.24) with different values of $\gamma$ for one-dimensional trigonometric basis functions with $||a|| = 3.4528$ and $x_i = 0$. Since EM involves numerical evaluation of expectations that involve $\mathcal{N}(y_i; \phi_i(x_i)^T a^{(t-1)}, \sigma^2)$, the numerical error in the calculation of these expectations increases with decreasing $\gamma$, worsening the performance of EM with decreasing noise. On the other hand, ILE does not suffer from such problems making it a more suitable choice for signal estimation under low noise levels or high signal to noise ratio (SNR).

![Figure 5-10](image)

Figure 5-10: Marginal probability distribution of $y_i$ for one-dimensional trigonometric basis functions with $||a|| = 3.4528$ and $x_i = 0$.

The mean computational time for EM is of the order of $10^3$ s while for ILE is of the order of $10^{-2}$ s. Fig. 5-11 shows the variation in the squared estimation error with ILE and EM for $N = 50$ and $N = 100$. It also shows the comparison with CRB. For a lower number of samples ILE outperforms EM with increase in $p$ and for a higher number of samples their performance is comparable. Note that here the unknown vector $(a)$ is 11 dimensional, so $N = 50$ refers to a reasonable level of oversampling while $N = 100$ implies high oversampling. Hence for low additive noise and a reasonable level of oversampling, ILE clearly outperforms EM in both the MSE and the computational time. Even for higher additive noise and number of samples, ILE is comparable to EM.
with a significant improvement in the computational cost making ILE an appealing candidate for real-time embedded applications such as mobile sensor networks for environmental monitoring.

![Comparison of the mean of the squared estimation error with ILE and EM as a function of N. The error bars indicate 1/2 standard deviations. The approximated CRB is presented by the black line. For all the simulations $\gamma = 1$.](image)

### 5.6 A Joint Maximum-a-Posteriori (JMAP) Estimator

The known distribution of $z_i$s can be used to develop a joint maximum-a-posteriori estimator [140], [121], [91] which obtains an estimate of the unknown parameters $a$ and the unknown locations $z$ as:

$$ (\hat{a}, \hat{z}) = \arg \max p(y|a, z)p(z). $$

(5.51)

Note that (5.51) is significantly different from the formulation of (5.8). Again, the optimization problem in (5.51) does not have an analytical solution and an iterative alternating scheme can be used to solve it.

An iterative strategy that alternates between estimating $a$ and $z$ gives

$$ a^i = \arg \max p(y|a, \hat{z}^{i-1}), $$

(5.52)
which is easy to compute and is just the least squares estimator. The objective function $G(z|a^{t-1})$ is defined as

$$G(z|a^{t-1}) = -\log(p(y|a^{t-1}, z)p(z)) = \sum_{i=1}^{N} \left( \frac{y_i - f_{i-1}(x_i + z_i)^2}{2\sigma^2} \right) + \frac{z_i^2}{2\sigma_z^2}, \quad (5.53)$$

where $f_{i-1}(x) = \sum_{k=1}^{M} a_{k-1}^t \phi_k(x)$ and an estimate of $z$ is obtained by

$$\hat{z}^t = \arg\min G(z|a^{t-1}). \quad (5.54)$$

Since $G(z|a^{t-1})$ is non-linear in $z$, the optimization problem is difficult to solve. A quadratic approximation of $G(z|a^{t-1})$ can be obtained using a Taylor series expansion as:

$$\hat{G}(z|a^{t-1}) = \sum_{i=1}^{N} \left( \frac{y_i - f_{i-1}(x_i) - z_i f'_{i-1}(x_i)^2}{2\sigma^2} \right) + \frac{z_i^2}{2\sigma_z^2}. \quad (5.55)$$

Note that the quadratic approximation is not a guaranteed upper bound of the original objective function, but a bound on $|G(z|a^{t-1}) - G(z|a^{t-1})|$ can be obtained in terms of upper bounds on $||a||$, $||z||$ and the derivatives of the basis functions using the Taylor remainder theorem. Taking the derivative of $\hat{G}(z|a^{t-1})$ and equating it to zero gives

$$\frac{\partial \hat{G}(z|a^{t-1})}{\partial z_j} = -\frac{(y_j - f_{i-1}(x_j) - z_j f'_{i-1}(x_j)) f'_{i-1}(x_j)}{\sigma^2} + \frac{z_j}{\sigma_z^2} = 0 \quad (5.56)$$

$$\Rightarrow z_j^t = \frac{(y_j - f_{i-1}(x_j)) f'_{i-1}(x_j)\sigma_z^2}{\sigma^2 + (f'_{i-1}(x_j))^2\sigma_z^2},$$

for $j = 1 \ldots N$.

