UNIVERSITY OF CALIFORNIA
Santa Barbara

Localization and Mapping using GNSS SNR measurements

A Dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Electrical and Computer Engineering

by

Andrew Theodore Irish

Committee in Charge:

Professor Upamanyu Madhow, Chair

Professor João P. Hespanha

Professor Yasamin Mostofi

Professor Heather Zheng

September 2019

The Dissertation of
Andrew Theodore Irish is approved:

Professor João P. Hespanha

Professor Yasamin Mostofi

Professor Heather Zheng

Professor Upamanyu Madhow, Committee Chairperson

September 2019

Localization and Mapping using GNSS SNR measurements

by

Andrew Theodore Irish

To Joanna, Jeff, Barbara, and Jen.

Acknowledgements

I consider myself very fortunate to have had Professor Madhow as my advisor. He always provided the thorough and honest feedback I needed to develop as a researcher and engineer, and was a fantastic colleague both inside and outside the university. He always gave me freedom to carry out some of my better work, while being available to advise and brainstorm whenever needed. Early on, he encouraged me to try to work on real-world problems with real-world data, and I'm very thankful for that advice. I thank Professors João Hespanha, Yasamin Mostofi, and Heather Zheng for agreeing to continue to serve on my committee, especially after the short pause in my pursuit towards this degree. My colleagues Danny, Jason, Francois, and Dinesh were simply a pleasure to work closely with, and I really appreciate their help in those initial years. Aseem, Hong, Sriram, Babak, Maryam, Faruk, and Zhinous were a pleasure to work with as well. I would like to thank Dr Mantegh for getting my research career started during my undergraduate years at the Canadian National Research Council. Finally, I thank my wife Joanna, father Jeff, grandmother Barbara, Madhow, my supervisors at Uber (Jaikumar and Hemabh), and Joanna's mother and father (Jen and Jack) for supporting and encouraging me to try to complete my doctorate after all these years.

Curriculum Vitæ
Andrew Theodore Irish

Education

2019	Doctor of Philosophy, Electrical and Computer Engineering University of California, Santa Barbara
March 2014	Master of Science, Electrical and Computer Engineering University of California, Santa Barbara
May 2011	Bachelor of Engineering, Electrical Engineering McGill University, Montreal

Publications

Journals

- F. Quitin, A. Irish, and U. Madhow. "A scalable architecture for distributed receive beamforming: analysis and experimental demonstration," *IEEE Transactions on Wireless Communication*, March 2016

Conferences

- A. Irish, F. Quitin, U. Madhow, and M. Rodwell. "Sidestepping the Rayleigh Limit for LOS Spatial Multiplexing: A Distributed Architecture for Long-range Wireless Fiber," *Information Theory and Applications Workshop* (ITA), February 2013, San Diego, California
- A. Irish, F. Quitin, U. Madhow, and M. Rodwell. "Achieving Multiple Degrees of Freedom in Long-range mm-wave MIMO Channels Using Randomly Distributed Relays," *Asilomar Conference on Signals, Systems, and Computers*, November 2013, Pacific Grove, California
- J. Isaacs, A. Irish, F. Quitin, U. Madhow, and J. Hespanha. "Bayesian Localization and Mapping using GNSS SNR Measurements," *IEEE/ION Position, Location, and Navigation Symposium* (PLANS), May 2014, Monterey, California
- A. Irish, J. Isaacs, F. Quitin, J. Hespanha, and U. Madhow. "Probabilistic 3D Mapping based on GNSS SNR Measurements," *IEEE International Conference on Acoustics and Signal Processing* (ICAASP), May 2014, Florence, Italy
- A. Irish, J. Isaacs, F. Quitin, J. Hespanha, and U. Madhow. "Belief Propagation Based Localization and Mapping Using Sparsely Sampled GNSS SNR Measurements," *International Conference on Robotics and Automation* (ICRA), June 2014, Hong Kong, China
- A. Irish, D. Iland, J. Isaacs, J. Hespanha, E Belding, and U. Madhow. "Using Crowdsourced Satellite SNR Measurements for 3D mapping and real-time GNSS positioning improvement," *Proceedings of the Workshop of the Students, By the Students, For the Students* (MOBICOM S3), September 2014, Maui, Hawaii

- A. Irish, D. Iland, J. Isaacs, J. Hespanha, E Belding, and U. Madhow. "ShadowMaps, the urban phone tracking system," *Proceedings of the International Conference on Mobile Computing and Networking* (MOBICOM), September 2014, Maui, Hawaii

Abstract

Localization and Mapping using GNSS SNR measurements

Andrew Theodore Irish

Recent advances in technology, in particular the miniaturization and commoditization of electronic wireless receivers, have enabled new applications and techniques for wireless communication, mapping, and localization.

The example we provide is focused on GNSS (Global Navigation Satellite System) receivers, which widely deployed in mobile consumer electronics for positioning, can be also be leveraged as environment sensors. The problem of estimating a 3D map, or a world model, via crowd-sourced wireless signal strength measurements from global positioning satellites, is explored in detail. The inverse problem – localizing the receiver against a known or partially known 3D map – is then discussed, as is the fully integrated simultaneous localization and mapping (SLAM) problem where both the trajectory of the GNSS receiver and 3D map are jointly estimated. For all of of these cases, novel probabilistic modeling approaches of the map and receiver location, as well as the channel model, are introduced. The proposed solutions are validated using real-world data from consumer-grade mobile devices (smart phones) on both small and larger (multi-city block scale) experiments. For positioning, the backend positioning server and associated smartphone app were prototyped in software, with improved localization using a particle filter being demonstrated in real-time.

Contents

Acknowledgements **v**

Curriculum Vitæ **vi**

Abstract **viii**

List of Figures **xi**

List of Tables **xiii**

1 Introduction **1**
1.1 Mapping . . . 2
1.1.1 Simultaneous localization and mapping (SLAM) . . . 2
1.2 Localization . . . 3
1.3 Contributions . . . 5
1.4 Outline . . . 6

2 Mapping using GNSS SNR measurements **7**
2.1 Related work . . . 9
2.2 Bayesian mapping model . . . 12
2.2.1 SNR measurement model . . . 13
2.2.2 Inferring the map . . . 18
2.3 Bayesian SLAM model . . . 27
2.3.1 Measurement error models and particle sampling . . . 28
2.3.2 Inferring the map and poses . . . 29
2.4 Learning the SNR model . . . 36
2.5 Mapping and SLAM Experiments . . . 37
2.5.1 UCSB mapping experiment . . . 38
2.5.2 UCSB SLAM experiment . . . 44
2.5.3 Downtown Santa Barbara SLAM experiment . . . 49
2.5.4 UCSB SNR model learning experiment . . . 50

3 Localization using GNSS SNR measurements **55**
3.1 Related work . . . 59
3.2 Bayesian localization model . . . 61
3.2.1 System model . . . 61
3.2.2 Particle filters . . . 70
3.2.3 Particle smoothing . . . 77
3.3 Practical implementation . . . 78
3.3.1 Robust particle sampling . . . 78
3.3.2 Efficient 3D map representation and ray tracing . . . 81
3.3.3 Fast particle smoothing . . . 86
3.3.4 SNR model . . . 87
3.3.5 Architecture . . . 88
3.4 Localization Experiments . . . 88
3.4.1 UCSB and Santa Barbara localization experiments . . . 90
3.4.2 San Francisco localization experiment . . . 92

4 Conclusion **99**
4.1 Mapping using GNSS SNR measurements . . . 99
4.1.1 Open Issues . . . 100
4.2 Localization using GNSS SNR measurements . . . 104
4.2.1 Open issues . . . 105

Appendices **108**

A **109**
A.1 Proof of Theorem 2.1 . . . 109
A.2 Proof of Theorem 2.2 . . . 111
A.3 SNR model fitting results . . . 113

B **116**
B.1 Results for the SF localization experiment . . . 116

Bibliography **119**

List of Figures

1.1 Illustration of occupancy grid and SNR measurement scenario 3
1.2 Positioning improvement using Probabilistic SM at UCSB 4

2.1 Rician/Log-normal SNR model 14
2.2 Nakagami-m/Generalized-K SNR model 16
2.3 Illustration of SNR measurement outage probability 17
2.4 Simplified measurement scenario and factor graph for mapping 18
2.5 Local message passing in Loopy BP 21
2.6 Simplified measurement scenario and factor graph for SLAM 30
2.7 Illustration of one iteration of synchronous MP 32
2.8 Horizontal map slice for UCSB mapping experiment 41
2.9 3D mapping at Kohn Hall using mapping algorithm 43
2.10 Horizontal map slices around Kohn Hall using mapping algorithm 43
2.11 Effect of increasing measurement data on mapping 44
2.12 Aerial view of UCSB campus and example GNSS traces 45
2.13 Horizontal map slice for UCSB SLAM experiment 47
2.14 Horizontal map slices around Kohn Hall using SLAM algorithm 48
2.15 Satellite shadows around Kohn Hall based on SLAM algorithm 48
2.16 Positioning improvement around in UCSB SLAM experiment 49
2.17 Aerial view and GNSS traces for Santa Barbara SLAM experiment . . . 50
2.18 Layers of the SLAM-generated map of downtown Santa Barbara 51
2.19 Effect of satellite elevation on SNR 52
2.20 Example SNR model fit 52
2.21 Cubic polynomial regressors for SNR model fits 53
2.22 UCSB SLAM-generated map for optimized SNR model 54

3.1 Illustration of SNR dependence on LOS/NLOS 58
3.2 Bayesian network for the localization system model 62
3.3 GNSS outlier error model. 67
3.4 Tail behavior of GNSS outlier models 68

3.5 Simple Sigmoid SNR model 70
3.6 Illustration of Probabilistic SM likelihood surface region 76
3.7 Illustration of an octree 82
3.8 Cross section of octree occupancy map for UCSB 83
3.9 FCC lattice for likelihood surface KDE 84
3.10 Geodesic dome for "sky grid" for the ray tracing cache 85
3.11 Proposed architecture of the localization system 89
3.12 Positioning improvement at UCSB 90
3.13 Positioning improvement in Santa Barbara 91
3.14 Positioning improvement in San Francisco 93
3.15 San Francisco distance error metrics 96
3.16 San Francisco velocity error metrics 97

A.1 Simplified illustration of the SNR measurement process 112

B.1 San Francisco forward position/velocity error metrics 117
B.2 San Francisco UI jumpiness metrics 118

List of Tables

3.1 Design trade-offs for the ray tracing cache 86

List of Algorithms

1 Building the factor graph for mapping 20

2 BSP-BP message passing for mapping 24

3 Learning the SNR model using EM 38

4 Bootstrap PF for Probabilistic Shadow Matching 71

5 Advanced PF for Probabilistic Shadow Matching 74

6 Primal-dual particle sampling for the Advanced PF 80

7 Sensor reset particle sampling for the Advanced PF 80

Chapter 1

Introduction

Reliance on GNSS, such as the Global Positioning System (GPS), has become ubiquitous with the widespread use of consumer electronics including smartphones, tablets, wearables, and others forming the internet of things (IOT). While principally serving as navigation aids, GNSS receivers can also be viewed as *passive environment sensors,* as follows: 1) As a GNSS receiver traverses an area, obstacles such as buildings, trees, and terrain frequently block line-of-sight (LOS) paths to various satellites, resulting in "shadowed", non line-of-sight (NLOS) channels that are characterized by statistically lower received signal strength; 2) The same receivers can record the data describing such shadowing events, including per-satellite azimuth, elevation and signal-to-noise ratio (SNR), along with the real-time estimated receiver coordinates. In this dissertation, such blocked-versus-unblocked type sensing is used for 3D map building and improved positioning purposes, both separately and in a combined fashion (Simultaneous Localization and Mapping).

1.1 Mapping

We propose and experimentally demonstrate a Bayesian algorithm for processing the preceding GNSS measurements into a probabilistic 3D map of the surrounding environment. Specifically, we show that by discretizing the environment into a probabilistic occupancy grid (OG) map and using a physically inspired sensor model, the posterior probability distribution of the map represents a sparse factor graph, on which an approximate and efficient mapping solution can be computed using the Belief Propagation (BP) algorithm. Furthermore, this factor graph can be built up over time and space (trajectories of GNSS receivers), and as a practical matter, can be fused with prior map information (such as building footprints from everyday sources, such as OpenStreetMap) to reduce convergence time and improve accuracy.

1.1.1 Simultaneous localization and mapping (SLAM)

In areas of high uncertainty in the 3D map (environment) and the location or trajectory of the GNSS receiver, both the map and trajectory can be jointly estimated using a process called Simultaneous Localization and Mapping (SLAM). As in the mapping case, by using a probabilistic OG for the 3D map, physically inspired sensor models for GNSS location fixes and satellite SNR measurements, and introducing particle sets to represent receiver trajectories through space, the posterior probability distribution of the 3D map and receiver trajectories form a sparse factor graph. In this case, SLAM solutions can also be estimated using Belief Propagation (BP).

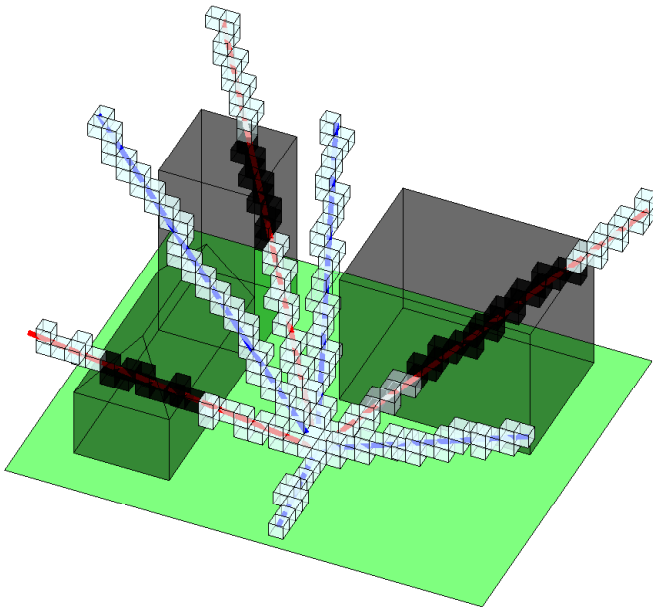


Figure 1.1: Example occupancy grid illustration together with a SNR measurement scenario. Light/dark grid cells approximate empty/occupied space. Blue/red lines represent LOS/NLOS signal paths to satellites, with NLOS signals characterized by statistically lower SNRs.

1.2 Localization

While many mobile applications require accurate geolocalization outdoors, it is an unfortunate fact that in dense urban environments GNSS positioning accuracy degrades significantly, with errors on the order of tens to hundreds of meters. The main culprit for this performance bottleneck is that in large cities the line-of-sight (LOS) to various satellites becomes occluded by buildings, leading to non-light-of-sight (NLOS) and multipath signal reception. As a result, in the best case, the only satellites useful for accurate trilateration come from a narrow area in the sky, yielding poor satellite geometries and positioning accuracy, particularly in the cross street direction. In the typical case, NLOS and *reflected* satellite measurements are also used, severely biasing the GNSS position,

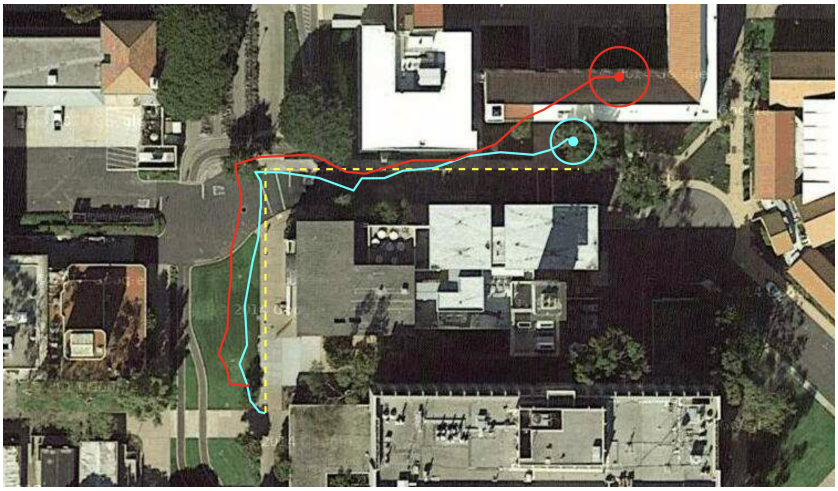


Figure 1.2: Illustration of positioning improvement using Probabilistic Shadow Matching on the UC Santa Barbara campus. True path in yellow, standalone GNSS output in red, and corrected (post-SM) path in light blue.

velocity, time (PVT) solution – and exasperating the cross-street error problem. The underlying geometry and reflection problems are not solved even as additional constellations of Global Navigation Satellite Systems – such as the Russian GLONASS, European Galileo, and Chinese Beidou – become available and supported by mobile devices.

Fortunately, by using available 3D environment information, which may even be only partially known, the location accuracy of GNSS receivers can be improved. This is possible via a technique called *Probabilistic Shadow Matching* (SM), that incorporates probabilistic representations of the map and wireless channel LOS/NLOS status for the satellites observed at a given epoch using a Bayesian Particle Filter (PF). In typical urban canyons, the main effect of Probabilistic SM is to improve cross-street positioning accuracy, naturally complementing GNSS.

1.3 Contributions

We contribute a cohesive architecture for 3D mapping and improved localization using GNSS measurements, which we base on physically sound probabilistic modeling and Bayesian inference methods such as Belief Propagation and Particle Filtering/Smoothing. These techniques let us not only naturally and efficiently represent sensor, map, and location uncertainty, but also enable calibration using additional data sources and constraints, such as building footprint data and motion models (and permitting fusion with inertial sensors and road networks in the future, for example).

Another important contribution are our experimental results for the proposed mapping, localization, and SLAM algorithms at different scales and environments. In particular:

- Our BP-based mapping and SLAM algorithms were tested in experiments which scaled from a corner of the UC Santa Barbara campus to cover most of downtown Santa Barbara. Our localization results for the PF-based system are shown for the pedestrian use case, both in low and medium rise scenarios (UCSB campus and downtown Santa Barbara) as well as high rise environments (San Francisco).

- We propose and experimentally demonstrate a practical Expectation Maximization (EM) algorithm for estimating SNR models, an important subproblem for both Probabilistic SM and 3D mapping/SLAM.

- The feasibility and practicality of the positioning system were demonstrated by implementing the Particle Filter in cloud-based software, and running it real-time against GNSS satellite measurements streamed from an Android smartphone.

For the localization system, a significant contribution is our fully Rao-Blackwellized PF that optimally samples from the likelihood surface and motion model, allowing for high accuracy tracking while preventing divergence in deep urban cores. We also present an accelerated offline estimator (Particle Smoother). In addition, we discuss features that were useful in implementing the real-time system, including cached ray tracing models that take advantage of naturally occurring temporal coherence of GNSS shadowing, as well as leveraging memory- and compute-efficient hierarchical (octree-based) representations of the 3D map.

1.4 Outline

We review the algorithms, architecture, implementation, and experimental results for mapping and positioning with GNSS SNR measurements in Chapters 2 and 3. We present our conclusions in Chapter 4.

Chapter 2

Mapping using GNSS SNR measurements

For decades, global navigation satellite systems (GNSS) have been widely used for geolocation purposes. Many satellite systems have been successfully deployed, including America's GPS, Russia's GLONASS, Europe's Galileo, China's Beidou, and Japan's GZSS. The end result is that an ever growing list of satellite constellations are broadcasting wireless signals so that earth-based GNSS receivers can determine their geolocation and related quantities. Widely used GNSS receivers, for instance those embedded in smartphones or wearable devices, land vehicles, aircraft, etc., can passively report information derived from different received satellite signals.

Satellite SNR: As discussed thoroughly in Chapter 3, for most end users of GNSS, this receiver-derived information principally pertains to the so-called Position-Velocity-Time (PVT) solution, computed using satellite range and range-rate observations. An

important set of satellite measurements often not utilized in full capacity, and the subject of much of this chapter, are the *signal strengths* of each satellite in view. In the literature, these are often referred to as Carrier-to-Noise (CN0) or Signal-to-Noise ratio (SNR), and are necessarily computed by the GNSS receiver in order to select and weigh pseudoranges and pseudorange-rates across satellites when computing PVT solutions. The reason for this is straightforward:

- Measurements from high SNR satellites are less likely to be NLOS and hence less likely to be corrupted by multipath.
- LOS satellite signals with higher SNR are by definition less corrupted by noise, leading to higher quality observations.

Thus, as the LOS to some various satellites is occluded by obstacles, such as buildings, trees, etc., NLOS signals are measured, which are characterized by a lower SNR than LOS signals. This very simple observation can be used to improve positioning using a technique called Shadow Matching as discussed extensively in Chapter 3. Conversely, signal strengths can provide geographical information about obstacles: If the receiver and satellite coordinates are known, whenever a GNSS signal is blocked, it is possible to determine the direction of an obstacle.

In this chapter, we demonstrate algorithms, architectures, and experimental results that leverage how GNSS devices can be viewed as passive environment sensors, relating the state of the world (the 3D map) and true position of the GNSS receiver via sets of

coarse (noisy) satellite SNR measurements and a coarse (noisy) PVT solutions. Namely, the two distinct applications of such sensing that we explore include:

1. Probabilistic 3D map building against known GNSS receiver locations (Section 2.2).
2. Simultaneous localization and mapping (SLAM), for 3D map building using partially observed (uncertain) receiver locations (Section 2.3).
3. Learning satellite SNR models for mapping (Section 2.4).

2.1 Related work

Since GNSS accuracy degrades in cluttered urban environments, the dual problem to mapping – refining position estimates using *known* environment maps – is more well researched to date. A technique which has been used to achieve significant localization improvement is *Shadow Matching* (SM), and is discussed in Chapter 3. Our work here provides a mechanism to generate maps *for* SM (or other purposes), which relies on up-to-date 3D maps that are not always available and can be expensive to obtain via other means. In [43] a flavor of mapping for SM is implicitly performed by first learning and then matching against SNR measurement models, which vary as a function of the receiver coordinates, an approach similar to ours in that LOS/NLOS channel conditions are modeled probabilistically. The problem of using GNSS signal strength to construct environment maps has received less attention than SM. Currently published approaches [34, 62] learn shadows of buildings from GNSS information as in SM, and then employ

non-probabilistic heuristics based on ray tracing to reconstruct environment maps. To the best of our knowledge, the present paper is the first to apply a systematic Bayesian approach to such shadow-based mapping. We also believe that a probabilistic approach is more appropriate in general, not only due to the large measurement uncertainty involved, but also because it turns out that empty space is more easily identified than occupied space.

Of course, Bayesian approaches are now standard in localization and mapping problems [53], with both the environment and sensor readings being modeled probabilistically. However, existing Bayesian algorithms are all based on implicit or explicit measurements of distances to obstacles, using a variety of sensing methods such as lidar/radar [55, 12], mono/stereo camera [10, 16], and WiFi [17]. Our sensing model is fundamentally different from these: an SNR measurement for a given satellite only gives us probabilistic information about whether or not the ray to the satellite is blocked, which makes the "data association problem" [53] of associating measurements with map features particularly challenging. To get around this, we represent the environment as an Occupancy Grid [14], a volumetric model that – while less common than feature-based representations – facilitates data correspondence by quantizing measurement rays, as illustrated in Figure 1.1. Unlike traditional Occupancy Grid mapping, however, we model the grid elements as strongly dependent using a factor graph [36], which captures the complex interactions between measurement rays that span (and thus intertwine) widely separated portions of the map.

In addition to the radically different measurement geometry, there is another key difference between our model and range-based Bayesian algorithms. Range sensors typically operate within a *local* reference frame, and thus provide readings that are sensitive to pose errors. In the GNSS mapping problem, *global* satellite/receiver coordinates (and hence the measurement beam directions) are known. As a result, for the case of small positioning errors the effects of positioning error are partially "averaged out" by using large amounts of measurement data and by restricting ourselves to coarse grids (map resolutions). This allows for significant algorithmic simplifications compared to the full simultaneous localization and mapping (SLAM) problem (see [54] for an introduction).

For purposes of handling larger positioning errors in the mapping process, we present a SLAM solution and discuss various extensions of it. Here, we formulate the posterior distribution of the latent variables (map and poses) as a factor graph, much as in GraphSLAM [57]. However, in our scenario, measurement rays observe many (tens to hundreds of) grid cells, thus intertwining widely separated portions of the map and forming large cliques in the graph. Techniques such as GraphSLAM that rely on linearity assumptions and variable elimination are thus impractical if not inapplicable. Instead, we employ a low complexity version of Loopy Belief Propagation [36] to efficiently estimate the marginal distributions of the variables.

2.2 Bayesian mapping model

We represent the environment as an *Occupancy Grid*, a 3D grid of "map cells" $m = \{m_i\}_{i=1}^{L}$ with $m_i \in \{0, 1\}$ denoting empty and occupied states respectively. The mapping problem is then formulated as estimating the occupancy probability of each cell m_i using information logged by GNSS receivers. The first piece of information is the set of *noisy* satellite SNR measurements, which consist of T vectors, $z = \{z_t\}_{t=1}^{T}$, where $z_t = [z_{t,1}, \ldots, z_{t,N_t}]$ contains individual SNR readings and N_t is the number of satellites in view at measurement index t. For every reading $z_{t,n}$, the receivers also provide satellite elevations and azimuths $[\theta_{t,n}, \phi_{t,n}]$, which we consider noiseless. The corresponding receiver position "fixes" are denoted $x = \{x_t\}_{t=1}^{T}$. As previously mentioned, in this section we treat these as noiseless as well.

Under the "static world" assumption in which the environment is modeled as constant over time, the SNR measurements can be modeled as conditionally independent given the map and poses. Applying Bayes theorem to the posterior distribution of the map, we then have

$$p(m|x, z) = \frac{p(m)p(z|m, x)}{p(z)} \propto p(m) \prod_{t,n} p(z_{t,n}|x_t, m) \tag{2.1}$$

where $p(m)$ is the prior distribution of the map. Although in this paper we assume the environment is completely unknown, the algorithm we develop in the upcoming sections also permits for map priors of the form $p(m) = \prod_i \psi_i(m_i)$.

2.2.1 SNR measurement model

The measured SNR of a given GNSS signal depends on a many factors in reality, including satellite elevation, environmental parameters, and receiver characteristics. While statistical models exist for the wireless channels of interest [39, 1], to avoid over-modeling and greatly simplify the LBP-based inference step, we define the measurement model as follows:

$$p(z_{t,n}|x_t, m) = \begin{cases} f_{LOS}(z_{t,n}), & m_i = 0 \ \forall \ i \in \mathcal{M}(t,n) \\ f_{NLOS}(z_{t,n}), & \text{otherwise} \end{cases} \tag{2.2}$$

where $\mathcal{M}(t,n)$ indexes the map cells intersected by the ray starting at receiver position x_t in the direction of the nth transmitting satellite $[\theta_{t,n}, \phi_{t,n}]$. In other words, an SNR reading is LOS-distributed if *all* cells "observed" by its associated ray are empty; otherwise, it is NLOS-distributed.

Rician/ Log-Normal model

In the above expression, one option for f_{LOS} and f_{NLOS} that is physically motivated are the Rice [2] and Log-Normal fading distributions, as illustrated in Figure 2.1. In other words, an SNR reading is LOS (Rice) distributed if *all* cells "observed" by its associated ray are empty; otherwise, it is NLOS (log-normal) distributed.

Physically, this approach assumes 1) that a LOS signal is well modeled as a superposition of a dominant LOS component and a circular Gaussian (multipath fading) component, and 2) a NLOS signal is subjected to random, multiplicative shadow fading. The setting of parameters, such as Ω, K for the Rice density, is discussed in Section 2.5.1.

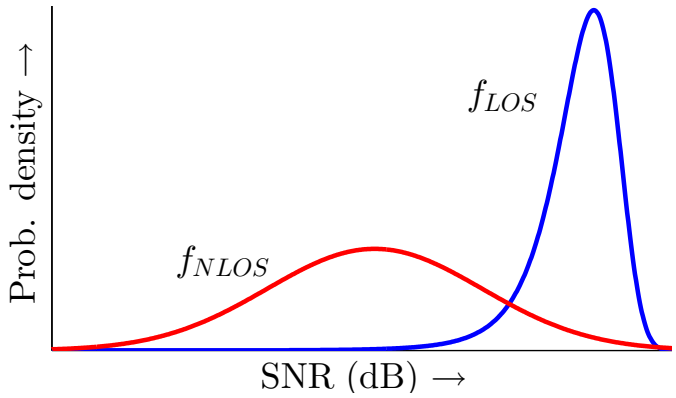


Figure 2.1: The LOS/NLOS satellite channels can be modeled according to Rician/log-normal distributions.

As illustrated in Figure 2.1 and discussed more in Section 2.5.1, one way to model received power under LOS and NLOS hypotheses using Rician (parameterized by Ω, K [2]) and log-normal fading distributions, respectively. Physically, this approach assumes 1) that a LOS signal is well modeled as a superposition of a dominant LOS component and a circular Gaussian (multipath fading) component, and 2) a NLOS signal is subjected to random, multiplicative shadow fading. After a change of variables to define the Rice density on the decibel scale, the following PDF (probability density function) is obtained

$$f_{LOS}(r) = \frac{\ln 10}{20}\frac{10^{r/10}}{s^2}\exp\left(-\frac{10^{r/10}+v^2}{2s^2}\right) I_0\left(\frac{10^{r/20}v}{s^2}\right) , \tag{2.3}$$

where $v^2 = \frac{K}{1+K}\Omega$, $s^2 = \frac{\Omega}{2(1+K)}$, $I_0(\cdot)$ is the 0th order modified Bessel function of the first kind, and Ω and K are the total estimated channel power and the Rician "K factor" (ratio of LOS to diffuse power), respectively. As for the NLOS hypothesis, in decibels the Log-Normal fading model for NLOS channels is described by a normal distribution:

$$f_{NLOS}(r) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{(r-\mu)^2}{2\sigma^2}\right) , \tag{2.4}$$

with mean fading value μ and variance σ^2.

Nakagami-m/ Generalized-K model

Another option for modeling the LOS component of SNR model is the Nakagami -m distribution. A benefit of this choice is that it is more flexible than the Rician model [49], i.e. is better at handling a variety of fading conditions. Its distribution, after transforming from linear domain to decibels ($r = 10^{r_{\text{dB}}/20}$), is

$$f_{LOS}(r) = \frac{\ln 10}{10} \frac{m^m}{\Gamma(m)\Omega^m} \exp\left(-m\Omega r\right) \tag{2.5}$$

where m is the well known Nakagami-m parameter (representing the severity of the fading channel) and Ω is the total LOS signal power.

The Generalized-K (K_G) distribution is a flexible model that is well suited for modeling shadowed wireless channels [37], and can be defined as the convolution of two Nakagami distributions [6]. It is defined by three parameters (m, c, μ), where (m, c) represent the two Nakagami fading parameters and μ is the overall mean received power. In decibels, the K_G probability density function is represented as

$$f_{NLOS}(r) = \frac{\ln 10}{5} \frac{1}{\Gamma(m)\Gamma(c)} u(r)^{m+c} \; K_{m-c}\left(2 \; u(r)\right) \tag{2.6}$$

where $u(r) = \sqrt{\frac{mcr}{\mu}}$ and $K_o(\cdot)$ is the modified Bessel function of the second kind of order o.

As explored in Section 2.4, and shown in Figure 2.2, an appealing attribute about the Nakagami/ K_G model is its flexibility, facilitating the task of offline parameter estimation.

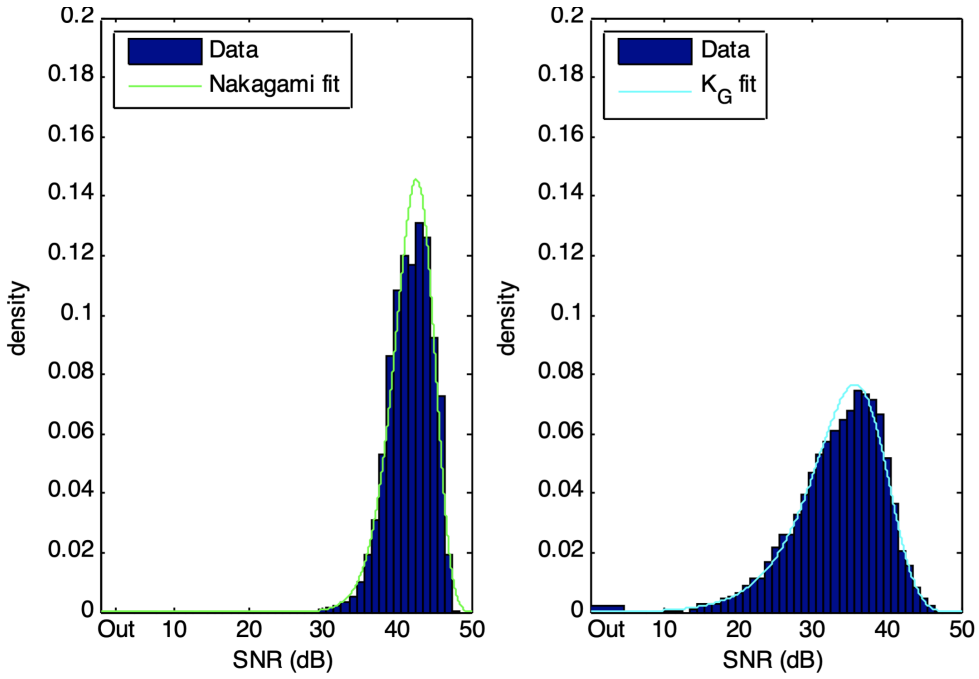


Figure 2.2: The LOS/NLOS satellite channels can also be modeled according to Nakagami-m/Generalized-K distributions, which are flexible models and easy to empirically fit against SNR data.

Outage probability

In reality, satellite signals may be so weak that the received power falls below the sensitivity of the GNSS receiver $z_{\min}$, resulting in outage. For Android smartphones (used to test our mapping algorithms in Section 2.5), outage is frequently represented by 0-SNR readings for satellites which otherwise should be detectable (i.e., ephemeris is decoded and the satellite was recently used).

Intuitively, outage is a useful measurement in and of itself – it indicates the channel is likely to be completely NLOS. For mapping purposes, we can define the outage probabilities of the LOS and NLOS models as their cumulative density functions $F_{LOS}(\cdot)$ and $F_{NLOS}(\cdot)$. For example, for the NLOS model, we have

$$\Pr(\text{Outage}|\text{NLOS}) = \int_{-\infty}^{z_{\min}} f_{NLOS}(r)\, dr \triangleq F_{NLOS}(z_{\min}). \tag{2.7}$$

Graphically, this is shown in Figure 2.3. In the case of NLOS outage, we can simply substitute in this expression wherever the non-outage NLOS SNR model is employed (and vice versa for the LOS model).

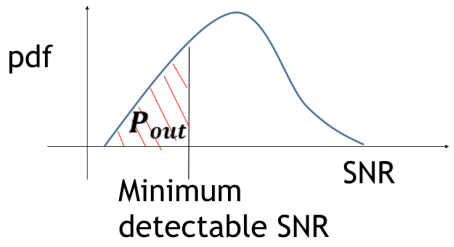
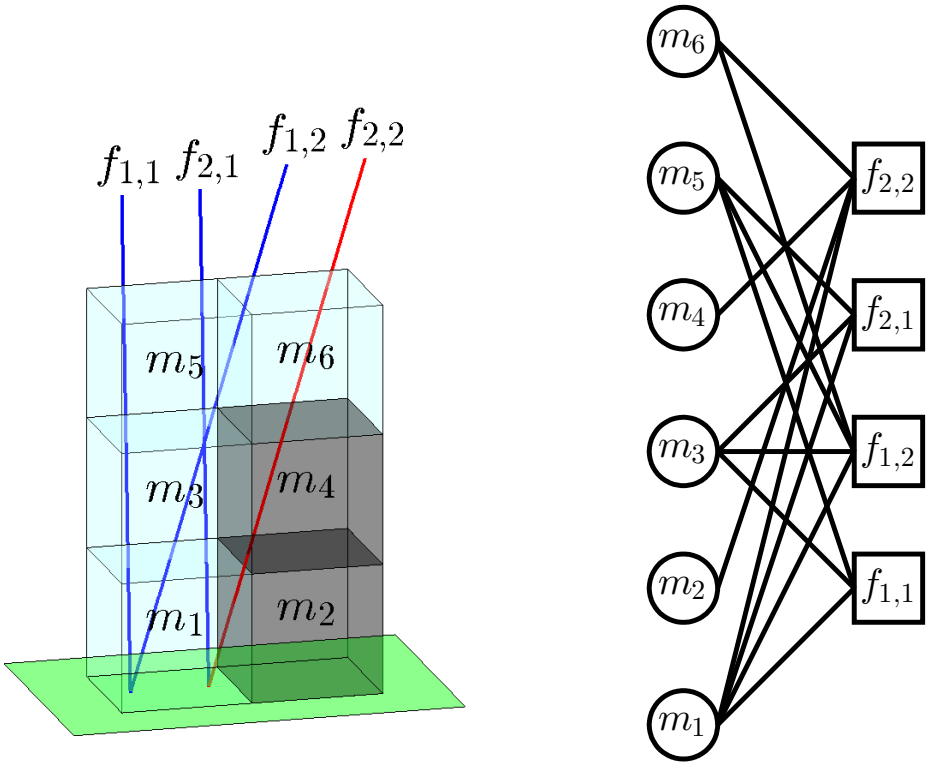


Figure 2.3: Illustration of satellite SNR measurement outage probability (P_out) and how it relates to the f_{LOS}/f_{NLOS} probability density functions (pdfs) in terms of "area under the curve".

2.2.2 Inferring the map



(a) Measurement scenario

(b) Factor graph

Figure 2.4: Simplified measurement scenario and factor graph, with circles/squares representing variable/factor nodes. Note that map prior factors $\{\psi_i(m_i)\}$ are singly connected to their corresponding $\{m_i\}$ and are not shown for simplicity.

The key realization of this paper is that the map posterior (2.1) describes a *factor graph* [36], a bipartite $G = (\{X, F\}, E)$, with variable nodes X, factor nodes F, and edges E. In our application, the variable nodes refer to $X = \{m_i\}$, which are associated with the 3D map state (map cells). The factor nodes, denoted $F = \{f_{t,n}\} \bigcup \{\psi_i(m_i)\}$,

correspond to the SNR measurement likelihoods $\{p(z_{t,n}|x_t, m)\}$ and the *apriori* estimate for the 3D map (see Figure 2.4 for an illustration).

Importantly, the graph is *sparse* in its edges $E = \{\mathcal{E}_{i,(t,n)}\}$ due to the sensor model (Equation (2.2)): each factor only depends on (is adjacent to) the relatively small set of map cells that are intersected by its associated ray. In addition, the factor graph contains cycles. The process for building the factor graph, including the ray tracing, is described in Algorithm 1.

Map projection

Note that GNSS fixes are usually represented in geographic coordinates (latitude, longitude, altitude), such as WGS84. When building the factor graph, a *Map Projection* must be considered, to put the map and fixes in the same (assumed Cartesian) coordinate reference frame. For these purposes, a local tangent plane approximation to WGS84 is suitable for small maps (a few hundreds of meters wide). Otherwise, a conformal (i.e., not warping) projection such as Universal Transverse Mercator (UTM) should be used. The reader is referred to [21] for more details.

Inference via reduced complexity LBP

A well-known technique that is used for efficient inference on such "loopy" factor graphs (and used in the context of mapping in [46]) is the *Loopy Belief Propagation (LBP)* [44], or *Sum-Product Algorithm* [36], an iterative message passing algorithm, which after

Algorithm 1 Building the factor graph for mapping

input (GNSS fix, SNR measurement vector pairs $\{(x_t, z_t)\}$, optional map prior Ψ, map cell resolution d, SNR model Z)
initialize $X = F = E = \emptyset$
for all GNSS fix, SNR measurement vector pairs (x_t, z_t) **do**
  for $n = 1$ to N_t **do**
    Create factor $f_{t,n}$ under SNR model Z and add to F
    Ray trace: $I_{i,t,n}$ = map cells intersected by ray from position x_t towards nth satellite at time t, under map resolution d
    for all map cells $\in I_{i,t,n}$ **do**
      Get variable m_i from X if present, or create and add to X
      Create edge $e_{i,t,n}$ between m_i and $f_{t,n}$, and add to E
    end for
  end for
  for all map cells in X, optionally **do**
    Create factor ψ_i under prior Ψ, connected to m_i, and add to E
  end for
end for
return Factor graph $G = (\{X, F\}, E)$

convergence (assumed but not guaranteed) yields estimates of the variables' marginal posteriors.

As in standard LBP and shown in Figure 2.7, we have two types of messages passed locally in the factor graph. The message from variable m_i to an adjacent measurement

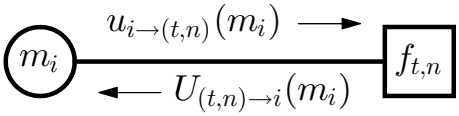


Figure 2.5: Local message passing in Loopy BP and the notation used in this paper.

factor $f_{t,n}$ is, for m_i defined on its domain $\{0, 1\}$,

$$u_{i\to(t,n)}(m_i) \propto \psi_i(m_i) \prod_{(\tau,\eta)\in\mathcal{F}(i)\setminus(t,n)} U_{(\tau,\eta)\to i}(m_i) = \frac{b_i(m_i)}{U_{(t,n)\to i}(m_i)} \tag{2.8}$$

where

$$b_i(m_i) = \psi_i(m_i) \prod_{(t,n)\in\mathcal{F}(i)} U_{(t,n)\to i}(m_i), \tag{2.9}$$

is the current "belief"" of node m_i, $\mathcal{F}(i)$ indexes the adjacent factors, $\psi_i(m_i)$ is the cell prior (uniform in this paper), and the messages are normalized to sum to one. In the other direction, employing the standard Sum-Product formula we have

$$U_{(t,n)\to i}(m_i) = \sum_{m\setminus m_i} p(z_{t,n}|m, x_t) \prod_{j\in\mathcal{M}(t,n)\setminus i} u_{j\to(t,n)}(m_j)\ , \tag{2.10}$$

where $\sum_{m\setminus m_i}$ is the marginalization sum over m_i, i.e., a sum over $2^{|\mathcal{M}(t,n)|-1}$ terms where $|\mathcal{M}(t, n)|$ in the number of cells observed by the SNR measurement. Because in our application measurement rays typically intersect tens of map cells, evaluating (2.10) directly is impractical. However, upon substitution of (2.2), we obtain a much simpler simple formula.

Theorem 2.1. *Computing the messages from factors to variables $U_{(t,n)\to i}$ is of complexity $O(E|\mathcal{M}(t,n)|_{t,n})$, where $E|\mathcal{M}(t,n)|_{t,n}$ is the average factor degree, and the expression for the messages is given by*

$$U_{(t,n)\to i}(m_i) \propto \begin{cases} F\left(z_{t,n}, \dfrac{\gamma_{t,n}}{u_{i\to(t,n)}(0)}\right), & m_i = 0 \\ 1, & m_i = 1 \end{cases} \tag{2.11}$$

where

$$F(a,b) \triangleq 1 + \left(\frac{f_{LOS}(a)}{f_{NLOS}(a)} - 1\right) b \tag{2.12}$$

is the SNR likelihood "belief" function

$$\gamma_{t,n} = \prod_{j\in\mathcal{M}(t,n)} u_{j\to(t,n)}(0) \tag{2.13}$$

is the current LOS belief, i.e. the belief that all neighbors of factor node $f_{t,n}$ are empty. The messages $U_{(t,n)\to i}(m_i)$ are normalized to sum to one.

Proof of Theorem 2.1: We give the proof of the theorem in Appendix A.1.

Remark on Theorem 2.1: Due to the LOS/NLOS measurement model, computing (2.10) has been reduced from exponential to linear complexity in the factor degrees. Moreover, these messages intuitively only depend on two quantities:

1. The LOS/NLOS likelihood ratio of the SNR measurement, $\frac{f_{LOS}(z_{t,n})}{f_{NLOS}(z_{t,n})}$.

2. The total current evidence (current belief) that all map cells intersected by the ray corresponding to the factor are empty (unblocked), $\gamma_{t,n}$, less (divided by) the evidence coming from variable m_i that it is empty (as is typical in BP, avoiding double-counting evidence during message propagation).

Running LBP and reading the output: As is standard practice, we run LBP in bulk synchronous parallel (BSP) [19] fashion over all edges, scheduling message passing between all variable nodes and their neighboring factors to be executed in parallel synchronously, and then vice versa. This is performed sequentially and repeatedly until convergence is detected or a maximum number of iterations K_max has been met. To speed up convergence, we also initialize the outgoing messages from the variables (map cells), as well as the beliefs, to their a-priori estimates, if available (or the uniform probability mass function if not available). The exact BSP-BP algorithm is specified in Algorithm 2.

After convergence of LBP, the marginal posterior of each map cell $p(m_i|x,z)$ is approximated by its "belief" from Equation 2.9

$$p(m_i|x,z) \approx b_i(m_i) \tag{2.14}$$

which is normalized to sum to one, i.e. $b(0) + b(1) = 1$, so that it represents a valid probability mass function. In this thesis, we define convergence to be when the max of all belief residuals R_i falls below a predetermined threshold ϵ, where residuals are defined as

$$R_i(m_i) \triangleq R_i(m_i = 1) = |b_i(1) - b_i^{old}(1)|. \tag{2.15}$$

Message damping

Note that, as is often the case in practical LBP implementations, to limit oscillations and help ensure that LBP converges, at each message passing iteration we apply damping

Algorithm 2 BSP-BP message passing for mapping

input factor graph $G = (\{X, F\}, E))$

initialize beliefs to priors $b_i(m_i) = \psi_i(m_i)$ for all variables m_i

initialize messages to priors $u_{i\to(t,n)}(m_i) = \psi_i(m_i)$ for all edges $e \in \{\mathcal{E}_{i,(t,n)}\}$

initialize previous beliefs $b_i^{old}(m_i) = \emptyset$ for all variables m_i

for k = 1 to K_{max} **do**

  update beliefs $\{\gamma_{t,n}(0)\}$ for all factors $f_{t,n}$ (in parallel)

  for all edges $e \in \{\mathcal{E}_{i,(t,n)}\}$, in parallel **do**

    update message $U_{(t,n)\to i}$ from factor $f_{t,n}$ to variable m_i

  end for

  update beliefs $\{b_i(m_i)\}$, residuals $\{R_i(m_i)\}$ for all variables m_i (in parallel)

  if $\max_i R(m_i) < \epsilon$ **then**

    converged, break

  end if

  for all edges $e \in \{\mathcal{E}_{i,(t,n)}\}$, in parallel **do**

    update message $u_{i\to(t,n)}$ from variable m_i to factor $f_{t,n}$

  end for

  assign previous beliefs: $b_i^{old}(m_i) = b_i(m_i)$ for all variables m_i

end for

return 3D map estimate: $\{b(m_i)\}$ for all variables m_i

(as in, e.g., [18] equation (3.7)) with damping factor $\rho \in (0, 1)$. Each outgoing message is then a convex combination of the old outgoing message and the (undamped) outgoing message as computed via Equations (2.8), (2.11):

$$\text{New Msg} = \rho \cdot (\text{Old Msg}) + (1 - \rho) \cdot (\text{Undamped New Msg}) \,. \tag{2.16}$$

Computational complexity

The bulk of the computational (processing and memory) complexity of our mapping algorithm is contained in the synchronous BP message passing (MP) iterations. Here, we provide a high-level analysis of the relevant algorithmic parameters in order to gain insight into its scaling in terms of computational complexity. Accordingly, it is shown that our algorithm scales linearly in the number of measurements and size of the map, if we make the following simplifying assumptions, without loss of generality:

1. The region to be mapped M is cylindrical with diameter D and height H.

2. The L map cells are contiguous, non-overlapping spheres of diameter $d \ll D$.

3. The receiver trajectory consists of T uniform samples across G_M, the ground plane of M.

4. An average of N satellites are in view at each time t, with coordinates uniform over the sky. Since the area element on the unit sphere is $\cos\phi \; d\phi \; d\theta$, this is equivalent to assuming that azimuths $\theta_{t,n}$ are uniform over $[0, 2\pi]$, and elevations $\phi_{t,n} \in [0, \frac{\pi}{2}]$ are independently distributed as $f_\phi(\phi) = \cos\phi$.

5. There is a small, minimum satellite elevation, $\phi_{\min}$, below which corresponding SNR measurements are discarded. This is motivated by the fact that low elevation satellites are subject to wider SNR fluctuations [45] caused by more severe multipath and shadowing conditions.

Theorem 2.2. *Let the above assumptions hold, and let $|\mathcal{N}(i)|$ denote the number of measurement factors $f_{t,n}$ adjacent to cell variable m_i. The total computational complexity for each iteration of message passing is then*

$$O\left(\sum_i |\mathcal{N}(i)|\right) = O\left(\sum_{t,n} |\mathcal{M}(t,n)|\right) \tag{2.17}$$

where

$$\mathbb{E}\left[\sum_i |\mathcal{N}(i)|\right] < \frac{3NT}{2}\left(\frac{H}{d}\right)\ln\left(\csc\phi_{\min}\right) \ . \tag{2.18}$$

Proof of Theorem 2.2: We give the proof of the theorem in Appendix A.2.

Remark on Theorem 2.2: Equations (2.17), and (2.18) state that the total complexity of each iteration of BP in our algorithm is linear in the number of measurement epochs T and the average number SNR measurements per epoch (i.e., the number of satellites tracked simultaneously by the GNSS receiver N). Moreover, computational complexity *intuitively* scales linearly with height of the map H, inversely with the size of the map cells d, and also decreases if we apply sensible satellite elevation masks ($\phi_{\min}$) (for example, complexity is approximately reduced by half if we increase $\phi_{\min}$ from 2° to 10°).

Note, however, for practical mapping applications where updates are applied to the 3D map (and factor graph) over time using newly collected GNSS measurements, the incremental computational complexity of BP-based map updates can be made much lower.

Specifically, the graph could be augmented on-the-fly, and an asynchronous message passing strategy (instead of sequential BSP) could be used to only perform message updates along edges connected to nodes whose beliefs haven't yet converged (such as recently added edges and their neighborhood in the factor graph). Details of such approaches are left for future work.

2.3 Bayesian SLAM model

In the previous sections, we presented a model and algorithm for 3D mapping using GNSS SNR measurements in cases where PVT (positioning) error was negligible. In this section, we explore the case of mapping in situations where GNSS positioning (PVT) uncertainty is non-negligible, such as in urban canyons where multipath and poor satellite geometry (dilution of precision) are commonplace.

Our algorithm relies on ray tracing from sets of hypothetical receiver locations (particles) towards given satellites. Probabilistic "beams" of rays – sets of parallel rays emanating from the same particle set and heading to the same satellite – are then assigned likelihoods of being LOS or NLOS, depending on the measured satellite SNR values. Finally, these beams are stitched together to form a soft probabilistic occupancy map using Loopy Belief Propagation, which concurrently re-weights position particles, yielding revised location estimates as well.

We model both the environment and receiver positions non-parametrically: 1) As in Section 2.2 the map is represented as a 3D grid of cells, $m = \{m_i\}_{i=1}^{L}$, with $m_i \in \{0, 1\}$

denoting empty and occupied space; 2) The space of possible GNSS receiver trajectories $x = \{x_t\}_{t=1}^{T}$ is represented using sets of particles, so that individual positions are $x_t \in \{x_t^k\}_{k=1}^{K}$. The SLAM problem is then formulated as estimating the marginal distributions of each latent variable m_i and x_t. As before, we make use of the satellite SNR measurement vectors at each epoch, $\{z_t\}_{t=1}^{T}$; the difference is that now, instead of being noiseless, each position x_t is uncertain and observed a noisy GNSS "fix", y_t.

Under the assumption of a static world (where the map m does not change over time), and (for simplicity, for now) assuming no a-priori information on the map and poses (such as information on building locations or a motion model governing x), the SNR measurements and GNSS fixes can be modeled as conditionally independent given the map and positions. Thus, the posterior distribution of the latent variables given the measurements factorizes as follows:

$$\begin{aligned} p(m, x|y, z) &\propto p(y, z|m, x) = p(z|m, x)p(y|m, x, z) \\ &= \prod_{t,n} p(z_{t,n}|m, x_t) \cdot \prod_{t} p(y_t|x_t). \end{aligned} \tag{2.19}$$

2.3.1 Measurement error models and particle sampling

While changes are required for the SNR Model (see Section 2.2.1), the new type information used are the receiver position estimates (GNSS fixes), which are noisy and modeled as independent Gaussian random variables

$$y_t = x_t + e_t, \quad e_t \sim N(0, C_t). \tag{2.20}$$

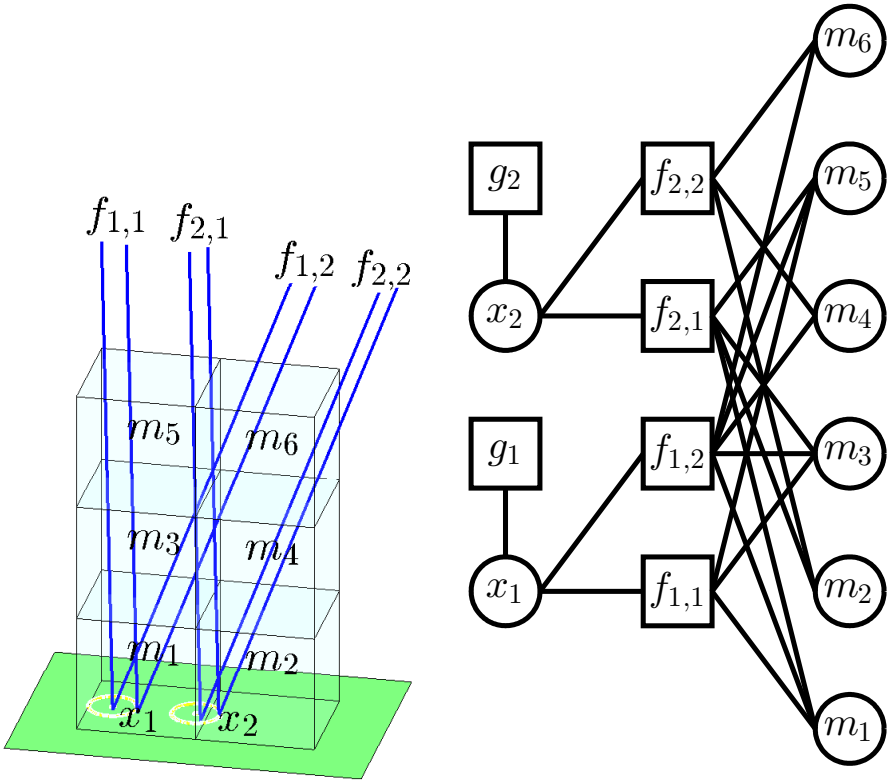
where $N(\mu, \Sigma)$ refers to the multivariate Gaussian distribution with mean vector μ and covariance matrix Σ. As in Chapter 7 from [33], we estimate the error covariance matrix C_t using the formula for HDOP scaled by the uncertainty reported by the receiver, so that the position fixes do not depend directly on the state of the map. Given the position uncertainties, the particles $\{x_t^k\}$ are then sampled according to $N(y_t, C_t)$ and assigned equal weights. Note that, since (2.20) assumes the pose errors are uncorrelated, while in reality consecutive GNSS fix errors are known to be correlated (as they are the output of correlated physical multipath processes and navigation Kalman Filters), it is important that the input data is sparsely sampled in time.