As in the case of ILE, I repeat the above calculations till $||a^t - a^{t-1}|| \leq \epsilon$ and $||z^t - z^{t-1}|| \leq \epsilon$ and call the final estimate of $a$ as the JMAP estimate. Fig. 5-12 compares the performance of the JMAP estimator with ILE. For $N = 50$ JMAP estimation fails to provide a good estimate while at higher $N$ it performs comparable to ILE at low $p$ and the performance improves with increase in $p$. Note that the computational times were of the order of the ones for ILE. In the maximum-a-posteriori formulation we have $N$ observation for the $N + M$ unknowns. This makes the problem more difficult.
to solve and may inherently be more non-linear (higher number of local extrema) compared to a maximum likelihood formulation. For future work I would like to study the connection between the JMAP estimator and ILE. Both these methods solve a sequence of convex approximations to the original optimization problem. I would like to understand the difference between the JMAP estimator and ILE as a function of basis function set, sampling density and noise levels. This can have a direct impact on the appropriate selection of an estimator for a particular mobile sensor deployment and field estimation mission.

![Graph](image.png)

Figure 5-12: Comparison of the mean of the squared estimation error \( \| \hat{a} - a \|^2 \) with ILE, JMAP and BE with the variation in \( p \). The error bars indicate 1/2 standard deviations.

### 5.7 Conclusion

In this chapter I addressed the problem of estimating a parametric signal from samples with location errors and independent additive noise. As interest in the use of mobile sensor platforms for environmental monitoring increases, addressing the effect of localization errors on the estimation of spatio-temporal fields will become increasingly important. I formulated the signal estimation problem as finding deterministic parameters by representing the signal in the span of a finite basis. The parameter estimation was then formulated as finding the maximum likelihood estimator. I derived an approximation of the likelihood function through linearizing the obser-
vation model and then employed an iterative strategy to develop a computationally
efficient iterative linear estimator. I highlighted practical issues in computing the
numerical approximation of the complete likelihood function and showed the suit-
ability of Monte Carlo integration over Gauss-Hermite quadrature for our problem.
Using MCI I derived a numerical scheme for finding the Cramér-Rao lower bound
and an Expectation-Maximization algorithm to implement the maximum likelihood estimation. I presented simulations to show the suitability of the proposed algorithm
in improving the estimation error at low computational cost and also presented a
detailed argument for the stability of ILE. When compared with the estimator with
no information (BE), ILE gives a lower estimation error and its performance increases
with the variance of the sample location error ($p$) but decreases with the number of
sampling points ($N$). I numerically demonstrated that the contribution of the bias in
the overall estimation error increases with $p$ and $N$ but decreases with the variance
of the additive Gaussian noise ($\gamma$). The mean computational time for ILE for the
simulated one-dimensional signal is of the order of $10^{-2}$ s which increases to 1 s for
the simulated two-dimensional signal. For a one-dimensional signal, I showed that
for low additive noise and a reasonable level of oversampling, ILE outperforms EM
while at higher additive noise and higher number of samples EM algorithm gives a
lower estimation error. I further showed that the performance of EM worsens with
decreasing additive noise due to high numerical errors associated with computing in-
tegrals that involve narrow Gaussian distributions, while ILE does not suffer from
such issues. Furthermore, ILE is around five orders of magnitude faster when com-
pared to EM making it an appealing candidate for real-time embedded applications.
I also presented a joint maximum-a-posteriori framework for signal estimation with
sample location errors and compared its performance with ILE.