2.3.2 Inferring the map and poses

As in the mapping case (Section 2.2.2), the overarching idea of this section is that the factorization (2.19) represents a factor graph on which the Loopy Belief Propagation algorithm can be used to perform the SLAM computation.

The factor graph

As before, a factor graph $G = (\{X, F\}, E)$, with variable nodes X, factor nodes F, and edges E, describes the factorization of a global function into a product of local functions (factors). It is a bipartite graph where an edge between a factor node and a variable node appears if and only if the variable is an argument of the factor. Referring to Figure 2.6 as an example, in our graph there are two classes of variables and factors,



(a) Measurement scenario (b) Factor graph

Figure 2.6: Simplified measurement scenario with $T = N_1 = N_2 = K = 2$ and its corresponding factor graph, with circles/squares representing variable/factor nodes.

$X = \{x_t\} \bigcup \{m_i\}$ and $F = \{f_{t,n}\} \bigcup \{g_t\}$. Each "SNR measurement" factor node $f_{t,n}$ corresponds to $p(z_{t,n}|m, x_t)$, and each "receiver position" factor g_t is associated with $p(y_t|x_t)$.

While each g_t is singly connected to x_t, the edges between SNR measurements and map cells in the factor graph are only known after ray tracing. The indices of the $\{m_i\}$ adjacent to $f_{t,n}$ are $\mathcal{M}(t,n) = \bigcup_k \mathcal{M}(t,n,k)$. Physically speaking, $\mathcal{M}(t,n)$ describes the set of cells sensed by the "beam" of parallel rays emanating from hypothetical receiver positions $\{x_t^1, \ldots, x_t^K\}$ in the direction of satellite n.

Building the factor graph: The process for building the factor graph for the SLAM case is very similar to that of mapping, outlined in Algorithm 1. The key differences are:

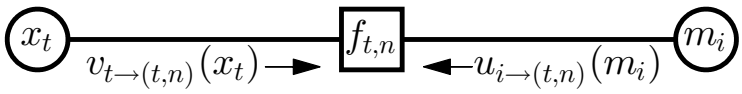
1. Particles x_t^k are drawn from each GNSS position fix y_t. These particles are then viewed attributes (the support) of the corresponding position variables x_t, and are used for ray tracing in an inner for-loop over particles.

2. Variable nodes x_t are created, with edges connecting x_t and corresponding fix factors y_t as well as x_t and each associated SNR measurement factors $f_{t,n}$.

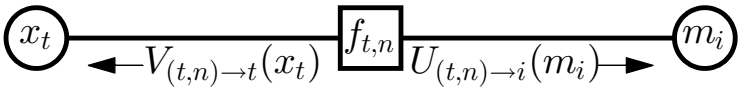
3. There is some extra book-keeping required to support belief propagation. For example, each particle-satellite ray must be associated with a set of observed map cells (i.e., $\mathcal{M}(t,n,k)$ used in Equations 2.24 and 2.29). These additional "subgraphs" are not *explicitly* part of the factor graph, and must be handled separately.

Low complexity LBP message passing

As can be seen in Figure 2.4, the factor graph representing the posterior contains cycles. An efficient algorithm used for approximate inference on such graphs, which has also been used for SLAM [46], is the Loopy Belief Propagation (LBP) (or Sum-Product) algorithm [36]. Essentially, LBP is a message passing (MP) algorithm whereby messages are exchanged locally along edges of the factor graph until convergence (assumed but not guaranteed in loopy graphs). In our implementation, we use synchronous MP analogous to the process outlined in Algorithm 2, and shown in Figure 2.5.



(a) Step 1: Send all messages from variables to factors



(b) Step 2: Send all messages from factors to variables

Figure 2.7: Illustration of one iteration of synchronous MP. Note that, because g_t is singly connected and all particles are initialized with the same weights, g_t continually sends uninformative "flat" messages to x_t, with the effect that g_t can be ignored in the MP framework.

At the variable nodes, computing the outgoing messages is simple. The first step in arriving at them is to compute the variables' beliefs, which are also used to determine convergence (see Section 2.3.2). For nodes m_i and x_t, on their respective domains ($m_i \in$

$\{0,1\}$ and $x_t \in \{x_t^1, \ldots, x_t^K\}$) the beliefs are given by

$$b_i(m_i) \propto \prod_{(t,n)\in\mathcal{F}(i)} U_{(t,n)\to i}(m_i), \qquad b_t(x_t) \propto \prod_{n=1}^{N_t} V_{(t,n)\to t}(x_t), \tag{2.21}$$

where $U_{(t,n)\to i}$, $V_{(t,n)\to t}$ are incoming messages from $f_{t,n}$, $\mathcal{F}(i) = \{(t,n) : i \in \mathcal{M}(t,n)\}$ indexes the $\{f_{t,n}\}$ neighboring m_i, and the beliefs are normalized to sum to one. These nodes' outgoing messages to factor node $f_{t,n}$ can then be written as

$$u_{i\to(t,n)}(m_i) \propto \frac{b_i(m_i)}{U_{(t,n)\to i}(m_i)}, \qquad v_{t\to(t,n)}(x_t) \propto \frac{b_t(x_t)}{V_{(t,n)\to t}(x_t)}, \tag{2.22}$$

which are also normalized to sum to one after computation.

In the other direction, computing the factor-to-variable messages is more complicated. For example, naively applying (6) in [36] to the calculation of the messages from $f_{t,n}$ to x_t, and evaluating at $x_t = x_t^k$, yields

$$V_{(t,n)\to t}(x_t^k) = \sum_m p(z_{t,n}|m, x_t^k) \prod_{j\in\mathcal{M}(t,n)} u_{j\to(t,n)}(m_j), \tag{2.23}$$

which involves the summation of $2^{|\mathcal{M}(t,n)|}$ terms. Since $|\mathcal{M}(t,n)|$ (the degree of $f_{t,n}$) counts the number of cells intersected by SNR measurement (t,n), which in our setup is often in the hundreds, directly evaluating the above expression is unfeasible. However, upon substitution of the binary sensor model (2.2), due to the normalization of the incoming messages, the above expression simplifies to

$$V_{(t,n)\to t}(x_t^k) \propto F\left(z_{t,n}, \gamma_{t,n}^k\right) \tag{2.24}$$

where we have reused the likelihood function $F(\cdot,\cdot)$ in Equation 2.12 and

$$\gamma_{t,n}^k = \prod_{j\in\mathcal{M}(t,n,k)} u_{j\to(t,n)}(0), \tag{2.25}$$

which is *linear complexity* in $|\mathcal{M}(t,n,k)|$. Note that this simplification is accomplished via very similar mechanics (and carries similar intuition to) that of Equation 2.11 in Theorem 2.1 (see Proof A.1 for more details).

Likewise, for the message from $f_{t,n}$ to m_i we initially have the (even more) complicated expression

$$\begin{aligned} U_{(t,n)\to i}(m_i) = &\sum_{m\setminus m_i}\sum_{k} p(z_{t,n}|m, x_t^k) \\ &\times v_{t\to(t,n)}(x_t^k)\prod_{j\in\mathcal{M}(t,n)\setminus i} u_{j\to(t,n)}(m_j) \end{aligned} \tag{2.26}$$

However, again by consequence of the binary sensor model (2.2), the above expression reduces to

$$U_{(t,n)\to i}(m_i) \propto \alpha_{t,n,i} + \beta_{t,n,i}(m_i) \tag{2.27}$$

where

$$\alpha_{t,n,i} = \sum_{k\notin\mathcal{K}(t,n,i)} v_{t\to(t,n)}(x_t^k)\ F\left(z_{t,n}, \gamma_{t,n}^k\right), \tag{2.28}$$

and where

$$\beta_{t,n,i}(m_i) = \begin{cases} \displaystyle\sum_{k\in\mathcal{K}(t,n,i)} v_{t\to(t,n)}(x_t^k)\ F\left(z_{t,n}, \frac{\gamma_{t,n}^k}{u_{i\to(t,n)}(0)}\right), & m_i = 0 \\ \displaystyle\sum_{k\in\mathcal{K}(t,n,i)} v_{t\to(t,n)}(x_t^k), & m_i = 1. \end{cases} \tag{2.29}$$

Here, $\mathcal{K}(t,n,i) = \{k : i \in \mathcal{M}(t,n,k)\}$ maintains a list of which particles among $\{x_t^1,\ldots,x_t^K\}$ observe cell m_i when looking at the nth satellite. Again, we note that the simplifications provided are accomplished using very similar mechanics to that of Equation 2.11 in Theorem 2.1, and the messages $U_{(t,n)\to i}(m_i)$ are normalized to sum to one, for convenience.

LBP convergence and belief readout

As in Algorithm 2, convergence of LBP is declared when the mean of all variables' belief residuals falls below a predefined threshold ϵ. To limit oscillations and help ensure that LBP converges, as is common in practice we apply message damping, as in Equation 2.16. The belief residuals of m_i and x_t are defined via the L_1 norm, as in [20] (4.1):

$$R_i \triangleq \sum_{m_i=0}^{1} \left| b_i(m_i) - b_i^{old}(m_i) \right|, \quad R_t \triangleq \sum_{k=1}^{K} \left| b_t(x_t^k) - b_t^{old}(x_t^k) \right|$$

where b_i^{old}, b_t^{old} are the beliefs from the previous iteration. Upon convergence, the approximate SLAM solution is simply taken to be the beliefs of all the latent variables, i.e., the marginal posteriors of the map and poses are estimated as $p(m_i|y,z) \approx b_i(m_i)$ and $p(x_t|y,z) \approx b_t(x_t)$, where the beliefs are normalized to sum to unity so that they represent valid probability mass functions.

Computational complexity

We state without proof that the expression for computational complexity for SLAM is similar to the mapping-only scenario (i.e., Equations 2.17 and 2.18), except for an additional scaling factor of K accounting for the number of particles used to sample the positions $\{x_t\}$. That is, the complexity is specified by Equation 2.17 where we instead have

$$\mathbb{E}\left[\sum_i |\mathcal{N}(i)|\right] < \frac{3NTK}{2}\left(\frac{H}{d}\right)\ln\left(\csc\phi_{\min}\right) \ . \tag{2.30}$$

2.4 Learning the SNR model

An important component of the GNSS SNR mapping problem is developing sound satellite SNR models. As mentioned in [45], it is known that satellite elevation plays a crucial role in the received power under both LOS and NLOS conditions. Hence, applying different SNR models for satellites in different elevation ranges is appropriate.

At the same time, one output of loopy BP from mapping and SLAM algorithms can be viewed as the satellite LOS beliefs (probabilities) $\gamma_{t,n}$, given by Equation 2.13. We can view these outputs as soft assignments under the two channel hypotheses (i.e., satellites are assigned LOS with probability $\gamma_{t,n}$, and NLOS with probability $1 - \gamma_{t,n}$). It is then possible to define an estimation algorithm for SNR model parameters $\theta = (\theta_{LOS}, \theta_{NLOS})$, and weights $w = (w_{LOS}, w_{NLOS})$ (relative occurrences of the models), based on Expectation Maximization (EM) [11], which is an easily implementable, iterative greedy estimation algorithm. That is, we formulate the overall SNR measurement model as the explicit mixture of two components

$$p(z|\theta) = w_{LOS} f_{LOS}(z) + (1 - w_{LOS}) f_{NLOS}(z) \tag{2.31}$$

where by definition, $f_{LOS}(z) \triangleq p(z|\theta_{LOS}, \text{LOS})$, $f_{NLOS}(z) \triangleq p(z|\theta_{NLOS}, \text{NLOS})$, $w_{NLOS} = 1 - w_{LOS}$, and $\theta_{LOS}/\theta_{NLOS}$ are the LOS and NLOS model parameters, such as the Rician/Log-Normal or Nakagami-m/Generalized-K models defined in Section 2.2.1.

One iteration of the EM based estimation algorithm is provided in Algorithm 3, with results provided in Section 2.5.4 and Appendix A.3. Note that:

1. The "E step" is based on the expectation of the log likelihood under the parameters and given the assignments, i.e.

$$E\left[\log L(\theta|z_1, ..., z_n)\right] = \sum_{i=1}^{N} p(z_i) \log p(z_i|\theta). \tag{2.32}$$

2. The "M step" is based on maximizing this expectation over parameter vector θ. For the candidate models in Section 2.2.1, an easy way to approximate this maximization is moment matching (see e.g. [61]), but in some cases, exact maximum likelihood (ML) estimation may be possible.

3. In practice EM can be run in a loop, with LBP followed by EM estimation repeatedly until convergence of the parameters.

4. The model parameters determined by EM are implicitly conditioned on the OG map estimated by LBP, and hence are generally only valid in that specific 3D environment.

5. The satellite measurements can be binned by satellite elevation, constellation/class, or carrier frequency to determine more specific models. In Section 2.5.4, we condition the models by elevation, for example.

2.5 Mapping and SLAM Experiments

Our 3D mapping and SLAM algorithms were validated in small scale experiments around the University of California, Santa Barbara (UCSB), as well as in one larger

Algorithm 3 Learning the SNR model using EM

1: input (LOS probability, SNR pairs from LBP $(\gamma_i, z_i)_{i=1}^N$)

2: $w_{LOS} = \frac{1}{N}\sum_{i=1}^N \gamma_i$

3: $w_{NLOS} = 1 - w_{LOS}$

4: $\theta_{LOS} = \arg\max_\Theta \sum_{i=1}^N \gamma_i \log p(z_i|\Theta, \text{LOS})$

5: $\theta_{NLOS} = \arg\max_\Theta \sum_{i=1}^N (1-\gamma_i) \log p(z_i|\Theta, \text{NLOS})$

6: **return** optimized SNR model $\theta = (\theta_{LOS}, \theta_{NLOS})$, $w = (w_{LOS}, w_{NLOS})$

scale experiment in downtown Santa Barbara. In this section, we present and discuss experimental results.

2.5.1 UCSB mapping experiment

Our 3D mapping algorithm was validated on GNSS measurement data taken outdoors from the eastern corner of the campus of the University of California, Santa Barbara (UCSB). The device used was a Samsung Galaxy Tablet 2.0 running on the Android operating system supporting both GPS and GLONASS. The overall data-set consisted of 14 streams of measurements taken over a few days, with each containing an average of 12 minutes of continuous measurement data arriving at 1 Hz. Reported satellite SNR readings ranged in the interval $7 - 48$ dB, and an average of 14 distinct satellites were in view at any point in time.

Parameter selection

A grid resolution of 4 m was selected and the map height was set to 24 m. For the LBP-based inference step, we used synchronous message passing [15] with a convergence threshold of 10^{-3}. To limit oscillations and aid convergence, 0.4 message damping was used (see [40] for a detailed explanation of damping). Referring to Section 2.2.1, the Rician/Log-Normal SNR model was used, and the total received power Ω under the LOS hypothesis was estimated as the maximum of all linear SNR readings $10^{z_{t,n}/10}$ from the same satellite during the same time window. For simplicity, we used a constant $K = 2$, indicating moderate fading conditions. Guided roughly by the results in [39, 45], the expected received power for NLOS links was set to 18 dB less than the "reference value" Ω, and the standard deviation was set to 10 dB, reflecting a large variability in shadowing conditions. Note that, although signals from low elevation satellites have wider power fluctuations [45], for simplicity, the widths of both LOS/NLOS distributions were fixed across satellite elevations. However, for the same reason and much as in [34], a threshold satellite elevation of 10° was chosen, below which SNR measurements were discarded. When visibility to a particular satellite was temporarily lost in the middle of an observation window (presumably most often caused by total occlusion), the satellite coordinates were interpolated and LOS/NLOS likelihoods of 0.1/0.9 were used.

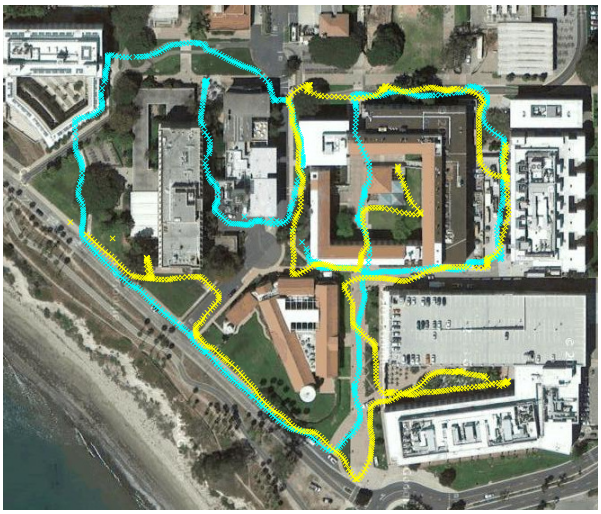
Discussion of results

Altogether, the 14 data-sets comprised of 1.4×10^5 SNR measurement rays and 4.6×10^4 map cells. However, the total execution time of our algorithm (ray tracing, graph indexing, LBP) was just under 5 minutes, carried out using Matlab on a 64 bit PC with 4 GB of memory and a 2.40 GHz Intel Core i7-3517U processor.

A horizontal layer (4-8 m) of the resulting map – the *raw LBP output* – is compared to an aerial view of the same area in Figure 2.8. White and black areas correspond to estimated occupancy probabilities close to zero and one, respectively, with shades of grey denoting values in between. Blue cells are unexplored regions (those not intersected by any measurement rays). Green borders enclose the region in which measurements were taken, i.e., surround the feasible mapping region. The building contours, shown in red, were obtained from OpenStreetMap (OSM). Though the raw map slice in Figure 2.8 contains errors (some of which are possibly caused by receiver localization errors), approximate building locations and several large trees can be clearly identified.

Note that open areas are more easily (and *better*) mapped than occupied ones with our sensing model: while *every* cell a LOS signal passes through is empty, a NLOS signal only informs that *some* occupied cell(s) occluded it. Hence, if a NLOS ray spans many occupied cells, then each cell will have a tendency to "blame" (via message passing in LBP) other occupied ones for the blockage, contributing to uncertainty. Mathematically, this can be gleaned from message and belief formulas (2.11)–(2.9). If a NLOS ray penetrates many cells $m_j, j \neq i$ with occupancy beliefs around 0.5 or higher, then $\alpha_{t,n,i}$ will be

(a) Generated map



(b) Aerial view

Figure 2.8: Horizontal slice (4-8 m above ground level) of generated map compared to Google Maps aerial view of the same area. In the aerial view, receiver paths from two typical datasets are shown.

very small and the message $U_{(t,n)\to i}(m_i)$ will be proportional to the uninformative value $[1, 1]$. If m_i receives many such uninformative messages, then its occupancy belief will become 0.5 as well. Large obstacles can therefore be expected to result in "gray zones" with occupancy beliefs around 0.5, whereas empty space is more easily identified with high certainty. With this in mind, we expect that thresholding and/or image segmentation schemes applied to the output of the LBP algorithm may improve the map. While such post-processing strategies are beyond the scope of this paper, they are an important topic for future work.

3D mapping capability: The 3D mapping capability of our algorithm is demonstrated in Figures 2.9 and 2.10, showing the generated map around one building (Kohn Hall). Although the building in reality is about 10 m tall, the lingering region of cells above 12 m (red tinted cells) can be explained by the fact that only ground-level GNSS measurements were used, so that no LOS satellite signals glanced the middle part of the building roof (this is a limitation for any ray-based sensing method). Nevertheless, it can be concluded from the map data that Kohn Hall is no taller than 20 m, and for the most part shorter than 12 m. Separately, we note the empty (transparent) space surrounding the building that is correctly identified.

Effect of increasing data: We explored the evolution of the map around another building, Harold Frank Hall, as more data becomes available in Figure 2.11. As expected, these results show that using a larger amount of measurement data improves the overall quality of the map and especially allows for better mapping of empty space.

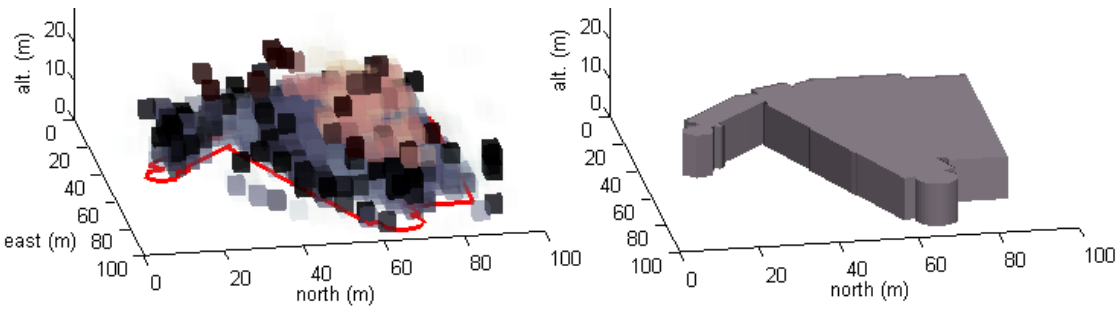
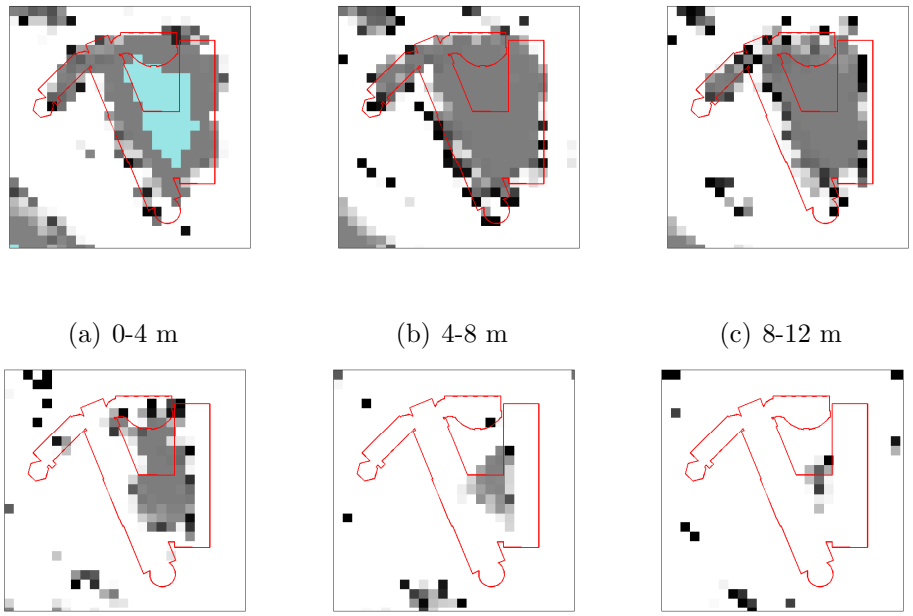


Figure 2.9: Left: 3D occupancy map around Kohn Hall, with cell transparency set to emptiness probability (i.e., areas assigned low occupancy probabilities are transparent), and red tinted cells being those above 12 m. Right: approximate ground truth based on OSM data.



(a) 0-4 m (b) 4-8 m (c) 8-12 m

(d) 12-16 m (e) 16-20 m (f) 20-24 m

Figure 2.10: Horizontal slices of the generated map around Kohn Hall at UCSB.

(a) 3 datasets (b) 7 datasets (c) 14 datasets

Figure 2.11: Generated maps (4-8 m above ground level) around Harold Frank Hall as a function of the quantity of measurement data used.

2.5.2 UCSB SLAM experiment

Our SLAM algorithm was also validated on GNSS measurement data taken outdoors from the eastern corner of the campus of UCSB (see Figure 2.12). A Samsung Galaxy Tablet 2.0 running the Android operating system was used as a GPS/GLONASS collecting device. Over the course of several days, we gathered a total of 34 test datasets, where the duration of each test ranged from 3 to 23 minutes and measurement data was logged at 1 Hz. During a typical test, SNR readings in the range of $7 - 48$ dB were recorded and approximately 13 satellites were visible at any one time.

To summarize the parameters used, a grid size of 5 m was selected, and the map height was set to 30 m. The SNR model used is identical to the one detailed in Section 2.5.1, and an identical elevation mask of 10° was applied.

In all, the 34 datasets comprised of 5.0×10^3 position samples and 6.5×10^4 SNR measurements interacting with 4.2×10^4 grid cells. To mitigate the effects of spatially

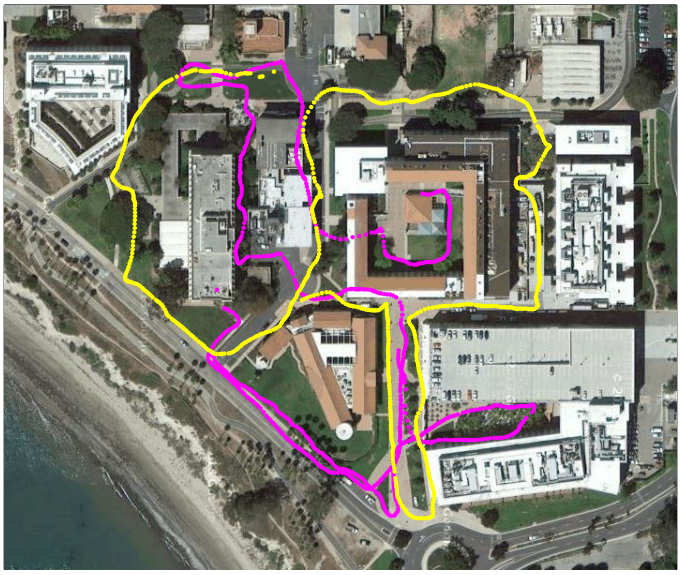


Figure 2.12: Google Maps aerial view of the eastern portion of the UCSB campus. Two typical receiver paths as logged by the GNSS collecting device are shown in yellow (shorter test run) and magenta (longer test run).

correlated geolocation errors, the 5.0×10^3 position samples represent a factor of 5 downsampling of the original datasets. Although in our experiment data was essentially thrown away by downsampling, in a real-world application, sparse sampling would allow for potential energy savings at the GNSS receiver. To represent the possible receiver positions for each time sample, we used $K = 30$ quasi-random particles (drawn from a Sobol sequence [41] which was then transformed to a set of Gaussian samples). Compared to drawing independent samples, low discrepancy sampling more uniformly separated the particles, allowing us to use smaller K. For the LBP-based inference step, $\rho = 0.4$ damping and a convergence threshold of $\epsilon = 10^{-3}$ were used. With these settings LBP terminated after 60 iterations, taking 13 minutes on a 64 bit PC with a 3.20 GHz Intel

Core i7 processor and 32 GB of RAM running Matlab R2013b. During runtime, total system memory usage never exceeded 6 GB.

Discussion of results

A portion of the resulting map – that is, the unprocessed LBP output – can be seen in Figure 2.13. The dark areas are those where the probability of occupancy is close to one, the white areas represent likely open-space, and the light blue regions are unobserved. The green border encloses the feasible mapping region (borders the convex hull of all position samples). Data from OSM was used to provide building contours, shown in red. Although the map includes errors, it closely resembles the aerial view in Figure 2.12. Qualitatively, the resulting map looks similar to the output from the mapping-only algorithm (Section 2.5.1), but is a bit less noisy and contains fewer isolated (outlier) occupied pixels.

3D mapping capability: To assess the 3D mapping capability of our algorithm, the first 6 layers of the resulting map are shown in Figure 2.14 for the area around Kohn Hall, which is approximately 10 meters in height. As can be seen in Figure 2.15(a) and 2.15(b), the outline of the building is well approximated by the map. Again, in comparison, the results are similar to those of the mapping-only algorithm (see Figure 2.10), however, they are bit less noisy and the building footprint is more accurately detected, particularly between 5-10m.

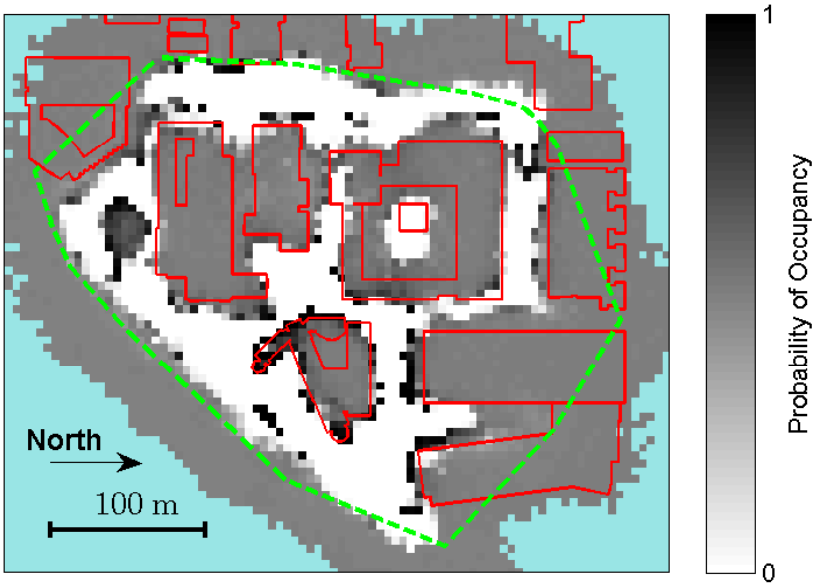
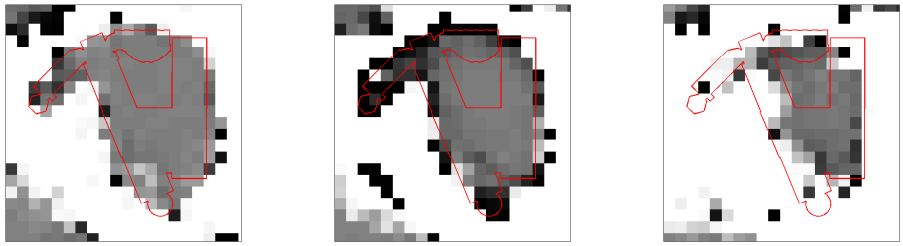


Figure 2.13: Horizontal slice (5-10 m above ground level) of generated map covering the eastern portion of the University of California, Santa Barbara campus. The measurement region borders are marked in green, and building contours obtained from OSM are shown in red.

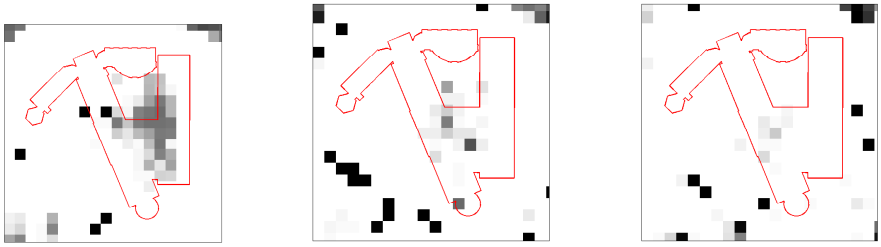
In addition, the 3D rendering of the occupancy grid map compared against ground truth from OSM, is shown in Figure 2.5.2. Here similar sets of shadows for two satellites are displayed, showing the utility of the 3D mapping algorithm for positioning purposes (using approaches such as Probabilistic Shadow Matching, as discussed in Chapter 3).

Positioning improvement: Finally, an example of the localization improvement can be seen in Figure 2.16, which shows a portion of a dataset isolated for analysis. While positioning uncertainty is lower after LBP in general, of particular interest are the points along the west side, north-west corner, and north side of the building. The proposed algorithm assigns low weights to particles that are in the building, and more highly weights those on the sidewalk near the true path, reducing uncertainty and improving

Figure 2.14: Horizontal slices of the generated map around Kohn Hall at UCSB.



(a) 0-5 m (b) 5-10 m (c) 10-15 m



(d) 15-20 m (e) 20-25 m (f) 25-30 m

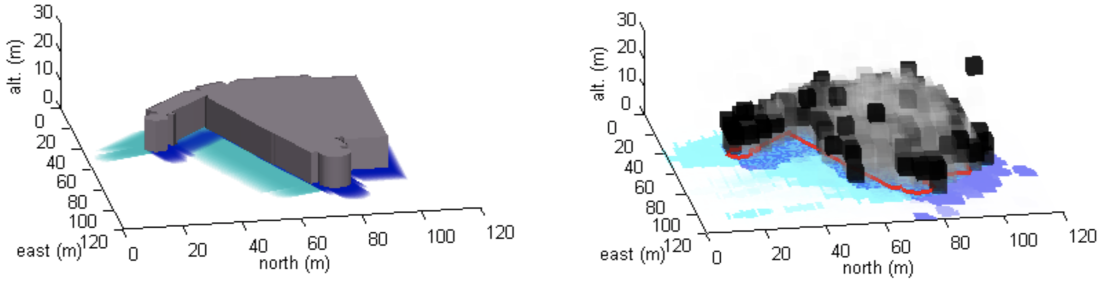


Figure 2.15: Comparison of shadows generated from (Left) approximate ground truth based on OSM data; (Right) SLAM-estimated 3D occupancy map around Kohn Hall, with cell transparency set to emptiness probability (i.e., areas assigned low occupancy probabilities are transparent).

overall geolocation accuracy. Without this position correction, the resulting map would have underestimated the occupancy of the cells on the northern wall of the building. Separately, we note that receiver positioning on the south side of the building on the sidewalk is more or less unimproved by our algorithm. This is to be expected because the positioning uncertainty there was lower to begin with (since fewer signal-occluding buildings are nearby).

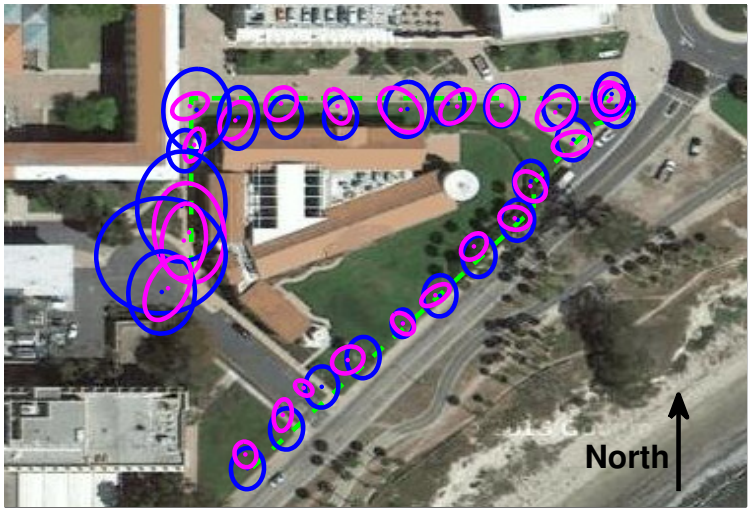


Figure 2.16: Positioning improvement around Kohn Hall. The true receiver path is marked with the dashed green line, with blue/magenta ellipses representing positioning uncertainty before/after LBP.

2.5.3 Downtown Santa Barbara SLAM experiment

The SLAM algorithm was also tested using a larger 25 hour dataset collected using 4 Android devices (2 tablets and 2 smartphones) in downtown Santa Barbara, using OpenStreetMap building footprints as a priori occupancy information for the first layer

(0-4m) of the map with 80% occupancy probability, and for the second layer (4-8m) of the map with 60% probability. In this case, to maximize the amount of input data, a 2x downsampling of the input data was performed (instead of 5x as Section 2.5.2). An identical SNR model was used as in Sections 2.5.1 and 2.5.2.

The results of this experiment can be seen Figures 2.5.3 and 2.5.3, which show the traces and the generated Occupancy grid map for about 25 hours of input data from 4 Android devices in downtown Santa Barbara. In these figures, white/black corresponds to areas identified as empty/occupied, with shades of grey in between.

Figure 2.17: (Left) OpenStreetMap top down view of the area of downtown Santa Barbara that was mapped; (Right) Google Maps aerial view of it, with GNSS traces in red and mapped region outlined in yellow.

2.5.4 UCSB SNR model learning experiment

Using the dataset collected for our UCSB SLAM experiment, we followed the EM-based SNR model learning procedure (Section 2.4) for the Nakagami-m/ Generalized-K

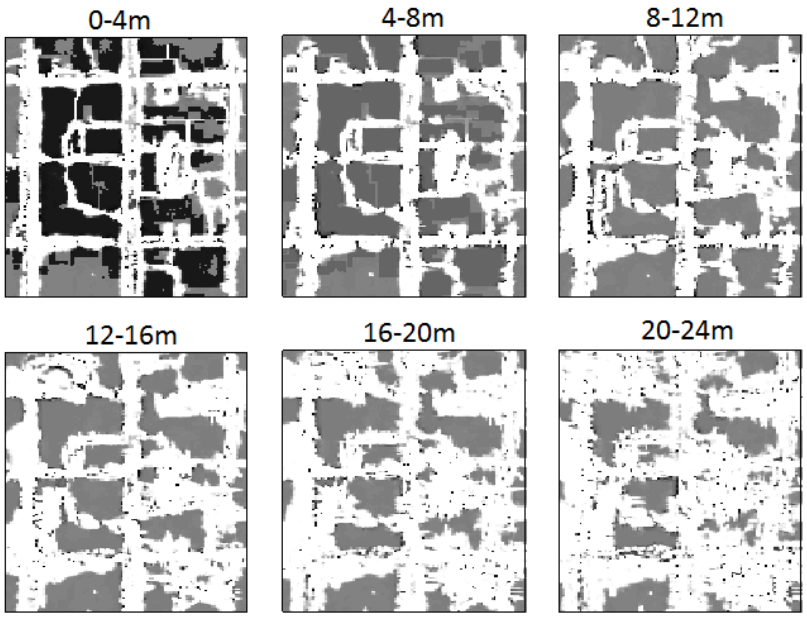


Figure 2.18: Layers of the generated occupancy map of downtown Santa Barbara.

SNR model (Section 2.2.1). For this evaluation, we binned the satellites by elevation, with 8 bins each of width 10 degrees ($10° - 20°, ..., 80° - 90°$[1]).

To validate the fits, we first plot the empirical SNR distributions under LOS and NLOS measurements, using the SLAM BP output and thresholds of LOS probabilities of 0.1/0.9 to classify LOS/NLOS. As can be seen in Figure 2.19, there is a clear relationship of the received SNR on the satellite elevation in both LOS and NLOS scenarios. Furthermore, the histogram for NLOS measurements is skewed to lower SNRs – motivating the use of the left-skewed Generalized-K model instead of the symmetric (0-skew) log normal model.

For one example elevation bin ($20° - 30°$), the computed SNR model fits, overlayed against raw data for LOS, NLOS, and all measurements, as well as comparisions of

[1] $0° - 10°$ is not represented as it falls below our elevation mask of $10°$

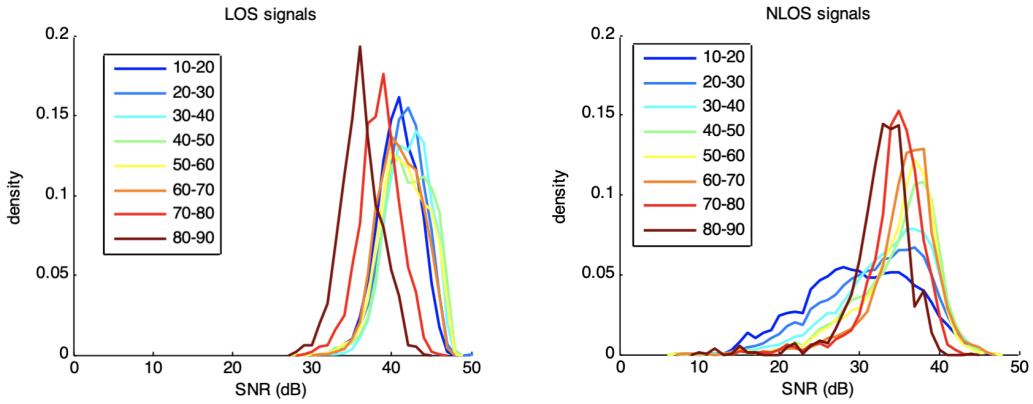


Figure 2.19: Effect of satellite elevation (degrees) for LOS and NLOS SNR measurements.

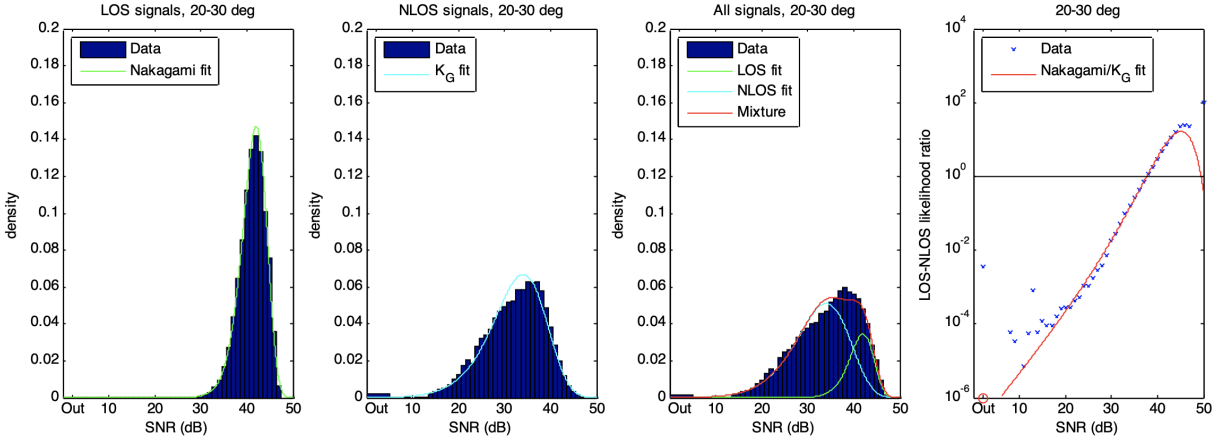


Figure 2.20: Example SNR model fit for a single satellite elevation bin.

LOS/NLOS likelihoods for data and fits (for normal and outage scenarios), is shown in Figure 2.20.

A full set of parameter fits can be seen in Appendix A.3. The aggregated empirical fits, along with cubic spline regressions of the parameter values for the Nakagami-m and Generalized-K (K_G) models is shown in Figure 2.21. Finally, versions of the Loopy

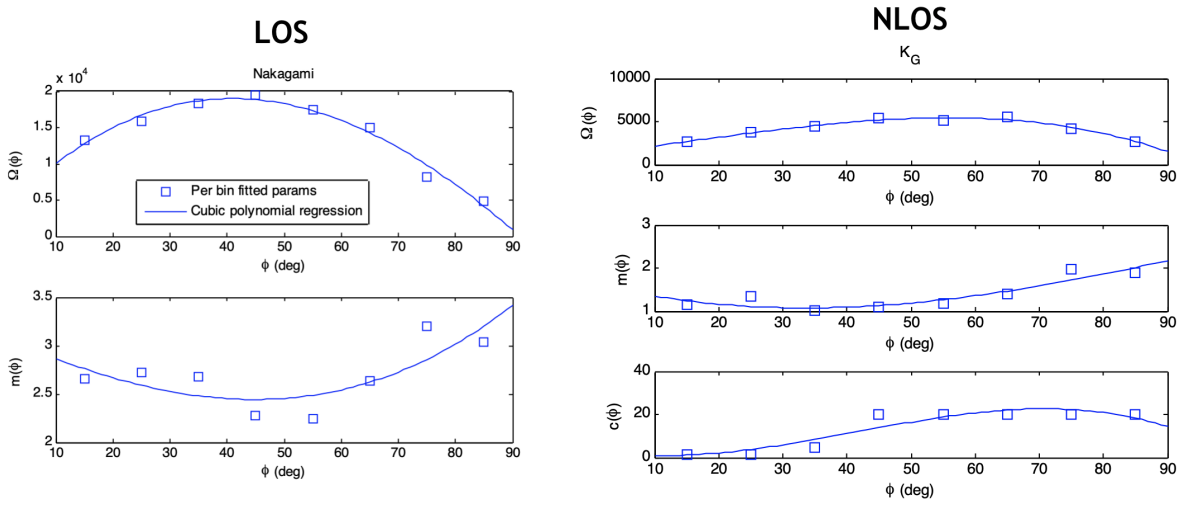


Figure 2.21: SNR model fits for LOS (Nakagami-m) and NLOS (Generalized-K) models and cubic polynomial regressors.

BP-based SLAM output, before and after applying the fitted SNR models, can be seen in Figure 2.22, with interesting differences highlighted in yellow. Here, it can be seen that the optimized ("after") SLAM output is slightly more aggressive and accurate at detecting empty vs occupied space, hinting at it being able to accomplish more accurate estimation with the same amount of input SNR data.

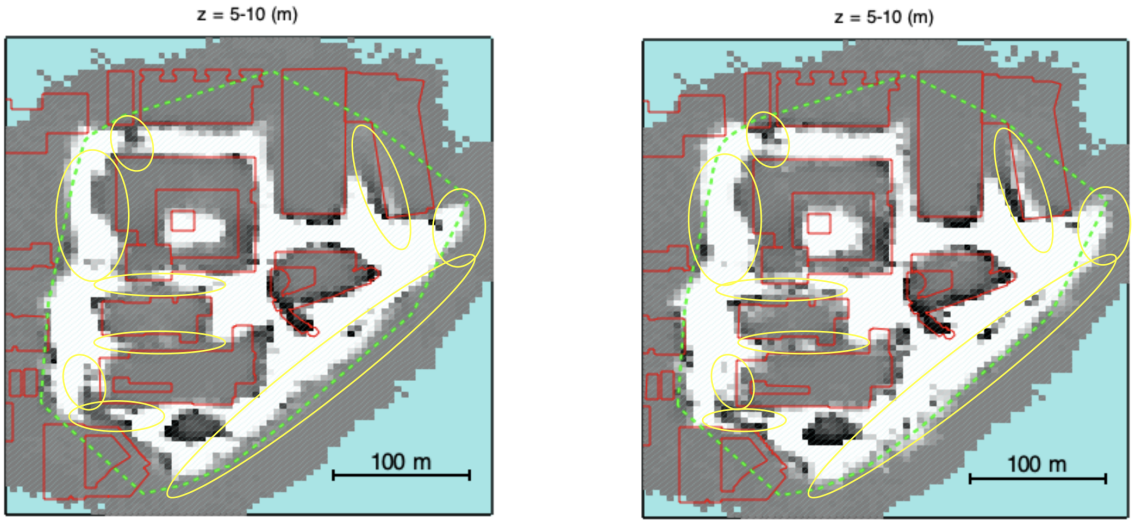


Figure 2.22: Single layer of SLAM 3D map output before (left) and after (right) applying the optimized SNR model.

Chapter 3

Localization using GNSS SNR measurements

For decades, global navigation satellite systems (GNSS) have been widely used for geolocation purposes. For the vast majority of end users, the most important information provided by GNSS receivers is the so-called Position-Velocity-Time (PVT) solution. As a time-of-arrival (TOA) localization system, the PVT solution is computed using satellite range and range-rate (Doppler) observations ρ_t^n and $\dot{\rho}_t^n$ at epochs t across N_t satellites n:

$$\begin{aligned} \rho_{t,n} &= ||x_t - r_{t,n}|| \ + \ c\tau_t \ + \ e_{t,n} \\ \dot{\rho}_{t,n} &= ||\dot{x}_t - \dot{r}_{t,n}|| \ + \ c\dot{\tau}_t \ + \ \dot{e}_{t,n}, \qquad n = 1, ..., N_t \end{aligned} \tag{3.1}$$

where $r_{t,n}$, $\dot{r}_{t,n}$ are the nth satellite's position and velocity at time t, $e_{t,n}$ and $\dot{e}_t^n$ are measurement errors, and c is the speed of light (approximately 3×10^8 m/s). The PVT solution refers to the tuple $(x_t, \dot{x}_t, \tau, \dot{\tau}_t)$, which more precisely represents:

Parts of this chapter are reprinted from our conference publications [29, 32, 28] with permission.

- **Position:** Three-dimensional Cartesian position coordinates x_t of the receiver (which can be transformed to geographic coordinates latitude, longitude, and altitude);

- **Velocity:** Receiver Cartesian velocity vector $\dot{x}_t$ in three dimensions, which for ground-based vehicles is typically provided as the transformed (projected) two dimensional velocity vector over the ground plane;

- **Time:** Precise time estimates (clock biases τ_t and drifts $\dot{\tau}_t$) for each distinct GNSS clock used to timestamp pseudorange observations.

As described in Chapter 7 of [33], each of these quantities can map to a geometry (or linear independence, condition number) factor called "dilution of precision" (DOP), which represents the quality of the solution, and is good (low) when many satellites are available and they are widely separated in the sky.

As a nonlinear system of equations (due to the Euclidean Norm $||\cdot||$ that is present), the PVT solution is typically estimated using techniques such as non-linear (or iterated linearizing) least squares or linearizing Bayesian estimators such as the Extended Kalman Filter [52], which effectively perform trilateration under the assumption that satellites measurements are received under direct (line-of-sight) conditions (the reader is referred to [33] for more details). In an open field where dozens of satellites across multiple constellations are available, using a standard single frequency (single band) GNSS receiver, the PVT is over-determined and typically accurate within 3 meters. With newer dual

frequency receivers that use new L5 GPS or E5 Galileo signals, for example, localization with errors less than 1m is possible in such open-sky conditions.

PVT in the urban setting: Unfortunately, PVT solutions are generally inaccurate in urban environments due to frequent non-line-of-sight (NLOS) channel conditions. This is due not only to poor geometry of the available satellites (causing increased dilution of precision in the non-linear system), but also due to multipath signal reception, where GNSS receiver algorithms are sometimes unavoidably fooled into interpreting reflected (NLOS) received signals as line-of-sight (LOS), incorporating extra path delay and range-rate error in the trilateration calculations. This phenomenon poses a severe problem for mobile services that benefit from accurate urban localization, such as navigation, hyper-local advertising, geofencing applications, autonomous driving, and ride-sharing, often producing large (tens or hundreds) meters of GNSS position error.

Satellite SNR: An important set of satellite measurements often not utilized in full capacity, and the subject of much of this chapter, are the *signal strengths* of each satellite in view. In the literature, these are often referred to as Carrier-to-Noise (CN0) or Signal-to-Noise ratio (SNR), and are necessarily computed by the GNSS receiver in order to select and weigh pseudoranges and pseudorange-rates across satellites when computing PVT solutions. The reason for this is straightforward:

- Measurements from high SNR satellites are less likely to be NLOS and corrupted by multipath reflection (since multipath fading on average decreases the combined received signal amplitude);

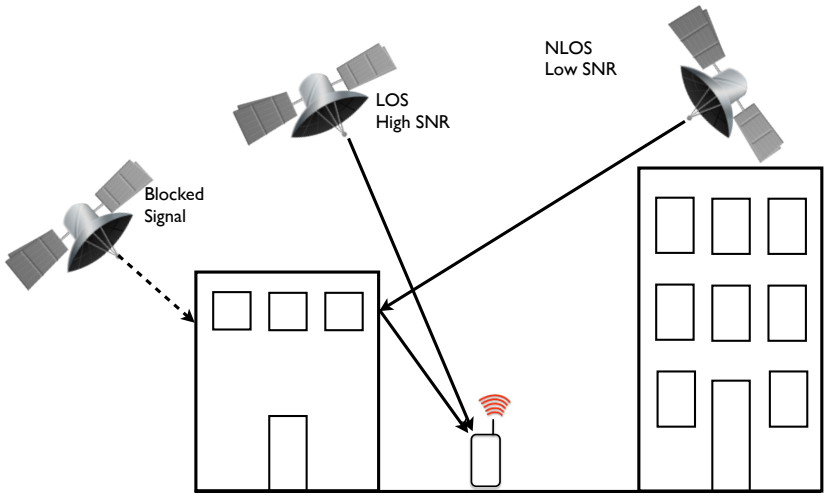


Figure 3.1: Satellite SNR readings depend on LOS/NLOS path between receiver and satellite. NLOS paths are often characterized by low SNR.

- LOS satellite signals with higher SNR are by definition less corrupted by noise, and so generate higher quality observations.

Thus, as the LOS to some various satellites is occluded by obstacles, such as buildings, trees, etc., NLOS signals are measured, which are characterized by a lower SNR than LOS signals. This very simple observation can be used to improve positioning using a technique called Shadow Matching: Given an approximate receiver PVT-based solution and a description of surrounding 3D topology (map), it is possible improve the position estimate by matching the satellite SNR measurements to areas inside or outside of the various satellite shadows. Conversely, signal strengths can provide geographical information about obstacles, as described in Chapter 2.

In this chapter, we demonstrate algorithms, architectures, and experimental results that leverage how GNSS devices can be viewed as passive environment sensors, relating

the state of the world (the 3D map) and true position of the GNSS receiver via sets of coarse (noisy) satellite SNR measurements and a coarse (noisy) PVT solutions. Namely, we explore *principled and practical* systems for improving positioning against known (or partially known) 3D maps, using a technique called Probabilistic Shadow Matching (Section 3.2).

3.1 Related work

The notion of refining GNSS positioning estimates using *known* environment maps is well researched to date, and in the literature is known as *Shadow Matching* (SM). Essentially, SM involves classifying signals as LOS/NLOS and matching their points of reception to areas outside/inside the "shadows" of signal-blocking buildings, thereby constraining the space of possible receiver locations. Our work differs with respect to past works in Shadow Matching [60, 5, 42, 23, 22, 59] in terms of filtering algorithm, map representation, likelihood surface estimation, and integration of GNSS fixes, as follows.

Filtering algorithm: Most SM approaches rely on different flavors of nonlinear Kalman Filters (e.g. Extended and Unscented Kalman Filters) for coarse inference [60, 5, 42, 23, 59]. These are commonly used for GNSS tracking, but poorly fitted for SM, which involves a highly non-linear measurement model. One exception is [22] which proposes a grid filter for tightly integrated GNSS and SM; another [59] which describes a particle filter approach (the same approach we use). However, grid filters are poorly suited and inefficient in tracking in multiple dimensions, wasting lots of points in low-likelihood

regions of space. Unlike the PF in [59], we promote a clean, Rao-Blackwellized [13] design that leverages the fact that in SM only position states are observed, so that velocities can be represented analytically (using Gaussian Kernels). Rather than sampling from the likelihood surface solely as [59] proposes, we instead sample from a locally optimal (per particle) mixture distribution, and improve robustness through primal-dual particle sampling. We also describe an alternative, simple Gaussian particle filter (GPF) [3] implementation. A very important difference is that [59] also does not incorporate a motion model, estimating position states only.

Map representation and ray tracing: Unlike in other approaches, ray tracing component is performed on the back-end (server-side, instead of client-side, or on device) against probabilistic 3D maps. This is an important detail which allows us to account model uncertain and varying levels of blockage due to obstacles in the map. Moreover, a ray tracing caching system is described. Keeping map data invisible from the client is important for practical reasons as well: 3D maps and shadowing information is extremely dense and costly to transport over mobile networks.