For future work, I would like to study the connection between JMAP estimator
and ILE. I would also like to study how the algorithm presented here can be in-
corporated in the frameworks for adaptive path planning of mobile sensor nodes for
the application of spatio-temporal field mapping. Furthermore, experimental studies
can be conducted with a set of mobile platforms equipped with a sensor (for ex-
ample light sensor) to understand the applicability and suitability of the algorithm. Computationally efficient estimators like ILE have the potential to become an integral component of mobile WSNs, allowing real time improvement of field estimates, improving path planning and reducing sampling at less important locations.
Chapter 6

Conclusion

In this thesis, I have demonstrated the development and use of a mobile sensor platform for addressing management of street lighting infrastructure, studied non-uniform mobile spatial sampling, obtained conditions on δ-dense sampling for stable field estimation and developed a computationally efficient estimator for signal estimation from samples with location errors. The studies and the results presented herein directly impact urban infrastructure management and environment monitoring. In this last chapter, I summarize the contributions of my thesis and propose directions for future research.

6.1 Summary of Contributions

- Mobile sensors present an opportunity to collect relevant data on urban infrastructure in a scalable and cost-effective way. As a practical application of infrastructure management, I have developed a car-mounted autonomous street light scanning platform (CASP). I have tested the prototype in several cities, and it collected data reliably.

- Appropriate analysis of the data can help the decision makers in planning maintenance and replacement needs. I have developed algorithms that make inferences from the noisy sensor data in the presence of false positives. The algo-
gorithms address important problems like improving vehicle trajectory estimation with noisy and sparse GPS data, creating luminosity maps, inventorying street lamps and estimating height of lamp posts.

- In the context of mobile spatial sampling, it is impractical and often impossible to implement uniform grid sampling. In this thesis, I have studied δ-dense sensor arrangement as a flexible, non-uniform sampling scheme with mobile nodes. I have proved sufficient conditions for reconstructing parametric signals modeled in finite basis through δ-dense sensor arrangement and shown that δ-dense sampling performs comparable to uniform sampling in terms of mean squared estimation error.

- Though mobility increases coverage and enables data collection at larger spatial scales, the locations of the mobile nodes may not be accurately known. One needs to address this issue when performing field estimation with data collected by mobile sensor nodes. I have developed an estimator for parametric signal estimation when observations are corrupted by both additive sensor noise and sample location errors. My approach offers several orders of magnitude reduction in computational time while achieving comparable mean squared estimation error to other techniques making it an appealing candidate for real-time embedded mobile sensing applications.

\section{Future Work}

- The sensor system developed in this thesis used off-the-shelf sensors and a microcontroller. Active research in microelectromechanical systems (MEMS) is constantly improving the quality of sensors and supporting electronics. For example, Raspberry Pi is a mini-computer that has attracted significant attention recently [8]. In future, I would like to study its applicability in our system and investigate if it improves sensing and data management.

- Developing algorithms that extract maximum information from noisy sensor
measurements collected by the deployment of mobile sensors in urban conditions is important. The algorithms and analysis have a significant impact on the acceptance of mobile sensor systems as a viable technology. Regarding estimation of vehicle locations, we would like to develop more realistic models of vehicle motion and supplement filtering results with map matching. We would like to explore deep learning for lamp identification. To achieve our vision of creating a “Street View” of street lighting, significant work needs to be done, both on experimental and modeling side, in developing a three-dimensional model of street lighting. We would also like to investigate if the power consumption of street lamps can be estimated through vehicle-mounted mobile sensing.

- We would like to extend mobile sensing for collecting data on other urban infrastructures. For example, multispectral and/or thermal imaging can be used to monitor overhead transmission lines and detect gas leaks. Imaging of public amenities and the subsequent analysis of the images can help the local government to improve the quality of service delivered to the citizens.

- Regarding field estimation through mobile sensors, we would like to study adaptive δ-dense sensor arrangements. The sufficient conditions presented in the thesis are a worst case analysis and we would like to obtain tighter bounds. Adaptive non-uniform sampling schemes will have a direct impact on the deployment strategies for mobile sensors for a particular mission, such as, estimating a radioactive field.

- We would like to investigate how the results on signal estimation with sample location errors can be integrated into δ-dense sampling. Creating a comprehensive framework that includes adaptive non-uniform sampling, location errors of samples and adaptive model selection and refinement is an open challenge. Addressing this challenge can have a significant impact on the understanding of field estimation and “hot spot” identification through mobile sensing under practical constraints on mobility.
Bibliography