Likelihood surface estimation: Similar to our grid representation of the likelihood surface, [22] achieves a similar effect via a grid filter. However, an important distinction is that we run a particle filter against a grid representation of the current measurement likelihood surface.

GNSS PVT integration: Integration of the coarse GNSS fixes is not described in [59] except in order to bound the search space through which SM is performed. In

contrast, we integrate a practical non-linear model for GNSS fixes that incorporates non-Gaussian outliers, dynamically changing dilution-of-precision (DOP), and environment conditions (how built-up the area of interest is). Through a deeper integration with GNSS, [22] does account for GNSS PVT solutions and confidence.

3.2 Bayesian localization model

The localization improvement problem can be described as follows: Given a stream of noisy GNSS SNR and location fixes for a device, denoted $z = (z_1, ..., z_t)$ and $y = (y_1, ..., y_t)$, along with an uncertain (noisy) estimate of the map m, what is the best estimate of the device's current location x_t? And how confident are we in that estimate?

For positioning, as in mapping and described in Section 2.2, we use an occupancy grid to represent the map. Denoting the occupancy probabilities of the map cells around the device as $o(m)$, and treating these as additional measurements, the quantity we are then interested in is $p(x_t|z, y, o(m))$. In thesis, we describe a particle filtering approach which allows one to apply Shadow Matching (SM) against the occupancy map in a recursive, real-time fashion, and purely in software at the mobile application level.

3.2.1 System model

A filter in the context of a state space model is characterized by two models that we will term the motion and the measurement models. Under the first order Markov assumption, whereby the current state only depends probabilistically on the previous

state, the motion model is generally described by

$$x_t = f(x_{t-1}, u_t) \tag{3.2}$$

which maps the previous state x_{t-1} (position, velocity, etc.) and a control u_t to the current state x_t. In the case of this thesis, it is assumed that u_t is completely unknown (we don't know our users' intentions) so it is modeled as zero mean noise. The measurement model is

$$[y_t, z_t]^T = g(x_t, v_t) \tag{3.3}$$

which maps the current state x_t and random noise v_t to the observed GNSS fix y_t and N_t SNR measurements $z_t = [z_{t,1}, \ldots, z_{t,N_t}]^T$. The overall system model forms a Markov Chain, whose Bayesian Network is shown in Figure 3.2.

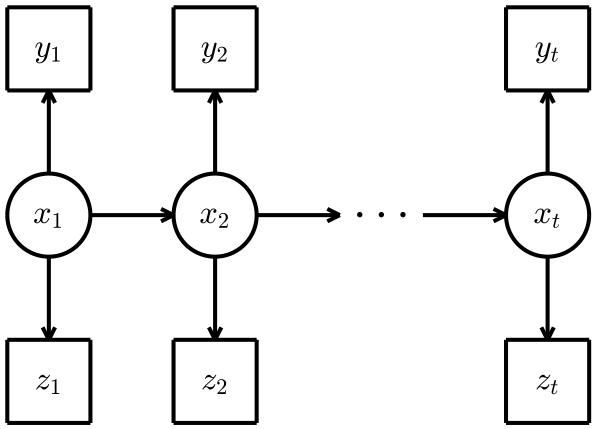


Figure 3.2: Bayesian network for the localization system model. States (circles) generate measurements (squares) at each time instant t over the measurement model $g(\cdot)$. States produce the next states over the motion model $f(\cdot)$.

A particle filter aims to estimate the receiver state conditioned all previous measurements (posterior distribution) which is modeled as a set of K particles

$$p(x_t|y_{1:t}, z_{1:t}, o(m)) \approx \sum_{k=1}^{K} w_t^k \delta_{x_t^k}(x_t) \triangleq \mathcal{X}_{1:t} \tag{3.4}$$

(which, we denote as $\mathcal{X}_{1:t}$ as shorthand). By doing so the PF can (loosely speaking) handle a wide variety of non-linear, non-Gaussian measurement and motion models. In our application, the primary motivation to use a PF comes from the measurement model: by ray tracing from discrete particle locations, PFs allow us to directly model arbitrarily shaped satellite shadows. (This often leads to multi-modal posterior distributions for which PFs are well suited, as well).

State space modeling and motion model

Although Probabilistic Shadow Matching is straightforward to extend to the general case of full three-dimensional motion, in this thesis, for simplicity we focus on the GNSS-based pedestrian navigation outdoors. In this scenario, ground level localization with *mostly* horizontal motion is observed. For this case, we explicitly set the state vector to be "2.5D"[1], i.e., we nominally use $x = [r_x, \dot{r}_x, r_y, \dot{r}_y, r_z]^T$, where r_e and $\dot{r}_e$ refer to position and velocity along axis e. The reference frame for this state space is the same one we use for the OG map, as described in Section 2.2.2.

A rule of thumb with PFs is that an order of magnitude more particles is required for each additional dimension. To cover the localization space with fewer particles (to

[1] "2.5D" in the sense that it describes horizontal continuous (2D) motion in the 3D space

require less computation and memory consumption), we model the position coordinates as particles and the non-position coordinates as distributions (Gaussians) so that

$$p(x_t|y_{1:t}, z_{1:t}, o(m)) = \sum_k w_t^k N(x_t|x_t^k, \Lambda_t) \tag{3.5}$$

where the covariance Λ_t is singular (zeros filling rows and columns) in the non-position dimensions.

For the Rao-Blackwellized PF [13] (more details below) it is most straightforward to use a linear Gaussian motion model of the form

$$x_t = \Phi_t x_{t-1} + u_t \tag{3.6}$$

where Φ_t is the state transition matrix and $u_t \sim N(0, Q_t)$ is the process noise. Epochs t and $t-1$ are typically separated by about one second. A basic option for the pedestrian motion model is the Nearly Constant Velocity (NCV) model from [38]. For example, the "2.5D" NCV model with $x_t = [r_x(t), \dot{r}_x(t), r_y(t), \dot{r}_y(t), r_z(t)]^T$ is defined as:

$$\Phi_t = \text{diag}\left[\Phi'_t, \Phi'_t, 1\right], \qquad \Phi'_t = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \tag{3.7}$$

where $\text{diag}\left[\cdot\right]$ refers to the block-diagonal matrix formed by its arguments, and T is the time step from epoch $t-1$ to t. The covariance matrix of the process noise for the NCV motion model is

$$Q_t = \text{diag}\left[Q'_t, Q'_t, \sigma_z^2 T\right], \qquad Q'_t = \sigma_{\dot{x},\dot{y}}^2 \begin{bmatrix} \frac{T^3}{3} & \frac{T^2}{2} \\ \frac{T^2}{2} & T \end{bmatrix} \tag{3.8}$$

where $\sigma^2_{\dot{x},\dot{y}}$ and σ^2_z are the power spectral densities (PSDs) of the white noise processes governing horizontal velocity and vertical position, respectively (so that X velocity, Y velocity, and Z position evolve as independent Gaussian random walks).

While the presented motion model is strictly linear, note, however, that by using an Extended Kalman Filter-inspired linearization scheme [52], the RB framework will still work as long as we have

$$x_t = f(x_{t-1}) + u(x_t) \tag{3.9}$$

where $u(x_t)$ is Gaussian (though its parameters can depend on the state) and $f(\cdot)$ is differentiable and smooth.

Measurement models

The measurement model is non-linear and non-Gaussian. Under the common assumption that simultaneous measurements are conditionally independent given the receiver state[2], we have the following factorization of the measurement likelihood function:

$$\begin{aligned} p(y_t, z_t|x_t) &= p(y_t|x_t)\ p(z_t|x_t, o(m)) \\ &= p(y_t|x_t) \prod_n p(z_{t,n}|x_t, o(m)) \end{aligned} \tag{3.10}$$

where for the SNR measurements we have introduced a dependence on the map occupancy probabilities $o(m) = \{p(m_i = 1)\}_i$ which are treated as measurement data.

GNSS fix measurement model: At time t, the GNSS receiver provides a GNSS fix y_t, which contains a reported PV solution m_t and an estimated accuracy value $\sigma_{UERE,t}$.

[2]Also known as the "static world" assumption, whereby the map parameters that also influence measurements (such as $o(m)$) are static.

For localization purposes, we model the overall PV distribution as a mixture (sum) of a Gaussian distribution $\tilde{y}_t \sim N(m_t, C_t)$ and a random outlier vector e_t which comes from a broader multivariate, elliptical distribution centered at m_t:

$$p(y_t) = (1 - \alpha)\ p(\tilde{y}_t) + \alpha\ p(e_t) \tag{3.11}$$

where $E(x_t) = m_t$ (i.e., m_t is treated as an unbiased estimator of the true state x_t), and α is the outlier probability, which can be coarsely adapted by scenario (i.e. α can be made larger for deep urban environments). The covariance matrix C_t can be estimated using typical dilution-of-precision (DOP) techniques (See Chapter 7 in [33]):

$$C_t = \sigma^2_{UERE,t}(H_t H_t^\intercal)^{-1} \tag{3.12}$$

where $\sigma_{UERE,t}$ is the user-equivalent range error (reported accuracy of the receiver), and

$$H_t = [h_{t,1}, \ldots, h_{t,N_t}] \tag{3.13}$$

is a $(3 \times N_t)$ matrix where each column is a unit vector of the form

$$h_{t,n} = \begin{bmatrix} \cos(\phi_{t,n})\cos(\theta_{t,n}) \\ \cos(\phi_{t,n})\sin(\theta_{t,n}) \\ \sin(\phi_{t,n}) \end{bmatrix} \tag{3.14}$$

and $\phi_{t,n}$ and $\theta_{t,n}$ are the satellite elevation and azimuth.

Note that the class of distributions used for e_t can be made non-Gaussian in the general case. For example, we may want to use the multivariate Student T distribution in certain cases for its broader tails, which is defined as (in the 3D position space)

$$p(e_t) = \frac{\Gamma\left[(\nu+3)/2\right]}{\Gamma(\nu/2)\nu^{3/2}\pi^{3/2}\left|\Sigma_t^e\right|^{1/2}} \left[1 + \frac{1}{\nu} e_t^T \Sigma_t^{e-1} e_t\right]^{-(\nu+3)/2} \tag{3.15}$$

where $\nu \geq 1$ is the order (with $\nu = 1$ referring to the multivariate Cauchy distribution) and $\Gamma(\cdot)$ the complete Gamma function. In the general case, we apply a scaling factor $\beta \geq 1$ for the "shape" parameter of the outlier elliptical distribution.

$$\Sigma_t^e = \beta^2 \, C_t \tag{3.16}$$

An illustration of the outlier model for "1-σ" error ellipses can be seen in Figure 3.3. A

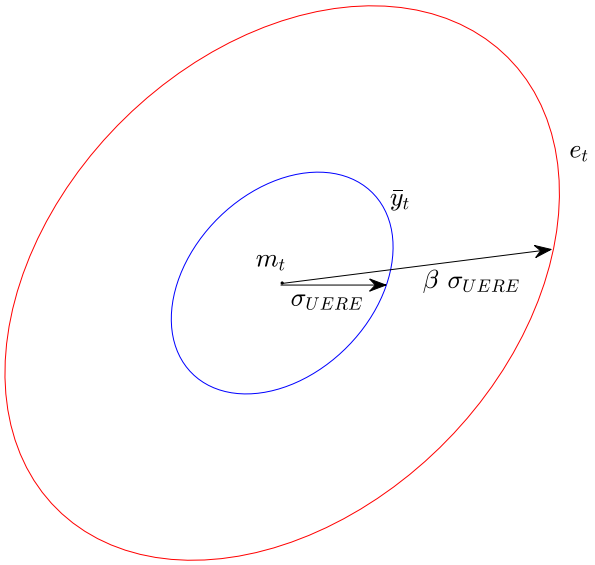


Figure 3.3: GNSS outlier error model.

comparison of different outlier models' tail behavior is provided in Figure 3.4 – as can be seen, Student or Cauchy models have much heavier tail behavior, and so provide simple and qualitatively better ways to explain "big-σ" (high Mahalanobis distance) errors that GNSS PVT solutions often exhibit in urban cores.

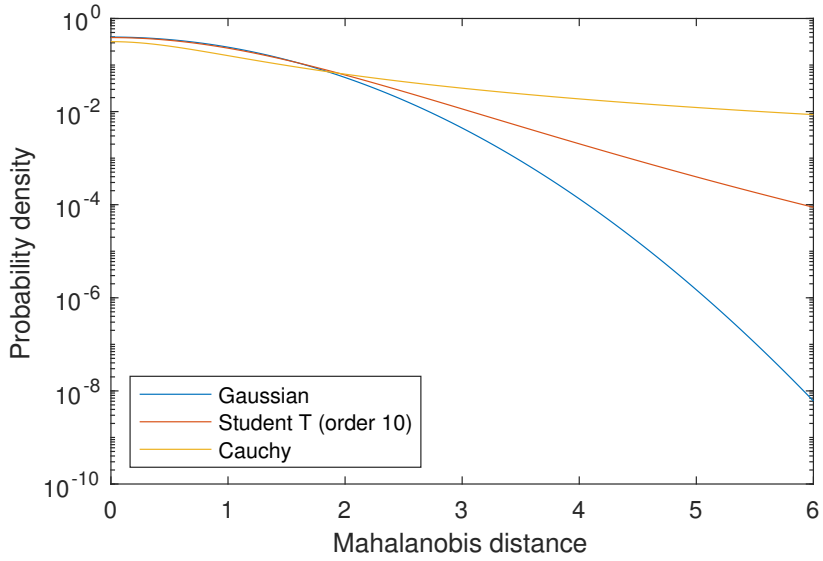


Figure 3.4: Comparison of tail behavior for different GNSS outlier models.

Optionally, for Rao-Blackwellized models, the GNSS position fix can be modeled using such a non-Gaussian mixture model, whereas the velocity component can be modeled as a separate Cartesian Gaussian observation, allowing for Kalman Filter (analytical) velocity updates (see e.g. [13]).

SNR measurement model: As in the GNSS SNR mapping algorithm, for individual SNR measurements we use a binary sensor model where the satellite is either line-of-sight (LOS) with probability P or non line-of-sight (NLOS) with probability $1-P$. Specifically the model is (as described in Section 2.2.1, and simplifying as described in Theorem 2.1)

$$\begin{aligned} p(z_{t,n}|x_t, o(m)) &\propto 1 + \left(\frac{f_{LOS}(z_{t,n})}{f_{NLOS}(z_{t,n})} - 1 \right) P \\ &= F\left(z_{t,n}, P\right) \end{aligned} \tag{3.17}$$

where f_{LOS} and f_{NLOS} are the LOS (line-of-sight) and NLOS (non-line-of-sight) likelihoods, $F(\cdot,\cdot)$ is the likelihood function defined in Equation 2.12, and

$$P = \prod_{i \in A} p(m_i = 0 | o(m)) \tag{3.18}$$

is total "emptiness" probability for all map cells $i \in A$ intersected by the ray from x_t to satellite n at time t. For the case of a particle filter (as assumed everywhere in this part of the thesis), we evaluate x_t at particle locations x_t^k, in which case $A = \mathcal{M}(t, n, k)$, as defined in Section 2.3.2.

A convenient (fewer parameters) alternative to the Rician/ Log-Normal fading model presented in Section 2.2.1 is to simply represent the likelihood ratio $\frac{f_{LOS}(\cdot)}{f_{NLOS}(\cdot)}$, which is the only SNR measurement-related quantity required in Equation 3.17. One basic option we have pursued for this purpose is a Sigmoid-type model, which involves a shifted and skewed version of the Sigmoid function $\text{sigmoid}(x) = (1+e^{-x})^{-1}$. In this model, we have:

$$\frac{f_{LOS}(r)}{f_{NLOS}(r)} = l_{\max} + \frac{l_{\max} - l_{\min}}{1 + \left(\frac{l_{\max}-1}{l_{\min}-1}\right) e^{-k(r-\Delta)}} \tag{3.19}$$

where $l_{\min}, l_{\max}$ parameters determine the dynamic range, k determines shape, and Δ is the unity-likelihood (SNR-offset) parameter. An illustration of this model in the log-likelihood domain is shown in Figure. 3.5. One added benefit of this model is that the LOS likelihood is guaranteed to be monotonically increasing, which was found to be useful for robust Shadow Matching. Important details about the runtime configuration of this SNR model are provided in Section 3.3.4.

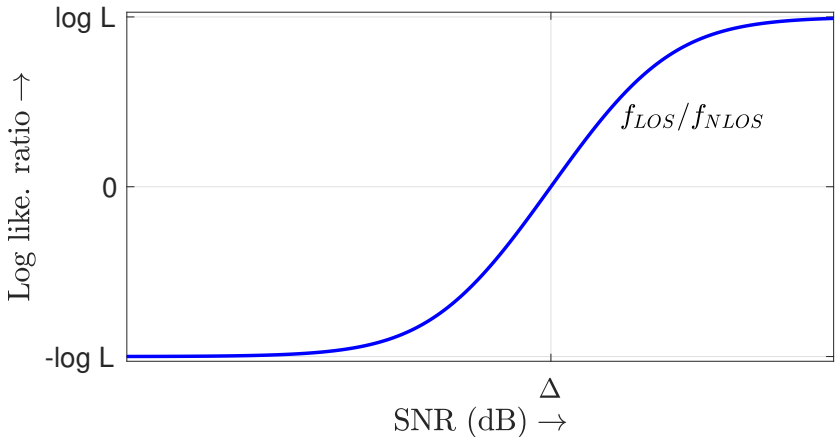


Figure 3.5: The LOS/NLOS satellite channels can be modeled as their likelihood ratio only, using a simple Sigmoid model (here with $l_{\max} = 1/l_{\min} = L$).

3.2.2 Particle filters

Both particle filters presented are backed by the models outlined above. The major differences between them is the sampling and weighting step: how do we propose new particle locations and how do we evaluate their weights?

Bootstrap PF

The bootstrap PF is a simple PF implementation where the particle proposal distribution for the kth particle, $q(x_t|x_{t-1}^k)$ is taken to be the motion-predicted distribution. The complete bootstrap PF procedure for probabilistic Shadow Matching is provided in Algorithm 4.

InitializeFromGNSSFix, SNRMeasurementUpdate: For the initial measurement update, the previous particle set does not yet exist. In this case we initialize by sampling uniformly from the GNSS fix as $x_1^k \sim p(y_t|x_t)$, and applying a shadow matching

Algorithm 4 Bootstrap PF for Probabilistic Shadow Matching

input $(\mathcal{X}_{1:t-1}, y_t, z_t)$

if $\mathcal{X}_{1:t-1} = \emptyset$ **then**

  $\mathcal{X}_{1:t} = \textbf{InitializeGNSSFix}(y_t)$

  $\mathcal{X}_{1:t} = \textbf{SNRMeasurementUpdate}(\hat{\mathcal{X}}_{1:t}, z_t)$

else

  $\hat{\mathcal{X}}_{1:t} = \textbf{MotionPredict}(\mathcal{X}_{1:t-1})$

  $\mathcal{X}_{1:t} = \textbf{MeasurementUpdate}(\hat{\mathcal{X}}_{1:t}, y_t, z_t)$

end if

if $\hat{K}_t(\mathcal{X}_{1:t}) < K_{\min}$ **then**

  $\mathcal{X}_{1:t} = \textbf{Resample}(\mathcal{X}_{1:t})$

end if

return Updated particle set $\mathcal{X}_{1:t}$

update to weigh the initial set of particles for $t = 1$:

$$w_1^k \propto p(z_{t,n}|x_t, o(m)) = \prod_n p(z_{t,n}|x_t, o(m)) \tag{3.20}$$

MotionPredict: The proposal distribution provides the motion-predicted particle set $\hat{\mathcal{X}}_{1:t}$, which under the nominal linear Gaussian model leads to

$$\begin{aligned} x_t^k \sim q(x_t|x_{t-1}^k) &= p(x_t|x_{t-1}^k) \\ &= N(x_t|\Phi_t x_{t-1}^k,\ \Phi_t \Lambda_{t-1} \Phi_t^T + Q_t) \end{aligned} \tag{3.21}$$

At this point, the particle set is "Rao-Blackwellized" so that each particle is actually represented by a Gaussian kernel.

MeasurementUpdate: So that we can evaluate particle weights, the position coordinates of $\hat{\mathcal{X}}_{1:t}$ are then sampled, with one updating the non-position coordinates using standard conditional Gaussian equations. The weight update is then

$$\begin{aligned} w_t^k \propto\ & w_{t-1}^k\ p(z_t, y_t|x_t^k, o(m)) \\ =\ & w_{t-1}^k\ p(y_t|x_t^k)\ p(z_{t,n}|x_t, o(m)) \\ =\ & w_{t-1}^k\ p(y_t|x_t^k)\ \prod_n p(z_{t,n}|x_t, o(m)) \end{aligned} \tag{3.22}$$

where we have substituted Equation 3.10, which contains the effects of the previous particle weights, GNSS fix matching, and probabilistic Shadow Matching.

Resample: *After* computing the output metrics (before would lead to information loss), for example the Minimum Mean Squared Error (MMSE) point estimate

$$\hat{x}_t = E(x_t) = \sum_k w_t^k x_t^k \tag{3.23}$$

and uncertainty (typically the radius around $\hat{x}_t$ that captures 68% of the particle mass), one can then resample the particles using standard resampling techniques in order to avoid particle collapse. In practice we do that if the effective sample size , as defined in [4]

$$\hat{K}_t = \left(\sum_k \left(w_t^k \right)^2 \right)^{-1} \tag{3.24}$$

is below a threshold $K_{\min}$.

Advanced PF

The bootstrap PF, while simple to implement, has the disadvantage that it isn't adventurous enough: particles are drawn from the motion model, which can lead to trapping in local maxima of posterior distribution, which, as it turns out, there are plenty of these in deep urban environments. To overcome this one can instead attempt to sample from the locally optimal (in terms of minimizing the variance of the weights) proposal distribution, that takes into account the current measurements as well.

$$x_t^k \sim q(x_t | x_{t-1}^k, y_t, z_t) = \frac{1}{Z}\, p(x_t | x_{t-1}^k)\; p(z_t, y_t | x_t) \tag{3.25}$$

where Z is normalization constant.

Unfortunately, there is no analytical expression for that quantity. However, by using a ray tracing cache (discussed in a following section), one can efficiently come up with a Kernel density estimate (KDE) (or Gaussian mixture) for the proposal distribution by first computing a kernelized estimate of the measurement likelihood surface. The full specification for this Advanced PF is detailed in Algorithm 5.

Algorithm 5 Advanced PF for Probabilistic Shadow Matching

input $(\mathcal{X}_{1:t-1}, y_t, z_t)$

if $\mathcal{X}_{1:t-1} \neq \emptyset$ **then**

  $\hat{\mathcal{X}}_{1:t} = \textbf{MotionPredict}(\mathcal{X}_{1:t-1})$

end if

$\mathcal{L}_t = \textbf{GenerateLikelihoodSurface}(\hat{\mathcal{X}}_{1:t}, y_t, z_t)$

if $\mathcal{X}_{1:t-1} = \emptyset$ **then**

  $\mathcal{X}_{1:t} = \textbf{InitializeOptimal}(\mathcal{L}_t)$

else

  $\mathcal{X}_{1:t} = \textbf{SampleOptimal}(\hat{\mathcal{X}}_{1:t}, \mathcal{L}_t)$

end if

if $\hat{K}_t(\mathcal{X}_{1:t}) < K_{\min}$ **then**

  $\mathcal{X}_{1:t} = \textbf{Resample}(\mathcal{X}_{1:t})$

end if

return Updated particle set $\mathcal{X}_{1:t}$

MotionPredict: Forward prediction of the particle set using the motion model is identical to that of Bootstrap PF (Equation 3.21).

GenerateLikelihoodSurface: The Advanced PF requires generating (estimating) the Shadow Matching likelihood surface, with support in the 3D (XYZ) position space:

$$\mathcal{L}_t \triangleq p(z_t, y_t|x_t) \approx \sum_{i=1}^{M} \rho_t^i N(x_t|\mu_t^i,\ \Sigma) \tag{3.26}$$

and kernel weights

$$\rho_t^i = p(y_t|\mu_t^i) \prod_n p(z_{t,n}|\mu_t^i) \tag{3.27}$$

which for simplicity can have circular bandwidths, $\Sigma = \sigma^2 I_3$. In practice, to cover space efficiently (to allow us to cover large areas) the kernel centers $\{\mu_t^i\}_{i=1}^M$ are generated on a regular grid (e.g., Cartesian grid or a dense sphere packing lattice, such the Face Centered Cubic lattice [8]) with inter kernel distances of around a meter or two. For 2D ground level motion, most of the surface can be built up around ground level. Obviously, getting the likelihood surface support right is crucial; for now, we attempt to do this by taking kernel centers inside the union of the S-σ ellipses (or ellipsoids) around the GNSS fix y_t and motion predicted particle sets $\hat{\mathcal{X}}_{1:t}$, as depicted in Figure 3.6.

InitializeOptimal: For the initialization step of the Advanced PF, we simply sample K equally weighted particles from the Shadow Matching likelihood surface at time $t = 1$, denoted LS_1. Note that this methodology is an "optimal" initialization in that it minimizes the variance in the initial particle weights to *zero*.

SampleOptimal: For non-initialization updates, we have access to a predicted particle set, very similar to that which would be produced in the Bootstrap PF. However,

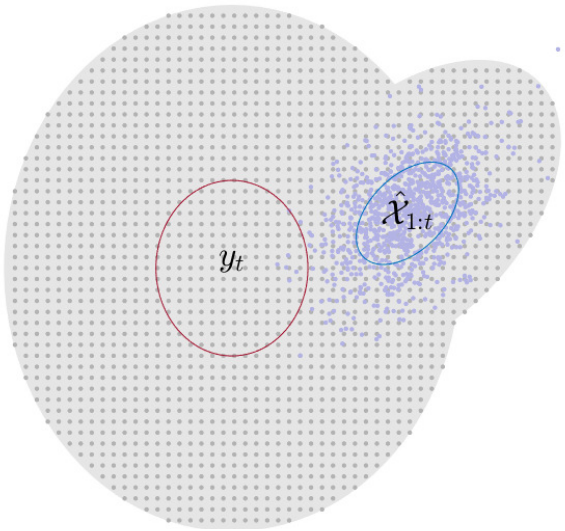


Figure 3.6: Illustration of the likelihood surface region (gray) and support (grid points), in the case where GNSS fix y_t and predicted particle set $\hat{\mathcal{X}}_{1:t}$ disagree heavily with each other.

we also have access to the full Shadow Matching likelihood surface $\mathcal{L}_t$. Therefore, with the nominal linear Gaussian motion model, the proposal density is approximately

$$q(x_t|x_{t-1}^k, y_t, z_t) \approx \frac{1}{Z} N(x_t|\Phi_t x_{t-1}^k,\ \Phi_t \Lambda_{t-1} \Phi_t^T + Q_t) \sum_{i=1}^{M} \rho_t^i N(x_t|\mu_t^i, \Sigma). \tag{3.28}$$

Since products of Gaussians are themselves Gaussians, sampling from this distribution for each particle k then boils down to Rao-Blackwellized sampling from a Gaussian mixture, which is easy. However, it can be slow: this approach scales as $O(KM)$ one must compute the M (typically 5000-10000) component likelihood surface ($O(M)$ step) for each of K (typically 2000-5000) particles. While this can be trivially parallelized, that only gets you so far.

The solution is to realize that for clusters of nearby particles, the vast proportion of the proposal KDE's weights are very small, chiefly due to motion constraints. By

using KD tree clustering on the particles and box-and-bound techniques (see [27] for an inspiration), one can upper bound these weights by a small number (say 10^{-8}) for a given cluster of particles, and then prune those kernels from *that cluster's copy* of the proposal KDE. One can then parallelize across clusters. Typically this leads to a 70-95% reduction in complexity, depending on in-cluster particle spread, volume of the KDE support, firmness of motion constraints, etc. (By using dual tree techniques as in [27], i.e. creating KD trees for both the particles and likelihood surface, one can probably do even better.)

As for particle weighting, referring to [4] the particle weight update for the "optimal" PF is

$$w_t^k \propto w_{t-1}^k \int p(x_t|x_{t-1}^k)\ p(y_t, z_t|x_t)\ dx_t \tag{3.29}$$

which evaluates (approximately) to the sum of the weights of the Gaussian mixture for $q(\cdot)$ in Equation 3.28.

3.2.3 Particle smoothing

For offline localization purposes, one is often most interested in the best (highest) accuracy estimate of the trajectory of a user given a *batch* of measurements. Particle smoothing (PS) estimates "smoothed" positions at time t given past and future GNSS fix and SNR measurements, $y_{1:T} = \{y_1, ..., y_t..., y_T\}$ and $z_{1:T} = \{z_1, ..., z_t..., z_T\}$. Under the first order Markov assumption of the motion model (i.e., $p(x_t|x_1, \ldots, x_{t-1}) = p(x_t|x_{t-1})$),

the smoothed estimate at time t becomes

$$p(x_t|y_{1:T}, z_{1:T}) = \underbrace{p(x_t|y_{1:t-1}, z_{1:t-1})}_{\text{forward predicted}} \times \underbrace{p(x_t|y_{t:T}, z_{t:T})}_{\text{backward filtered}}, \tag{3.30}$$

which represents the product of the forward predicted and backward filtered solutions at time t (note we have dropped the dependence on the occupancy map $o(m)$ for readability).

In our implementation of PS for Probabilistic Shadow Matching, we use the "two filter" strategy described in [35], which leverages kernel density estimation (KDE) and branch and bound techniques to accelerate the process of approximating the product operation over two particle sets, as in Equation 3.31. The backward filter is generated by simply running the PF in reverse, with reverse-ordered measurements and a time-reversed motion model.

3.3 Practical implementation

In this section, we go over a few of the primary implementation details for our Probabilistic Shadow Matching framework. We choose to cover these as some of them proved necessary in order to accelerate the costly ray tracing-based Shadow Matching algorithm, as well as make it more robust to GNSS positioning error in urban environments.

3.3.1 Robust particle sampling

For the Advanced PF, we note a few additional optional steps (optimizations) namely for the sampling and likelihood surface generation steps.

Primal-Dual sampling

In areas of high position uncertainty (such as downtown urban cores where multipath reflections and GNSS PVT error are severe), empirically we have found that drawing a fraction of particles from the likelihood surface, and weighing by the motion model, provides extra robustness. The intuition for this is that – even though as it theoretically reduces optimality of the sampling step – it allows us to generate particles further away from the motion model prediction. Thus it improves particle diversity spatially, which naturally guards against divergence. To implement this, we follow the primal-dual sampling step as outlined in [56] with a total $K_d = \rho_{\mathrm{d}} K$ particles dually sampled from $\mathcal{L}_t$ at each measurement update. Note that, in this case, maintaining normalization factors for the likelihood surface and sampling steps are crucial – making implementation a bit tricky. Substituting Equation 3.21, the dual sampled particles are generated as $x_t^k \sim \mathcal{L}_t$ with weights

$$\begin{aligned} w_t^k &\propto p(x_t^k | x_{t-1}) \\ &= \sum_k w_{t-1}^k \; N(x_t^k | \Phi_t x_{t-1}^k, \; \Phi_t \Lambda_{t-1} \Phi_t^T + Q_t) \end{aligned} \tag{3.31}$$

which can be evaluated approximately and efficiently using KDE evaluation techniques provided in [27]. The entire primal-dual sampling approach is provided in Algorithm 6.

Particle resetting

Beyond primal-dual sampling, an additional feature of the Advanced PF is that since we have a KDE for the measurement likelihood surface, for improved robustness (as

Algorithm 6 Primal-dual particle sampling for the Advanced PF

1: input $(\hat{\mathcal{X}}_{1:t}, \mathcal{L}_t, K, \rho_d))$

2: $\mathcal{X}^p_{1:t} = \textbf{SampleOptimal}(\hat{\mathcal{X}}_{1:t}, \mathcal{L}_t)$ for $K_p = (1 - \rho_d)K$ particles

3: $\mathcal{X}^d_{1:t} = \textbf{DualSample}(\hat{\mathcal{X}}_{1:t}, \mathcal{L}_t)$ for $K_d = \rho_d K$ particles

4: **return** Combined particle set $\mathcal{X}^{pd}_{1:t} = \mathcal{X}^p_{1:t} \bigcup X^{\mathrm{d}}_{1:t}$

a heuristic) one can "sensor reset" [56] a proportion of the particles at each iteration by sampling some fraction directly from the measurement surface (ignoring the motion model). The added robustness comes from the fact that the the localization space is often highly multi-modal and this strongly encourages the PF to explore it; although resetting too much leads to jagged paths and exaggerated estimation uncertainty. To implement this, we simply sample $\rho_r K$ particles from $\mathcal{L}_t$ and combine in with primal-dual sampled particles. The particle reset sampling approach is provided in Algorithm 7.

Algorithm 7 Sensor reset particle sampling for the Advanced PF

1: input $(\hat{\mathcal{X}}_{1:t}, \mathcal{L}_t, K, \rho_r)$

2: $\mathcal{X}^{pd}_{1:t} = \textbf{PrimalDualSample}(\hat{\mathcal{X}}_{1:t}, \mathcal{L}_t)$ for $K_{pd} = (1 - \rho_{\mathrm{r}})K$ particles

3: $\mathcal{X}^r_{1:t} = \textbf{Initialize}(\mathcal{L}_t)$ for $K_r = \rho_r K$ particles

4: **return** Combined particle set $\mathcal{X}_{1:t} = \mathcal{X}^{pd}_{1:t} \bigcup \mathcal{X}^r_{1:t}$

Rao-Blackwellization

For the Advanced PF, particles are sampled from a Gaussian KDE representing the likelihood surface $\mathcal{L}_t$. Because of this, a very convenient feature is that with the Advanced

PF Rao-Blackwellization [13] can be fully built into the Advanced PF. That is, position *and* velocity coordinates can both be represented analytically as Gaussians, so that each particle essentially maps to its own miniature Kalman Filter in velocity space. Note, however, that for certain non-linear transportation models not considered here, e.g. the constant turn or curvilinear, polar velocity models detailed in [38], we would need the particles to be *singular* (non-Rao-Blackwellized) in all except the velocity coordinates – in order for a single covariance matrix to apply to each particle, which is very desirable computational (memory and runtime) complexity reasons.

3.3.2 Efficient 3D map representation and ray tracing

Octree ray tracing

Naively, occupancy grid maps are represented at a single, fine resolution (down to roughly 1m for the case of GNSS SM[3]). However, this is extremely wasteful – real-world 3D environments are highly spatially correlated. That is, for uniform OG grids, most cells contain roughly the same occupancy probabilities as their neighbors.

One approach to reducing the size of the OG map in memory, as well as to accelerate common operations through OGs (such as ray tracing), is to use an octree representation, as illustrated in Figure 3.7). Essentially a binary tree extended to three dimensions, an octree is defined as a cubic, hierarchical partitioning of space whereby non-leaf nodes in the tree are recursively subdivided into eight, equally-sized children nodes, until a target

[3]1m is roughly interaction size of typical GNSS signals at around 1 GHz frequency

resolution (level of detail) is met. The benefit is that, with an octree OG representation,

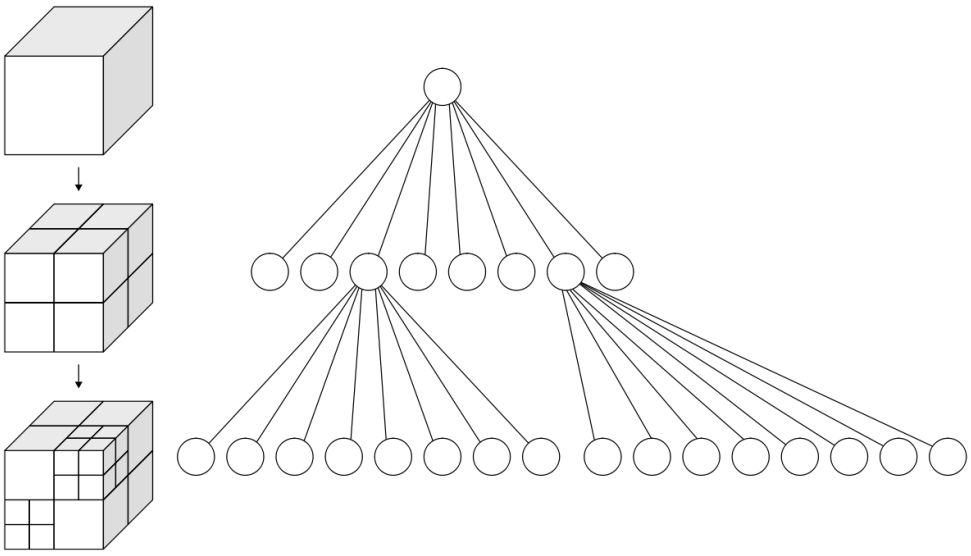


Figure 3.7: Illustration of an octree (creative commons license).

large regions of empty and occupied space are represented with a relatively small number of map cells.

An example horizontal slice of an octree for a OG map of UCSB is shown in Figure 3.8. As a popular form of map representation, various high performance open source octree frameworks are available, such as Octomap [25]. In addition, fast, easily implementable, and parallelizable octree ray tracing algorithms are readily available. One option is [47], which as a runtime optimization can be modified[4] to early terminate ray casting once a threshold blockage probability is achieved.

[4]Since only blockage probabilities are required, instead of entire lists of cells intersected

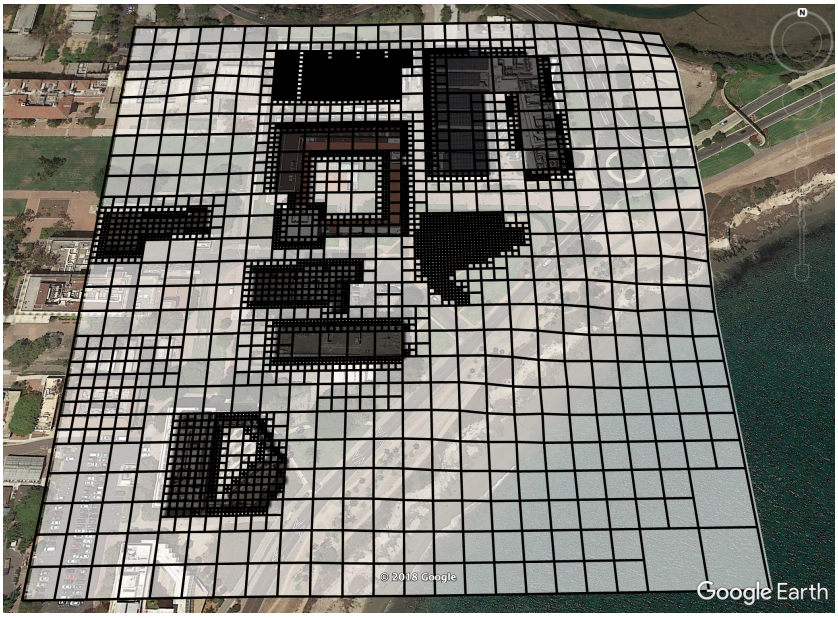


Figure 3.8: Example quadtree cross section of an OSM-based octree OG map for the northeast corner of UC Santa Barbara.

Ray tracing cache

An important feature of GNSS satellites is that they move relatively slowly through the sky, owing to their designed medium earth orbit (MEO) altitudes. In addition, a common feature of GNSS receiver motion (e.g. pedestrians, cars) is that – in areas where Shadow Matching provides the most benefit (i.e., built up urban cores) – they move at slow speeds relative to the size of the *large* search space (likelihood surface). Hence, for a single receiver, the same satellites will be casted to from the same locations in space, across multiple successive measurement epochs.

With the Advanced PF (Section 3.2.2), much of this duplication of work is made unnecessary. In this thesis, this is accomplished by using a Least Recently Used (LRU)

cache for ray tracing (i.e., a first-in, first-out look-up table), so that costly ray tracing is only performed when necessary.

To make the cache useful, we impose two conditions that fix rays to a finite search space:

1. The coordinate for ray origins (likelihood surface kernel center points) are quantized to a deterministic set of locations. This is made possible by using a grid, or lattice structure on the *earth* when building the likelihood surface.

2. Ray destinations (satellites) are quantized to a deterministic set of (azimuth, elevation) coordinates. This is made possible by using a grid in the *sky*, or lattice structure, to build the likelihood surface.

To maximize spatial efficiency for the "*earth* grid" (minimizing the number of required kernels M), it was earlier proposed to use a regular Cartesian grid or dense sphere packing lattice, such as the "honeycomb" pattern [8] (Figure 3.9). For the "*sky* grid", various

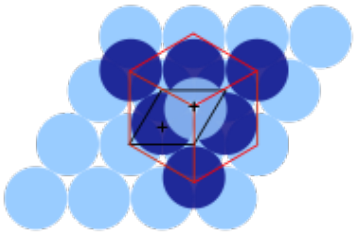


Figure 3.9: The tightly-packed Face-Centered Cubic (FCC) lattice is an ideal representation of the likelihood surface "earth grid" for the ray tracing cache (creative commons license).

regular recursive grids can be used: for example, the geodesic polyhedron (also known

as the geodesic dome, shown in Figure 3.10) [9], or the geographic hashing of the unit sphere called Schneider 2 (S2) cells [48]. A "Ray tracing cache" can then be utilized to look up blockage probabilities for different rays (a process that succeeds on a cache "hit"), before conducting costly ray tracing and updating the cache (on a cache "miss"). In common scenarios where the likelihood surface is rather large (due to large amounts

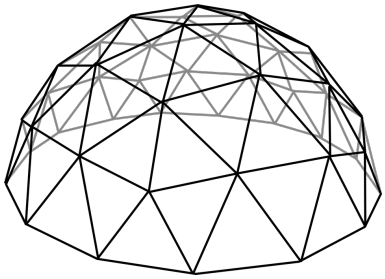


Figure 3.10: Vertices of the geodesic dome form a natural set of locations for the "sky grid" for the ray tracing cache (creative commons license).

of positioning uncertainty), and where the receiver is moving slowly, cache hit rates can be as high as 99% or greater.

Design trade-offs: Perhaps the easiest way to maximize cache efficiency (so called "hit rate" as well as its memory footprint) is to decrease the resolution of the ray tracing grids, thereby reducing the space of possible rays that could be queried in the cache. However, this coarsening comes at a positioning accuracy penalty.

For our application, because the octree is already discretized, an earth grid spacing of roughly the octree resolution (1-2m) is appropriate. For ray directions (the sky grid), because satellites move relatively slowly through the sky, quantization of less than a degree (e.g. subdivision level 6 in the geodesic dome) to a couple degrees (subdivision

level 4) can be used, depending on the ray tracing accuracy required. The exact set of trade-offs are provided in Table 3.3.2[5], which relates the sky grid resolution to the cache size in number of points, average and maximum quantization error in degrees, and average cache invalidation time (time-to-live, or TLL) in seconds. Note that the memory footprint of the cache scales with the number of points, and the average TTL essentially represents the expected time window over which the cache is useful for ray tracing to a particular satellite.

Level	**Size**	**Avg Err. (°)**	**Max Err. (°)**	**Avg TTL (s)**
2	89	6.11	10.3	1.48×10^3
3	337	3.08	5.22	813
4	1.31×10^3	1.50	2.65	410
5	5.19×10^3	0.757	1.35	209
6	2.06×10^4	0.383	0.678	103
7	8.22×10^4	0.189	0.337	53.6
8	3.28×10^5	0.0954	0.167	26.4
9	1.31×10^6	0.0475	0.0829	13.4

Table 3.1: Design trade-offs between ray tracing cache resolution, size, satellite coordinate quantization error, and cache time-to-live (TTL).

3.3.3 Fast particle smoothing

For the Advanced PF, to further accelerate smoothing (as described in Section 3.30), likelihood surfaces $\mathcal{L}_t$ estimated in the forward filtering pass can be saved and reused in the backward pass. This approximation can significantly speed up the backward filtering pass by avoiding duplicating costly ray tracing, at the cost of a an increased memory

[5]For GPS and GLONASS satellite trajectories at San Francisco, CA latitude

footprint and a possible reduction in positioning accuracy (as the backward pass can sometimes provide better likelihood surface support).

3.3.4 SNR model

For the Sigmoid SNR model defined in Section 3.2.1, the setting of the unity likelihood SNR value Δ is particularly important, and different approaches can followed. For example, one can try to learn these parameter values in an offline fashion as in Section 2.4, or one can instead use an adaptive, online estimation scheme. Per satellite values can be estimated, or a single offset parameter can be shared among all satellites.

In this work, for the real-time PF scenario, we propose a simple adaptive scheme, that is robust to subtle changes in GNSS receiver handling, body shadowing, and satellite elevation. Specifically, we estimate it recursively and on a per satellite basis:

$$\Delta_{t,n} = \max\left(z_{t,n} - \delta,\ \Delta_{\min},\ \Delta_{t-1,n}\ e^{-\frac{T}{\tau}}\right), \tag{3.32}$$

where T is the time step from epoch $t-1$ to t, τ is the filter time constant, $\Delta_{\min}$ is minimum (initial) value, δ is the offset. A typical configuration for a smartphone with a low gain GNSS antenna would be $\tau = 5$ minutes, $\Delta_{\min} = 25$ dB, $\delta = 10$ dB.

The SNR model used for the offline (smoothing) use case can be two sided in the sense that future SNR data can be used to estimate Δ. A simple modification to Equation 3.32 that we use is to calculate two versions of $\Delta_{t,n}$, iterating over the SNR data in forward

and reverse directions, and taking a maximum:

$$\begin{aligned}
\Delta_{t,n}^{\text{fwd}} &= \max\left(z_{t,n} - \delta,\ \Delta_{\min},\ \Delta_{t-1,n}^{\text{fwd}}\ e^{-\frac{|T|}{\tau}}\right), \\
\Delta_{t,n}^{\text{bwd}} &= \max\left(z_{t,n} - \delta,\ \Delta_{\min},\ \Delta_{t+1,n}^{\text{bwd}}\ e^{-\frac{|T|}{\tau}}\right), \\
\Delta_{t,n} &= \max\left(\Delta_{t,n}^{\text{fwd}}, \Delta_{t,n}^{\text{bwd}}\right).
\end{aligned} \tag{3.33}$$

3.3.5 Architecture

Much about the software architecture of a practical Probabilistic Shadow Matching system is beyond the scope of this thesis, but certain attributes of the architecture remain relevant. For example, in our demonstration of the prototype system [28], localization is carried out on a real-time particle filter (PF) server, which houses the 3D occupancy grid maps, performs cached ray tracing, and runs a PF responsible for estimating the overall location of the device. In addition, the 3D mapping component (Chapter 2) is performed on the backend as well, supplies offline generated maps to the localization module. A mobile application running on the GNSS device reports the necessary GNSS measurement data over the cellular network in the uplink, and consumes corrected locations in the downlink at a latency. A high-level system diagram of the localization system is provided in Figure 3.11.

3.4 Localization Experiments

Our localization algorithms were validated in small scale, pedestrian tests around the University of California, Santa Barbara (UCSB) and downtown Santa Barbara. For these

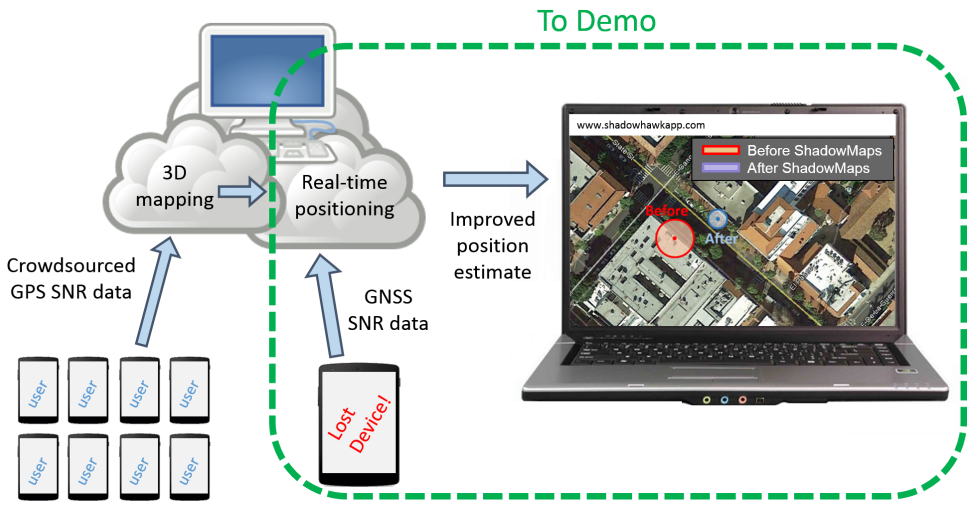


Figure 3.11: Proposed and demonstrated architecture of the localization system.

experiments, the GNSS multipath conditions experienced were relatively minor, and the PF algorithm used relatively simple, and we show qualitative only results.

A larger scale experiment was then completed in San Francisco, again for the pedestrian transportation model. For this experiment, more scalable and robust PF algorithms were employed, as was higher quality 3D mapping data sourced from public aerial Lidar surveys. Forward-backward particle smoothed outputs, in addition to the normal real-time PF outputs, were also computed. In addition, more comprehensive positioning metrics are provided for this set of urban experiments.

Note that in all experimental results, we use the Minimum Mean Squared Error (MMSE) point estimates for the output particle sets, as defined in Equation 3.23.

3.4.1 UCSB and Santa Barbara localization experiments

Initial small-scale validation of the Probabilistic Shadow Matching framework was performed with two small experiments, one at UCSB and one in (slightly more built up) downtown Santa Barbara. Example results can be seen Figures 3.4.1 and 3.4.1, which show Google Maps aerial views of positioning improvement for two different mobile devices on the UCSB campus and in downtown Santa Barbara. Note that the improved positioning track is significantly closer to the ground truth path, whereas the raw device positioning exhibits significant cross-track distance error (deviation from the ground true path), even in these low rise environments, presumably due to multipath from nearby buildings.

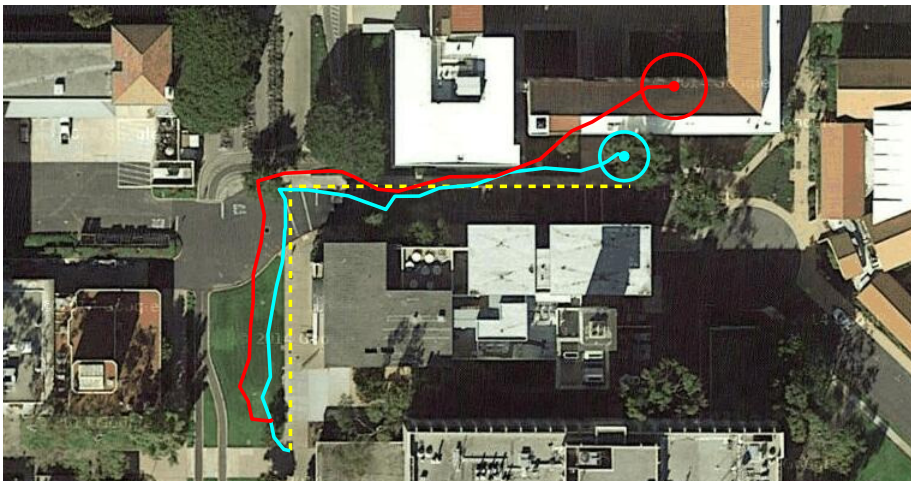


Figure 3.12: Positioning improvement at UCSB: true path in yellow, standalone GNSS output in red, and corrected path in light blue.

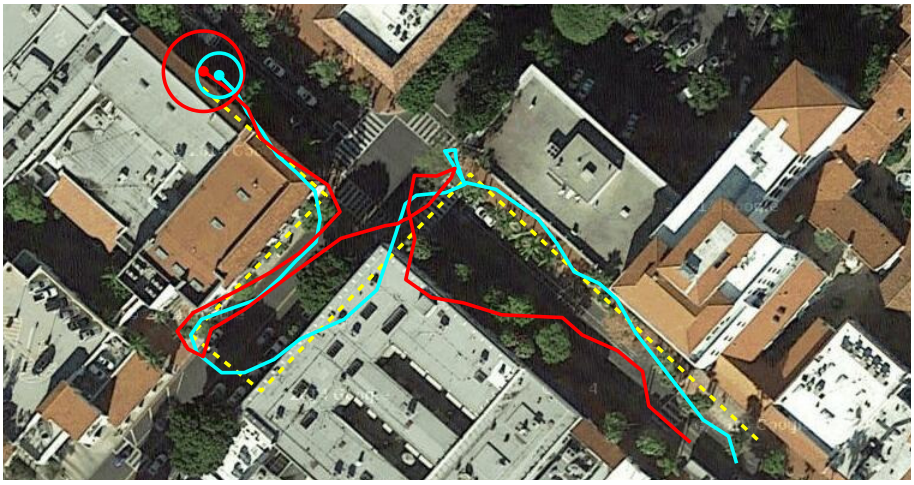


Figure 3.13: Positioning improvement in downtown Santa Barbara: true path in yellow, Android Fused Location Provider (FLP) output in red, and corrected path in light blue.

3D maps

The maps used to enable this location improvement at UCSB are presented in Sections 2.5.2 and 2.5.3 – themselves generated using GNSS SNR measurements. Note that these maps are uniform occupancy grids (not in octree representation). In the UCSB experiment, a Samsung Galaxy Tab 2 was used with only GPS+GLONASS enabled to arrive at the original position fixes, whereas in downtown Santa Barbara a Motorola Moto X smartphone using the Android Fused Location provider was used (which generally fuses GNSS with WiFi- and cellular network-based positioning).

Algorithm settings

The algorithm used to generate these results was our initial Bootstrap PF (Algorithm 4). Motion was restricted entirely to 1.5m above ground level (in our maps, the zero-altitude plane). A four state (2D position-velocity) NCV motion model was used with

$\sigma^2_{\dot{x},\dot{y}} = 2$. $K = 5000$ particles were employed in the Bootstrap PF, as was an identical Rician/Log-Normal SNR modeling approach to the one described in Section 2.5. Finally, no GNSS outlier model was utilized (that is, equation 3.11 was applied with outlier probability $\alpha = 0$).

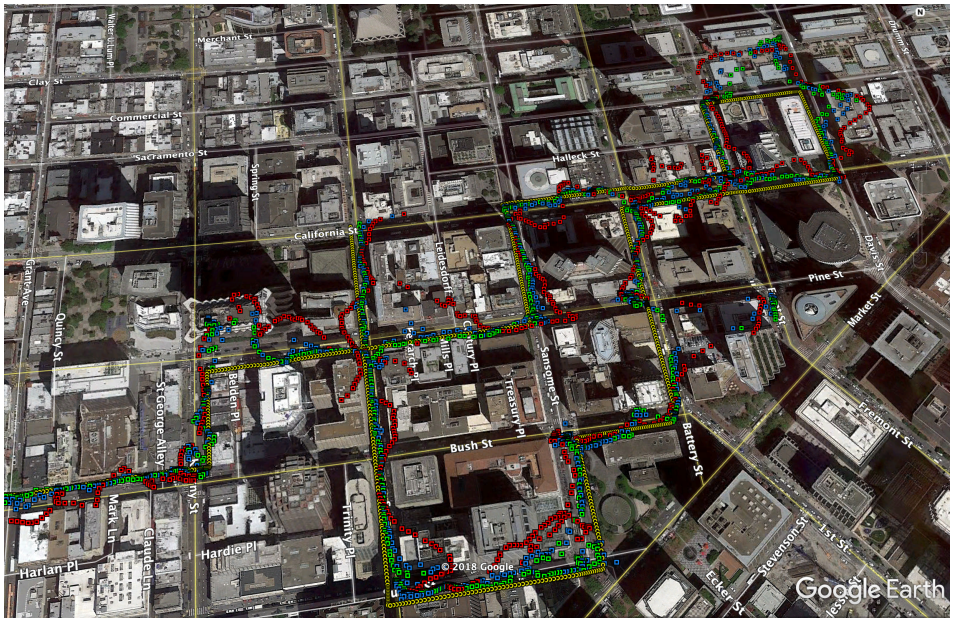
3.4.2 San Francisco localization experiment

A longer pedestrian experiment was conducted in San Francisco. The data collection window was a little over one hour (63 minutes), and the path covered both the high rise Financial District and medium rise Tenderloin and Union Square neighborhoods. For this data collection, only the Android Fused Location Provider (FLP) for a Nexus 5X Android was used as the reference "raw" locations.

Screen shots of Google Earth visualizations of the two main portions of the experiment are shown in Figures 3.14(a) and 3.14(b). From these figures it can be gleaned that positioning error is reduced with Probabilistic SM in the urban environment (however many errors are left unresolved – a feature that is discussed in the concluding remarks in Section 4.2).

Ground truth

Time-stamped ground truth location and velocity was determined by interpolating "pins" dropped (via long press) against Google Maps satellite imagery in a custom Android application. These pins were placed every time the user changed velocity (speed or



(a) Financial district (high rise)

(b) Tenderloin/ Union Square (medium rise).

Figure 3.14: Positioning improvement in San Francisco: true path (yellow), Android FLP (red), real-time PF (blue), and offline PS (green).

heading), so that simple linear interpolation could be used to estimate the approximate ground truth path.

3D map data and construction

Precise locations of the buildings and other GNSS shadowing features (such as trees) were determined using publicly available aerial Lidar data sourced from the ARRA (American Recovery and Relief Act) Golden Gate survey from 2010 [58]. Publicly available building footprints (polygons) for San Francisco were downloaded from the city government website [7], and were used to roughly classify Lidar points as buildings versus non-buildings (i.e. vegetation, etc). These were the was then used to decrease the occupancy probability in the octree represented map for non-buildings from the nominal value (0.5) to 0.1, so that for example vegetation would be predicted to not shadow GNSS satellites as heavily. Finally, as San Francisco is hilly, publicly available digital terrain maps (DTMs) were downloaded from the USGS National Map [51] and used to get ground elevation information for the PF.

Algorithm settings

For this experiment the Advanced PF was used (Algorithm 4). Motion was restricted entirely roughly 1.5m above ground level using a tight Gaussian prior ($\sigma = 0.5m$) at that height, as determined by the terrain map. A four state (2D position-velocity) NCV motion model was used with $\sigma^2_{\dot{x},\dot{y}} = 2$. $K = 2000$ particles were employed in the Advanced PF, and simple Sigmoid SNR (defined in Sections 3.2.1 and 3.3.4) model was used, with

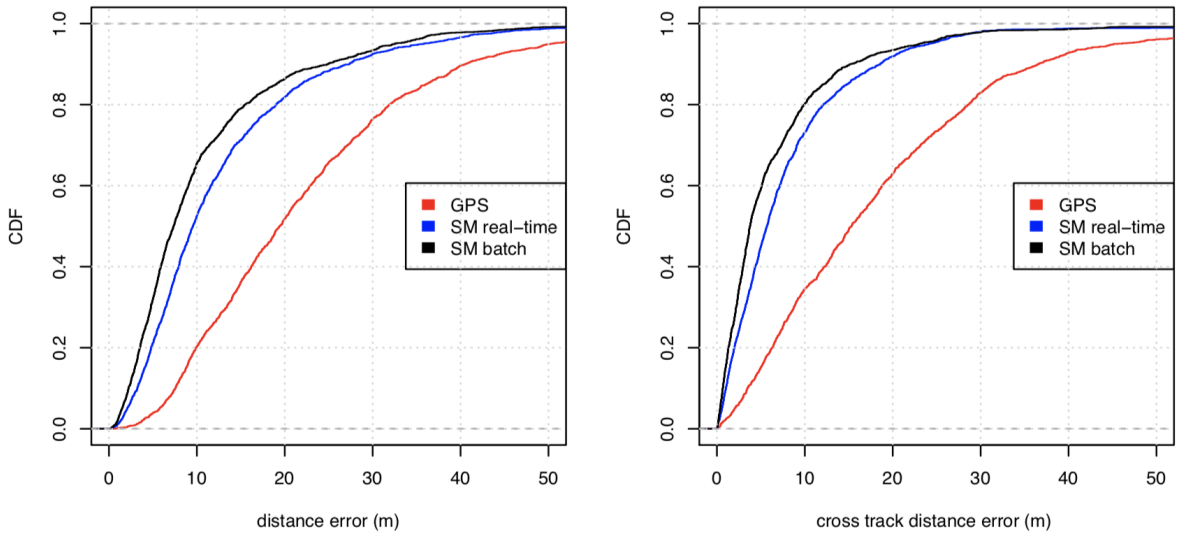
$l_{\max} = \frac{1}{l_{\min}} = 7$, $k = 0.7$, $\Delta_{\min} = 25$ dB, $\tau = 5$ minutes, and $\delta = 10$ dB. A full ray tracing cache (as defined in Section 3.3.2) was used, with 5 levels in the sky grid, and 2.2m spacing in the likelihood surface (earth) grid, which was represented using the proposed FCC lattice at ground level. The full GNSS outlier model was utilized with outlier probability $\alpha = 0.5$ and $\beta = 3$, with a Cauchy outlier distribution. To reduce the effects of GNSS position correlation, the input measurement data were subsampled from 1 Hz to 0.5 Hz (that is, one GNSS fix arriving with one set of satellite SNR measurements every two seconds).

Performance metrics

As a primary metric, distance error was calculated against the ground truth position time series. In addition, we also estimated cross track distance error, defined as distance from the local sixty second snapshot of the ground truth trajectory:

$$\text{Cross track distance error} = \min_{t_{\min} \leq \tau \leq t_{\max}} ||\hat{x}_t - \hat{x}_\tau^G|| \tag{3.34}$$

where $\hat{x}_\tau^G$ is the ground truth position estimate at epoch τ, and $t_{\min}/t_{\max}$ are min/max epochs within $\pm 30s$ of epoch t. Cumulative distributions and select percentile values are shown in Figure 3.4.2, exhibiting roughly 35-50% reduction in distance error for the real-time PF over Android FLP (GPS), and approximately 45-60% reduction in error for the offline (batch) PS algorithm. It can be seen that relative error reduction in the cross track (cross street) direction is slightly greater. This is intuitively due to the fact that GPS errors are typically in the cross street direction (due to multipath reflections adding range

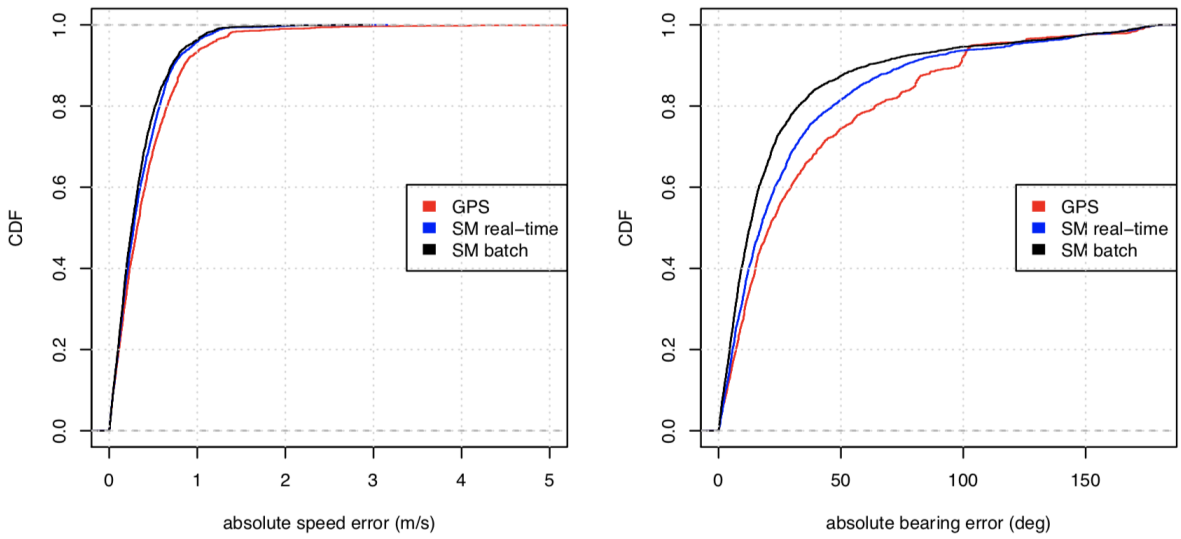


	GPS	SM real–time	SM batch
25%	11.5	5.63	4.16
50%	19.4	9.56	7.35
67%	25.7	13.5	10.4
90%	40.5	27.3	24.9
95%	50.2	35.8	32.5
98%	63.8	45	42
mean	22.5	13.2	11.1

	GPS	SM real–time	SM batch
25%	7.5	2.58	1.76
50%	15.3	5.7	3.77
67%	21.9	8.57	6.5
90%	36.6	18.6	15.2
95%	45.4	24.1	23
98%	60.3	30.1	30.2
mean	18.6	8.55	7.08

Figure 3.15: Distance and cross track distance error metrics for the San Francisco localization experiment.

error for those satellites), whereas Probabilistic Shadow Matching aids in estimating the side of the street the user is on (see [23] for discussion about this phenomenon). Velocity



	GPS	SM real–time	SM batch
25%	0.141	0.129	0.123
50%	0.317	0.278	0.256
67%	0.476	0.419	0.38
90%	0.849	0.729	0.707
95%	1.12	0.953	0.904
98%	1.38	1.18	1.11
mean	0.421	0.355	0.332

	GPS	SM real–time	SM batch
25%	8.87	7.04	5.73
50%	20.8	17.3	12.7
67%	38	28.7	20.8
90%	98.3	75.6	60.9
95%	104	120	110
98%	167	160	157
mean	36.2	30.5	24.8

Figure 3.16: Speed and bearing (heading) error metrics for the San Francisco localization experiment.

error, calculated against the ground truth velocity time series, is broken down in to speed and bearing (heading) error distributions in Figure 3.4.2. As can be seen, heading error is improved in the higher percentiles (40-90%) by up to half, which is intuitive – if side

of street error is reduced, the heading should be snapped better to the sidewalk the user is on. Speed error is only very slightly improved for both real-time and batch-mode processing.

With the ray tracing cache enabled, overall processing time per measurement update (every 2 seconds) averaged roughly 150 milliseconds for the Particle Filter. Additional metrics, including those representing forward (along track) position and velocity error, as well as UI (User Interface) "jumpiness" metrics, are defined and presented and discussed in Appendix B.1.

Chapter 4

Conclusion

In this dissertation, we give two examples of how wireless GNSS signals can be used in novel ways, for practical positioning and ubiquitous mapping using satellite SNR measurements.

4.1 Mapping using GNSS SNR measurements

For three-dimensional (3D) map building using GNSS SNR measurements, we have presented a *principled Bayesian* technique that employs physically motivated probabilistic models for the mapping problem and its components. The proposed approach provides a novel way to incorporate a large set (hundreds of thousands) of satellite measurements that can be crowd-sourced from GNSS users. We have shown how a binary line-of-sight (LOS) versus non-line-of-sight (NLOS) measurement model for the satellite signal strengths naturally simplifies a graph based optimization routine for estimating the map (Belief Propagation, BP) from exponential to linear complexity in the number of measurements.

We presented two mapping algorithms: a simple one optimized for scenarios where the location of the receiver is well known, and a slightly more complex but more accurate, Simultaneous Localization and Mapping (SLAM) algorithm that is more appropriate when the receiver position is only roughly available. Practical extensions to the mapping problem, such as fusing in prior map estimates, were discussed. We presented experimental results for small-scale mapping experiments at the University of California, Santa Barbara (UCSB) campus, as well as a larger mapping experiment in downtown Santa Barbara. In addition, an Expectation Maximization (EM) based approach for optimizing important SNR model parameters was presented along with an encouraging first set of experimental results.

4.1.1 Open Issues

The open issues for the 3D map building work fall into several categories, which relate broadly to improving efficiency in terms of measurement usage and map representation, as well as accelerating the mapping process generally.

Improving the map representation

While the proposed approach was in the context of uniform occupancy grid (OG) maps, embedding it in more scalable hierarchical (e.g. octree) representations remains to be done. While complex (requiring heuristics for map cell splitting and merging, as well as more dynamic maps and factor graphs generally), one benefit of adopting this

approach would be to reduce the memory footprint of the map as well as the number of corresponding edges in the factor graph. In addition, using an octree OG representation would likely speed up and improve the accuracy of the mapping process, as fewer variables (map cells) would need to be estimated with the same amount of measurement data.

Incorporating a motion model

The SLAM approach could also be improved using a motion model to connect the poses $\{x_t\}$, much as in the localization (Particle Filter) model. This extra constraint would smooth the poses and likely improve 3D map accuracy and measurement efficiency as well. In addition, more measurements could be used – no decimation of the inputs would be needed to compensate for correlated GNSS positioning errors. Solving such a system would perhaps be more difficult due to the extra edges in the factor graph, but running iterative Belief Propagation (mapping) and Particle Filtering (localization), in an EM-like loop, is one option. Another option is to perform motion modeling *directly* in BP by introducing additional factor nodes connecting consecutive poses x_t and x_{t+1}. This could then be solved using Particle BP [26] or Non-Parametric BP algorithms [50].

Practical semantic mapping

Occupancy grids are naturally suited for certain applications, such as robotic navigation and obstacle avoidance. However, generating semantic outputs (i.e. building footprints and heights) via post processing techniques would be an exciting avenue for using GNSS SNR data to generate practical, *denoised* maps for other use cases (human

navigation, automatic map updates in cities, etc.). This sort of post processing could leverage recent advances in computer vision, image segmentation, and vectorization.

Enabling map updates at scale

In our implementation, we experimented on static datasets (and maps), and used a naive, Bulk Synchronous Parallel (BSP) scheduler for message passing in BP. To handle large mapping problems and updates to maps over time, the factor graph could instead be appended to over time, or itself used as a lower dimensional prior for future measurement data (factor graphs). In addition, to make BP scalable, map updates should not scale with the size of the map (as would be the case with BSP), but instead should size with amount of new information presented. One way to achieve this would to be to only pass message along factor graph edges whose neighbors' estimates (beliefs) had not yet converged, much as in [18].

Moving towards online SLAM

Online SLAM approaches as presented in our related work from UCSB [32] remain an interesting direction for scaling and speeding up the GNSS map building problem (possibly enabling on-device mapping in real-time). In addition, advancing this solution would help us move towards consolidating the mapping framework it with its cousin, PF-based localization. This dissertation did not focus on the online SLAM case, but simple improvements on [32], such as:

- Tuning down (using a less aggressive) inverse sensor model that doesn't over-interpret NLOS readings;
- Using hierarchical octree representations to make sensing easier and more efficient;
- Employing additional sensors and transportation models (such inertial fusion and road network map matching for vehicular models),

could make this simpler mapping algorithm more accurate and practical for real-world applications.

Reflection modeling

A very interesting direction that could be explored is the explicit modeling of reflections for NLOS signals. In the case of first order reflections (the most common scenario), by attributing raw pseudorange path delay to the reflection itself, an approximate reflection point could be localized. This is particularly appealing if other means (such as SLAM or sensor fusion) are available for determining the location of the GNSS receiver. In cities, rich multipath environments could then be *exploited* to map the facades of buildings. Recently, such approaches have been proposed for reflection-aided localization using GNSS and laser range finders [63].

4.2 Localization using GNSS SNR measurements

As a follow-up to our mapping work, we presented a practical, real-time capable, Bayesian localization framework using GNSS SNR measurements. We presented the *Probabilistic* Shadow Matching system, which reused many of the same physically inspired probabilistic models from mapping (for example, the binary LOS/NLOS SNR measurement model and the occupancy grid map representation).

Two different flavors of localization algorithms based on Particle Filters (PF) were presented: The initial, simple "Bootstrap PF" and the more robust and high performing, optimal sampling "Advanced PF". Both PFs make significant use of Rao-Blackwellization, which is highly desirable and made possible for Shadow Matching because only the position states are highly non-linear and non-Gaussian in the measurement model. An efficient implementation for the batch (offline) Particle Smoother solution was outlined. Important practical considerations for robust particle sampling, efficient hierarchical (octree) map representation, and accelerated (cached) ray tracing were presented.

Finally, we presented two sets of experimental results in low rise and high rise environments, with different PF implementations. For the longer experiment in San Francisco, we presented metrics on position and velocity error, showing both were significantly reduced. In particular, the cross track distance error (component of distance error in the cross-street direction) and heading error were shown to be drastically reduced for the real-time and offline localization algorithms. In addition, the jumpiness of the localization filter output was shown to be improved over Android's native position provider.

4.2.1 Open issues

Much of the open work for Probabilistic Shadow Matching relates to deeper, more tightly coupled GNSS architectures, as well as porting the positioning algorithms down to the GNSS device.

Tight coupling with raw GNSS measurements

In our Probabilistic SM framework, we use a loosely couple GNSS architecture. This architecture is still very susceptible to large PVT errors in deep urban environments which occur when large numbers of reflected signals are received. As a future improvement, more tightly coupled measurement models should be used, directly integrating the pseudorange and Doppler measurement Equations 3.1 with the SNR model. Recent research [22] has shown this to work well in the non-probabilistic Shadow Matching context, and it is the author's belief that this would work even better with our probabilistic PF-based architecture. This direction of research is particularly relevant given the Android smartphone platform's recent push to open up availability of raw GNSS measurements.

Coupling with other sensors

Fusing Shadow Matching with other sensors, such as inertial sensors (accelerometers and gyroscopes) for measuring motion, pressure sensors (barometers) for altitude, Bluetooth or WiFi RSSI for proximity detection, etc., is an open issue. Many of these techniques require low latency, high bandwidth signal processing running on the device

– essentially Kalman Filters – which is quite different than the cloud-, PF-based architecture we propose. However, by moving some of the probabilistic Shadow Matching on device, or feeding back its output at a latency from the cloud, these additional sensors can be more easily used. This would enable extending the localization framework to other important applications (such as 3D aircraft localization in cities).

Probabilistic Shadow Matching on the device

By encoding the 3D shadowing environment as tightly-packed octrees, or instead providing precomputed shadow surfaces or shadowing skyviews from the server, it should be possible to port the Shadow Matching PF down to the device to perform *on board* positioning. This will simplify the overall cloud based architecture we presented, cut cloud-based server operating costs, reduce network latency (and add resiliency against network outages), and facilitate sensor fusion. Crucial concerns are the memory footprint of the 3D shadowing information, the required long term storage (disk usage) if cached shadows are used, and mobile network bandwidth utilization. Currently, these considerations likely limit the practicality of such algorithms except for narrow use cases. Note, however, that this may change with the upcoming roll out of high speed (multi-gigabit) fifth generation (5G) mobile networks, as well as the always-improving hardware that is available on mobile GNSS devices (such as graphics processing units [GPUs] for ray tracing on smartphones).

Reflection modeling

Much as in the mapping case, directly modeling reflections could be used to improve localization given a precise map. We note that precise maps are relatively easy to come by in some geographies (e.g., from aerial Lidar surveys such as [58]), making this avenue of future work particularly interesting.

Incorporating other forms of map data

The particle filter architecture is a uniquely flexible approach that can fuse other sorts of map data. Incorporating traditional "road network" based map matching (see [24]) is an interesting line of work that could be useful for improving vehicular positioning in urban cores.

Appendices

Appendix A

A.1 Proof of Theorem 2.1

Proof. The initial expression for the message from a measurement factor $f_{t,n}$ to an adjacent map cell variable m_i is provided in Equation 2.10. Without loss of generality, let the incoming messages be normalized such that $u_{j\to(t,n)}(0) + u_{j\to(t,n)}(1) = 1$.

For $m_i = 1$, the ray corresponding to $f_{t,n}$ is blocked by cell m_i. In this case, $p(z_{t,n}|m, x_t) = f_{NLOS}(z_{t,n})$, and since the incoming messages are normalized to sum to unity, we have

$$\begin{aligned} U_{(t,n)\to i}(1) &= \sum_{m\setminus m_i} p(z_{t,n}|m, m_i = 1, x_t) \prod_{j\in\mathcal{M}(t,n)\setminus i} u_{j\to(t,n)}(m_j) \\ &= f_{NLOS}(z_{t,n}) \sum_{m\setminus m_i} \prod_{j\in\mathcal{M}(t,n)\setminus i} u_{j\to(t,n)}(m_j) \\ &= f_{NLOS}(z_{t,n}) \end{aligned} \tag{A.1}$$

For $m_i = 0$, the ray corresponding to $f_{t,n}$ is unblocked by cell m_i but could be blocked by any other cell intersected by the receiver-satellite ray. Let $\mathcal{M}(t,n) = \{j_1, j_2, ..., i\}$ represent the indices of these other cells and cell i. Then we can define two possible

complimentary events:

$$\begin{aligned} A : m_1 = 0 \wedge m_2 = 0 \wedge ... &\implies p(z_{t,n}|m, x_t) = f_{LOS}(z_{t,n}) \\ A^c : m_1 = 1 \vee m_2 = 1 \vee ... &\implies p(z_{t,n}|m, x_t) = f_{NLOS}(z_{t,n}) \end{aligned} \tag{A.2}$$

With this definition, we have the following reordering of Equation 2.10 as sums over these two events.

$$\begin{aligned} U_{(t,n)\to i}(0) &= \sum_{m\setminus m_i} p(z_{t,n}|m, m_i = 1, x_t) \prod_{j\in\mathcal{M}(t,n)\setminus i} u_{j\to(t,n)}(m_j) \\ &= f_{LOS}(z_{t,n}) \sum_{m\setminus m_i : A} \prod_{j\in\mathcal{M}(t,n)\setminus i} u_{j\to(t,n)}(m_j) \\ &\quad + f_{NLOS}(z_{t,n}) \sum_{m\setminus m_i : A^c} \prod_{j\in\mathcal{M}(t,n)\setminus i} u_{j\to(t,n)}(m_j) \end{aligned} \tag{A.3}$$

However, since the messages are normalized to sum to one and event A^c is complementary to event A, we have

$$\sum_{m\setminus m_i : A^c} \prod_{j\in\mathcal{M}(t,n)\setminus i} u_{j\to(t,n)}(m_j) = 1 - \sum_{m\setminus m_i : A} \prod_{j\in\mathcal{M}(t,n)\setminus i} u_{j\to(t,n)}(m_j) \tag{A.4}$$

Furthermore, event A only happens in one scenario (all map cells are empty), so:

$$\sum_{m\setminus m_i : A} \prod_{j\in\mathcal{M}(t,n)\setminus i} u_{j\to(t,n)}(m_j) = \prod_{j\in\mathcal{M}(t,n)\setminus i} u_{j\to(t,n)}(0) = \frac{\gamma_{t,n}(0)}{u_{i\to(t,n)}(0)} \tag{A.5}$$

where we have introduced the definition from Equation 2.13.

Finally, substituting Equation A.5 into Equation A.4, and then that into Equation A.3, we have

$$U_{(t,n)\to i}(0) = f_{LOS}(z_{t,n}) \left(\frac{\gamma_{t,n}(0)}{u_{i\to(t,n)}(0)} \right) + f_{NLOS}(z_{t,n}) \left(1 - \frac{\gamma_{t,n}(0)}{u_{i\to(t,n)}(0)} \right) \tag{A.6}$$

Reorganizing terms, and dividing Equations A.1 and A.6 by $f_{NLOS}(z_{t,n})$, we obtain Equation 2.11. □

A.2 Proof of Theorem 2.2

Proof. We prove these results briefly by bounding the computation in the two stages which form the bulk of the Loopy Belief Propagation computation: message passing from variables to factors, and then vice versa.

Message passing from variables to factors

In the log domain, efficient computation of messages from each variable node m_i to its neighboring factors using (2.8) involves the summation of all incoming messages (computing the belief) and then subtracting out the contribution from individual messages. This amounts to $2 \times |\mathcal{N}(i)|$ operations, proving (2.17).

To bound $E\left(\sum_i |\mathcal{N}(i)|\right)$, consider the cell m_i located at height $h_i < H$ along the axis of M, and define the Bernoulli random variable (RV) $I_{t,n}(m_i)$ which indicates whether m_i is intersected by the LOS path ray at time t to the nth satellite. Additionally, define $S(m_i|\phi_{t,n}, \theta_{t,n})$ as the elliptical shadow of m_i on the ground plane for a given set of satellite coordinates, as shown in Figure A.1. By straightforward trigonometric calculation, if

$$D > 2h_i \cot \phi_{\min} + d \csc \phi_{\min} \tag{A.7}$$

(i.e., G_M is "large enough"), then $S(m_i|\phi_{t,n}, \theta_{t,n}) \subset G_M$ and

$$\Pr\left(I_{t,n}(m_i) = 1 | \phi_{t,n}, \theta_{t,n}\right) = \left(\frac{d}{D}\right)^2 \csc \phi_{t,n} \tag{A.8}$$

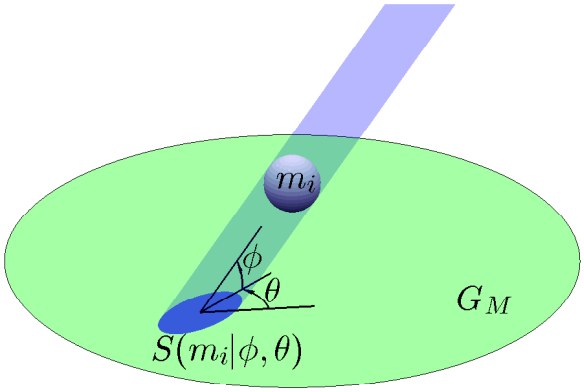


Figure A.1: Simplified illustration of the SNR measurement process. The GNSS receiver must sample from inside $S(m_i|\phi,\theta)$ for cell m_i to be intersected by the LOS path.

(the ratio of the area of the shadow to that of the ground plane). Since $f_\phi(\phi) = \cos(\phi)$ and $\theta \sim U(0, 2\pi)$, this implies:

$$\begin{aligned} E\left(I_{t,n}(m_i)\right) &= \int_{\phi_{\min}}^{\pi/2} \left(\frac{d}{D}\right)^2 \cot\phi \; d\phi \\ &= \left(\frac{d}{D}\right)^2 \ln\left(\csc\phi_{\min}\right) \end{aligned} \tag{A.9}$$

Noting that $|\mathcal{N}(i)| = \sum_{t,n} I_{t,n}(m_i)$ is just the RV which counts the total number of times m_i is intersected over the entire batch of measurements, by the union bound we have

$$E|\mathcal{N}(i)| < NT \left(\frac{d}{D}\right)^2 \ln\left(\csc\phi_{\min}\right) \tag{A.10}$$

for m_i as defined. However, for cells m_j closer to the edges, more of the shadows $S(m_j|\phi_{t,n},\theta_{t,n})$ would fall outside the borders; similarly, for cells at lower altitudes h_j, shadows would overlap to a greater degree. Consequently, the bound is only looser in both cases, and therefore holds for every cell. Finally, multiplying by the number of map

cells

$$L \approx \frac{3}{2}\left(\frac{H}{d}\right)\left(\frac{D}{d}\right)^2 \tag{A.11}$$

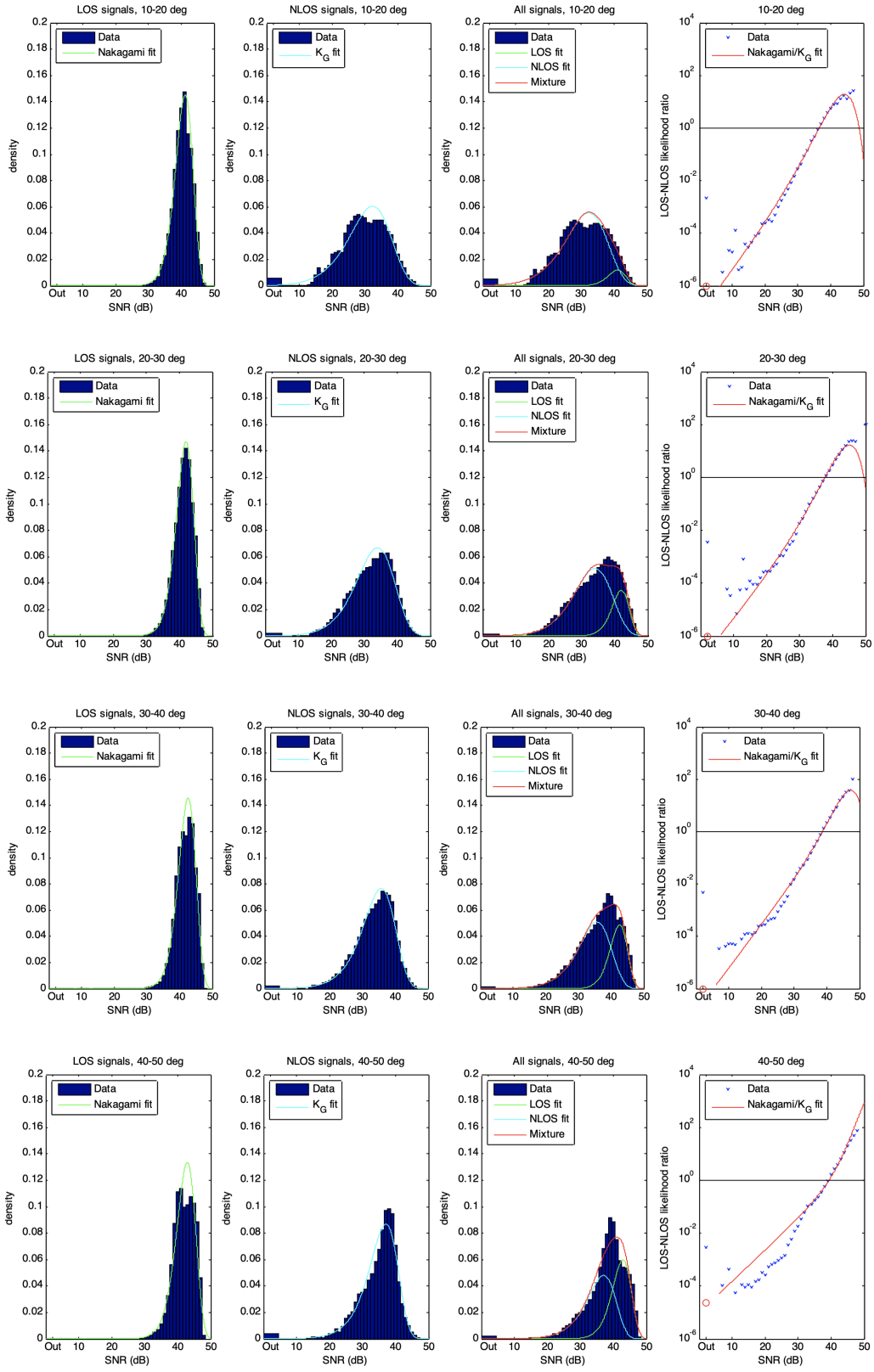
(since we have a cylindrical mapping region), we obtain (2.18).

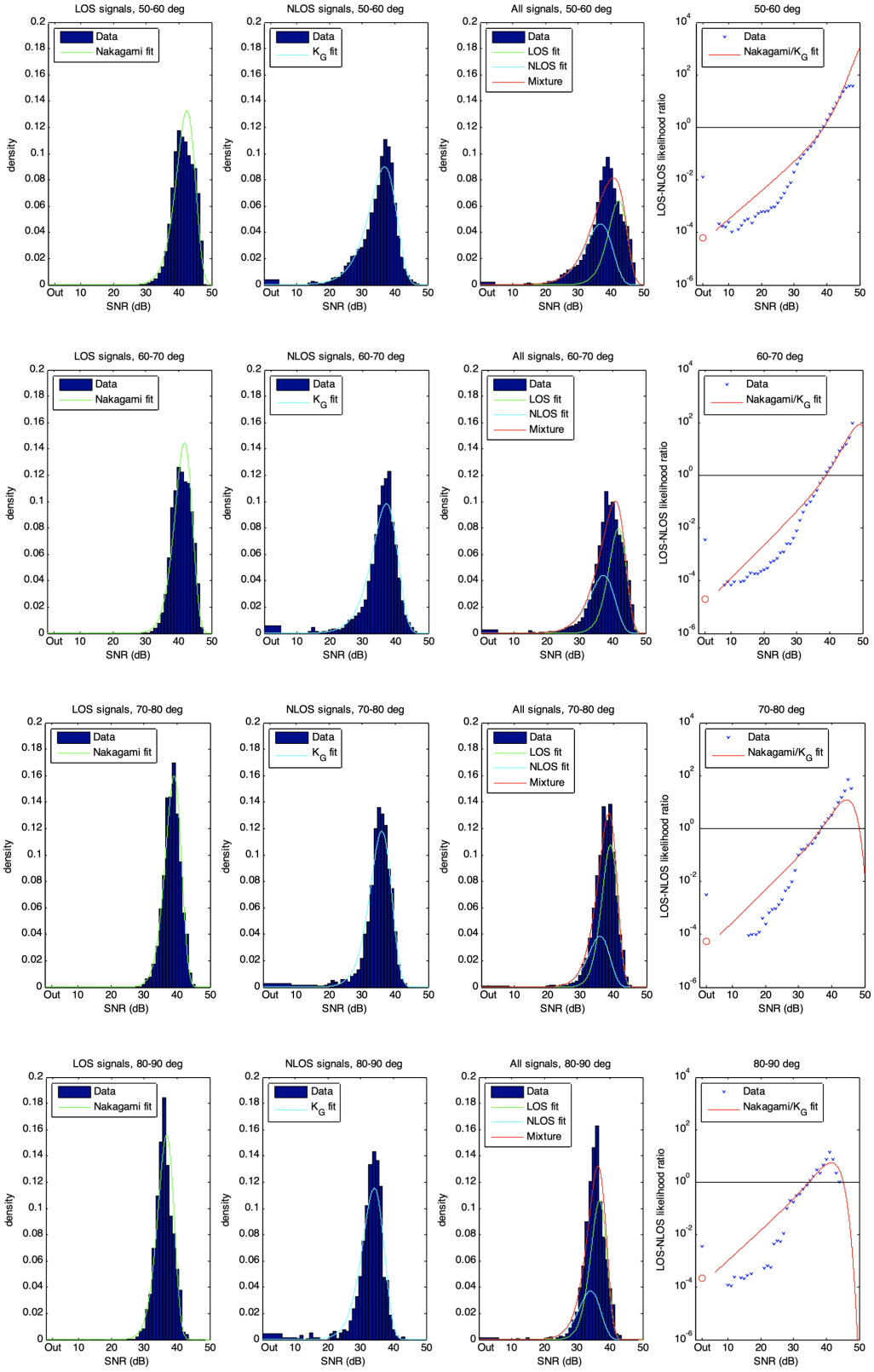
Message passing from factors to variables

For all outgoing messages from each measurement factor $f_{t,n}$, efficient log-domain use of (2.13) followed by (2.11) can be shown to require $5 \times |\mathcal{M}(t,n)|$ additions, $4 \times |\mathcal{M}(t,n)|$ multiplications, and $|\mathcal{M}(t,n)|$ exponential/logarithm function evaluations. Because $\sum_{t,n}|\mathcal{M}(t,n)|$ is just counting edges of the sub-graph with nodes $\{f_{t,n}\}$ and $\{m_i\}$, we have $\sum_{t,n}|\mathcal{M}(t,n)| = \sum_i |\mathcal{N}(i)|$, proving Equation (2.17). □

A.3 SNR model fitting results

In Section 2.5.4, we presented the SNR model fits for a single satellite elevation bin. Here, we provide the results for all nine bins for reference. We note the quality of the fits are better at lower elevations, as NLOS satellites become rarer (i.e., fewer points to fit against) at higher elevations.


LOS signals, 10-20 deg
NLOS signals, 10-20 deg
All signals, 10-20 deg
10-20 deg
LOS signals, 20-30 deg
NLOS signals, 20-30 deg
All signals, 20-30 deg
20-30 deg
LOS signals, 30-40 deg
NLOS signals, 30-40 deg
All signals, 30-40 deg
30-40 deg
LOS signals, 40-50 deg
NLOS signals, 40-50 deg
All signals, 40-50 deg
40-50 deg
Data
Nakagami fit
K_G fit
LOS fit
NLOS fit
Mixture
Nakagami/K_G fit
density
LOS-NLOS likelihood ratio
SNR (dB)
Out


LOS signals, 50-60 deg
NLOS signals, 50-60 deg
All signals, 50-60 deg
50-60 deg
LOS signals, 60-70 deg
NLOS signals, 60-70 deg
All signals, 60-70 deg
60-70 deg
LOS signals, 70-80 deg
NLOS signals, 70-80 deg
All signals, 70-80 deg
70-80 deg
LOS signals, 80-90 deg
NLOS signals, 80-90 deg
All signals, 80-90 deg
80-90 deg
Data
Nakagami fit
K_G fit
LOS fit
NLOS fit
Mixture
Nakagami/K_G fit
density
LOS-NLOS likelihood ratio
SNR (dB)
Out

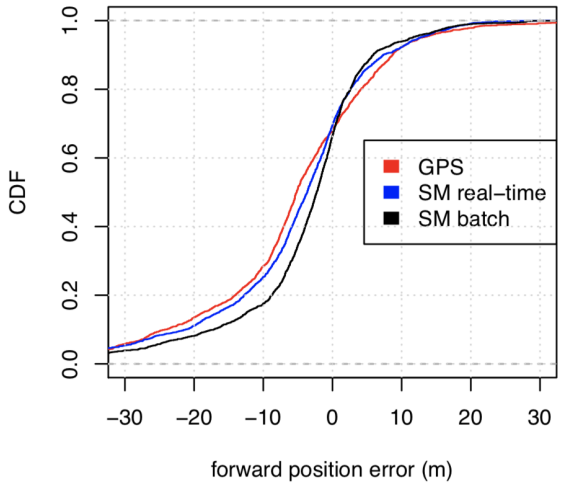
Appendix B

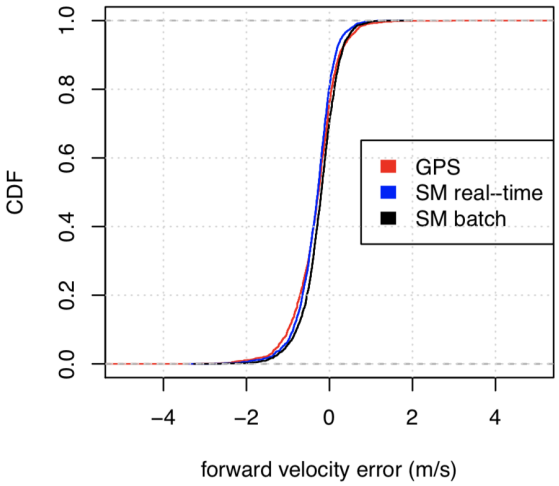
B.1 Results for the SF localization experiment

Additionally, we provide some results for the San Francisco localization experiment (Section 3.4.2).

Firstly, we show the forward position and velocity error distributions before and after applying Probabilistic SM. We define these metrics as the (signed) components of position/velocity error projected along the ground truth path. As can be seen in Figure B.1, we see slight reductions in these metrics, particularly forward position error for the batch case: this is intuitive, as incorporating future measurements inherently should reduce lag in the system output.

Secondly, we present what we term "User Interface" (UI) metrics for "jumpiness" in the position output. These are defined as simple discrete derivatives of the input and output location traces, with lower values signifying smoother (more pleasant, less jumpy) positioning. For example, the metric "UI speed" is defined as the single discrete





	GPS	SM real–time	SM batch
10%	–24	–21.2	–17.1
25%	–11.2	–10.1	–7.04
50%	–5.18	–3.64	–2.28
75%	2.3	1.15	1.29
90%	8.67	7.33	5.87
mean	–5.85	–5.51	–3.91
std	13.5	12.3	10.8

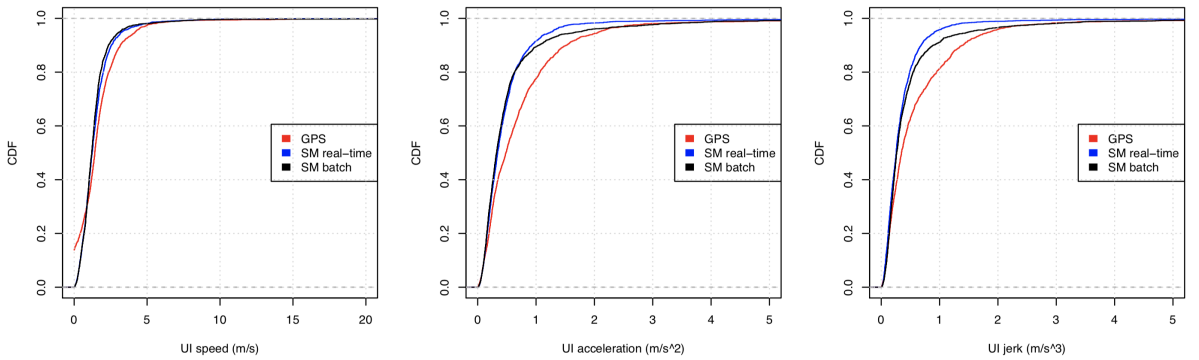
	GPS	SM real–time	SM batch
10%	–0.997	–0.873	–0.792
25%	–0.607	–0.567	–0.463
50%	–0.269	–0.281	–0.197
75%	–0.0148	–0.0624	0.0554
90%	0.24	0.145	0.285
mean	–0.323	–0.336	–0.231
std	0.542	0.451	0.45

Figure B.1: Forward position and velocity error metrics for the San Francisco localization experiment.

derivative

$$\text{UI speed} = \frac{\hat{x}_t - \hat{x}_{t-1}}{T} \tag{B.1}$$

where $\hat{x}_t$ is the MMSE *position* point estimate for the particle set (computed via Equation 3.23), and T is the time difference between epochs $t-1$ and t. "UI acceleration" and "UI jerk" are then derivatives and second derivatives of the "UI speed" metric. It can be readily observed in Figure B.1 that the Probabilistic SM framework generally reduces jumpiness both in real-time mode and further with additional offline processing.



	GPS	SM real–time	SM batch
25%	0.655	0.781	0.795
50%	1.42	1.23	1.21
67%	1.82	1.59	1.5
90%	3.16	2.55	2.35
95%	4.19	3.28	3.14
98%	5.33	4.77	4.67
mean	1.64	1.49	1.46

	GPS	SM real–time	SM batch
25%	0.226	0.191	0.179
50%	0.483	0.34	0.324
67%	0.748	0.481	0.461
90%	1.52	0.925	1.04
95%	2.08	1.27	1.72
98%	2.87	1.77	3.14
mean	0.743	0.491	0.576

	GPS	SM real–time	SM batch
25%	0.168	0.138	0.156
50%	0.347	0.257	0.265
67%	0.571	0.358	0.386
90%	1.39	0.682	0.888
95%	1.87	0.915	1.5
98%	2.8	1.33	2.88
mean	0.626	0.368	0.495

Figure B.2: UI jumpiness metrics for the San Francisco localization experiment.

Bibliography

[1] A. Abdi, W. C. Lau, M.-S. Alouini, and M. Kaveh. A new simple model for land mobile satellite channels: first-and second-order statistics. *IEEE Trans. on Wireless Communications*, 2(3):519–528, 2003.

[2] A. Abdi, C. Tepedelenlioglu, M. Kaveh, and G. Giannakis. On the estimation of the K parameter for the Rice fading distribution. *IEEE Communications Letters*, 5(3):92–94, 2001.

[3] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on signal processing*, 50(2):174–188, 2002.

[4] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Trans. on Signal Processing*, 50(2):174–188, 2002.

[5] B. Ben-Moshe, E. Elkin, H. Levi, and A. Weissman. Improving accuracy of GNSS devices in urban canyons. In *Proc. of the 23rd Canadian Conference on Computational Geometry*, 2011.

[6] P. S. Bithas, N. C. Sagias, and P. T. Mathiopoulos. The bivariate generalized-k (k g) distribution and its application to diversity receivers. *IEEE Transactions on Communications*, 57(9):2655–2662, 2009.

[7] City and C. of San Francisco. City maps. Available at `https://sfgov.org/services/city-maps` (2019/03/26).

[8] J. H. Conway and N. J. A. Sloane. *Sphere packings, lattices and groups*, volume 290. Springer Science & Business Media, 2013.

[9] H. Coxeter. Virus macromolecules and geodesic domes. *A spectrum of mathematics*, pages 98–107, 1971.

[10] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 29(6):1052–1067, 2007.

[11] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

[12] M. W. M. G. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, and M. Csorba. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Trans. on Robotics and Automation*, 17(3):229–241, 2001.

[13] A. Doucet, N. De Freitas, K. Murphy, and S. Russell. Rao-blackwellised particle filtering for dynamic bayesian networks. In *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, pages 176–183. Morgan Kaufmann Publishers Inc., 2000.

[14] A. Elfes. Occupancy grids: A stochastic spatial representation for active robot perception. In *Proc. of the Sixth Conference on Uncertainty in AI*, 1990.

[15] G. Elidan. Residual belief propagation: Informed scheduling for asynchronous message passing. In *Proc. of the 22nd Conference on Uncertainty in AI*, 2006.

[16] P. Elinas, R. Sim, and J. J. Little. σSLAM: Stereo vision SLAM using the Rao-Blackwellised particle filter and a novel mixture proposal distribution. In *Proc. of the International Conference on Robotics and Automation.*, pages 1564–1570, 2006.

[17] B. Ferris, D. Fox, and N. Lawrence. WiFi-SLAM using Gaussian process latent variable models. In *Proc. of the 20th International Joint Conference on Artificial Intelligence*, pages 2480–2485, 2007.

[18] J. Gonzalez, Y. Low, and C. Guestrin. Parallel splash belief propagation. *Journal of Machine Learning Research*, 1:1–48, 2009.

[19] J. Gonzalez, Y. Low, and C. Guestrin. Residual splash for optimally parallelizing belief propagation. In *Artificial Intelligence and Statistics*, pages 177–184, 2009.

[20] J. E. Gonzalez, Y. Low, C. Guestrin, and D. O'Hallaron. Distributed parallel inference on large factor graphs. In *Proc. of the Conference on Uncertainty in Artificial Intelligence*, pages 203–212, 2009.

[21] E. W. Grafarend and F. W. Krumm. *Map projections*. Springer, 2014.

[22] P. Groves and M. Adjrad. Multi-epoch 3d mapping aided gnss using a grid filter. In *Proc. of the 31st International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2018)*. Institute of Navigation (ION), 2018.

[23] P. D. Groves. Shadow matching: A new gnss positioning technique for urban canyons. *The journal of Navigation*, 64(3):417–430, 2011.

[24] M. Hashemi and H. A. Karimi. A critical review of real-time map-matching algorithms: Current issues and future directions. *Computers, Environment and Urban Systems*, 48:153–165, 2014.

[25] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard. Octomap: An efficient probabilistic 3d mapping framework based on octrees. *Autonomous robots*, 34(3):189–206, 2013.

[26] A. Ihler and D. McAllester. Particle belief propagation. In *Artificial Intelligence and Statistics*, pages 256–263, 2009.

[27] A. T. Ihler, E. B. Sudderth, W. T. Freeman, and A. S. Willsky. Efficient multiscale sampling from products of gaussian mixtures. *Advances in Neural Information Processing Systems*, 16:1–8, 2004.

[28] A. Irish, J. Isaacs, D. Iland, J. Hespanha, E. Belding, and U. Madhow. Demo: Shadowmaps, the urban phone tracking system. In *Proceedings of the 20th Annual International Conference on Mobile Computing and Networking*, MobiCom '14, pages 283–286, New York, NY, USA, 2014. ACM.

[29] A. T. Irish, D. Iland, J. T. Isaacs, J. P. Hespanha, E. M. Belding, and U. Madhow. Using crowdsourced satellite snr measurements for 3d mapping and real-time gnss positioning improvement. In *Proceedings of the 6th annual workshop on Wireless of the students, by the students, for the students*, pages 5–8. ACM, 2014.

[30] A. T. Irish, J. T. Isaacs, F. Quitin, J. P. Hespanha, and U. Madhow. Belief Propagation based localization and mapping using sparsely sampled GNSS SNR measurements. In *Proc. of IEEE International Conference on Robotics and Automation*, 2014.

[31] A. T. Irish, J. T. Isaacs, F. Quitin, J. P. Hespanha, and U. Madhow. Probabilistic 3D mapping based on GNSS SNR measurements. In *Proc. of IEEE International Conference on Acoustics and Signal Processing*, 2014.

[32] J. T. Isaacs, A. T. Irish, F. Quitin, U. Madhow, and J. P. Hespanha. Bayesian localization and mapping using GNSS SNR measurements. In *Proc. of IEEE Position Location and Navigation Symposium*, 2014.

[33] E. D. Kaplan and C. J. Hegarty. *Understanding GPS: Principles and Applications*. Artech House, second edition, 2005.

[34] K. Kim, J. Summet, T. Starner, D. Ashbrook, M. Kapade, and I. Essa. Localization and 3D reconstruction of urban scenes using GPS. In *Proc. of the IEEE International Symposium on Wearable Computers.*, pages 11–14, 2008.

[35] M. Klaas, M. Briers, N. De Freitas, A. Doucet, S. Maskell, and D. Lang. Fast particle smoothing: If i had a million particles. In *Proceedings of the 23rd international conference on Machine learning*, pages 481–488. ACM, 2006.

[36] F. Kschischang, B. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Trans. on Information Theory*, 47(2):498–519, 2001.

[37] A. Laourine, M.-S. Alouini, S. Affes, and A. Stéphenne. On the capacity of generalized-k fading channels. *IEEE Transactions on Wireless Communications*, 7(7):2441–2445, 2008.

[38] X. R. Li and V. P. Jilkov. Survey of maneuvering target tracking. part i. dynamic models. *IEEE Trans. on Aerospace and Electronic Systems*, 39(4):1333–1364, 2003.

[39] C. Loo. A statistical model for a land mobile satellite link. *IEEE Trans. on Vehicular Technology*, 34(3):122–127, 1985.

[40] K. P. Murphy, Y. Weiss, and M. I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Proc. of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, pages 467–475, 1999.

[41] H. Niederreiter. Low-discrepancy and low-dispersion sequences. *Journal of number theory*, 30(1):51–70, 1988.

[42] M. Obst, S. Bauer, P. Reisdorf, and G. Wanielik. Multipath detection with 3D digital maps for robust multi-constellation GNSS/INS vehicle localization in urban areas. In *Proc. of the Intelligent Vehicles Symposium.*, pages 184–190, 2012.

[43] M. Obst and G. Wanielik. Probabilistic non-line-of-sight detection in reliable urban GNSS vehicle localization based on an empirical sensor model. In *Proc. of the Intelligent Vehicles Symposium.*, 2013.

[44] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausble Inference*. Morgan Kaufmann Pub, 1988.

[45] F. Perez-Fontan, M. Vazquez-Castro, S. Buonomo, J. P. Poiares-Baptista, and B. Arbesser-Rastburg. S-band LMS propagation channel behaviour for different environments, degrees of shadowing and elevation angles. *IEEE Trans. on Broadcasting*, 44(1):40–76, 1998.

[46] A. Ranganathan, M. Kaess, and F. Dellaert. Loopy SAM. In *Proc. of the International Joint Conference on Artificial Intelligence.*, pages 6–12, 2007.

[47] J. Revelles, C. Urea, and M. Lastra. An efficient parametric algorithm for octree traversal. In *Journal of WSCG*, pages 212–219, 2000.

[48] I. Schneider. Cell lines derived from late embryonic stages of drosophila melanogaster. *Development*, 27(2):353–365, 1972.

[49] S. Sharma and R. Mishra. A simulation model for nakagmi-m fading channel with m. *Power*, 2(2):2, 2015.

[50] E. B. Sudderth, A. T. Ihler, M. Isard, W. T. Freeman, and A. S. Willsky. Nonparametric belief propagation. *Communications of the ACM*, 53(10):95–103, 2010.

[51] U. G. Survey. The national map, 3d elevation program web page. Available at `https://www.usgs.gov/core-science-systems/ngp/3dep/about-3dep-products-services` (2019/03/26).

[52] G. A. Terejanu et al. Extended kalman filter tutorial. *University at Buffalo*, 2008.

[53] S. Thrun. Robotic mapping: A survey. *Exploring Artificial Intelligence in the New Millennium*, 1:1–35, 2003.

[54] S. Thrun. Simultaneous localization and mapping. In *Robotics and cognitive approaches to spatial mapping*, pages 13–41. Springer, 2008.

[55] S. Thrun, W. Burgard, and D. Fox. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping. In *Proc. of the International Conference on Robotics and Automation.*, volume 1, pages 321–328, 2000.

[56] S. Thrun, D. Fox, W. Burgard, et al. Monte carlo localization with mixture proposal distribution. In *AAAI/IAAI*, pages 859–865, 2000.

[57] S. Thrun and M. Montemerlo. The GraphSLAM algorithm with applications to large-scale mapping of urban structures. *International Journal on Robotics Research*, 25(5/6):403–430, 2005.

[58] M. E. U.S. Geological Survey, OpenTopography Facility. California arra lidar. Available at `https://catalog.data.gov/dataset/california-arra-lidar` (2019/03/26).

[59] L. Wang. Kinematic gnss shadow matching using particle filters. In *Proc. of the 27th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS+ 2014)*, pages 1907–1919. The Institute of Navigation, 2014.

[60] L. Wang, P. D. Groves, and M. K. Ziebart. Shadow matching: Improving smartphone GNSS positioning in urban environments. In *Proc. of the China Satellite Navigation Conference.*, pages 613–621, 2013.

[61] N. Wang, X. Song, and J. Cheng. Generalized method of moments estimation of the nakagami-m fading parameter. *IEEE Transactions on Wireless Communications*, 11(9):3316–3325, 2012.

[62] A. Weissman, B. Ben-Moshe, H. Levi, and R. Yozevitch. 2.5D mapping using GNSS signal analysis. In *Proc. of the Workshop on Positioning Navigation and Communication*, pages 1–6, 2013.

[63] W. Wen, G. Zhang, and L.-T. Hsu. Correcting gnss nlos by 3d lidar and building height. In *Proc. of the 31st International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS+ 2018)*, pages 3156–3168, 10 2018.