UNIVERSITY OF CALIFORNIA, SAN DIEGO

STRATEGIES IN TRACKING AND LOCALIZATION OF DISTRIBUTED UNDERWATER SYSTEMS

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

Doctor of Philosophy

in

Electrical Engineering

(Computer Engineering)

by

Diba Mirza

Committee in charge:

Professor Bhaskar D. Rao, Chair
Professor Curt Schurgers, Co-Chair
Professor William S. Hodgkiss
Professor Jules Jaffe
Professor Ryan Kastner
Professor Mohan Trivedi

2010

The Dissertation of Diba Mirza is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

_____

_____

_____

_____

_____

Co-Chair

_____

Chair

University of California, San Diego

2010

DEDICATION

To my family

EPIGRAPH

...the ambiguous is the reality and the unambiguous is merely a special case of it, where we finally manage to pin down some very special aspect.

- *David Bohm*

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

Chapter 2 appears, in part in the *Proceedings of* MTS/IEEE Oceans'08.

Chapter 3 appears in part in the IEEE *Journal on Selected Areas in Communication (JSAC) Special Issue on Underwater Wireless Communications and Networks, 2008.* Chapter 3 also appears in part in *the fifth ACM International Workshop on Underwater Networks, 2010.*

Chapter 4 appears, in part, in the *Proceedings of the ACM International Workshop on Underwater Networks (WUWNET) in conjunction with ACM SenSys, 2009.*

Chapter 5 appears, in part, in the *Proceedings of the ACM International Workshop on Underwater Networks in conjunction with ACM MobiCom 2008.* Chapter 5 also appears in part, in the IEEE *Journal on Selected Areas in Communication (JSAC) Special Issue on Underwater Wireless Communications and Networks, 2008.*

Part of Chapter 3 was done in collaboration with Feng Lu. Appendex A.1 was written under the supervision of Dr. Jules Jaffe. I would like to thank him for providing me with relevant material from his own research and also referring other scientific articles on oceanography. In all other aforementioned published work, I was the primary researcher and Curt Schurgers supervised the research. This work was supported, in part, by the NSF ECCS-0622005 SCHURGERS grant.

I am deeply grateful to my advisor, Prof. Schurgers. I thank him for supporting me consistently and critiquing my work at every stage. I was fortunate to receive feedback that always raised the bar and made every document I produced

better. I sincerely appreciate his insight and quick thinking which helped steer away from dead-end problems and made our discussions very engaging. I especially thank him for thoroughly reading my dissertation and all my writing to date. The PhD journey has not been an easy one. But Curt's broad perspective on life, sense of humor and sensitivity brought the kind of optimism that is so crucial to any difficult undertaking. For all this and more, I express my heartfelt gratitude.

I would also like to thank all the members of the committee for giving me their constructive feedback, at every opportunity. I sincerely thank Prof. Bhaskar Rao for extending his support at crucial times. Thank you for always being responsive in spite of your busy schedule. I am grateful to Dr. Jules Jaffe for educating me on the oceanographic side of things, for always being approachable and providing me with informative articles. Prof. Hodgkiss, thank you for your time, your helpful comments and for urging me on to delve deeper into existing work. I greatly appreciate Prof. Trivedi for his feedback and responsiveness. Prof. Kastner, thank you for freely sharing your lab resources and collaborating with us and the folks at CalIT to make (almost) impossible projects possible.

I thank all my friends for their warmth and affection. My dear parents and brother, thank you for always being supportive. I owe so much to you. And finally, Yoga, this would not have been possible without you.

VITA AND PUBLICATIONS

VITA

2002	Bachelors in Electrical and Computer Engineering, Birla Institute of Technology and Science, Pilani, India.
2003	System Engineer at Motorola, India.
2004	Recipient of the University of California, San Diego, Jacobs Fellowship.
2006	Master of Science in Electrical Engineering, University of California, San Diego
2008	Candidate of Philosophy in Electrical Engineering, University of California, San Diego
2010	Teaching Assistant in Electrical and Computer Engineering Department, University of California San Diego
2010	Doctor of Philosophy in Electrical Engineering, University of California, San Diego

PUBLICATIONS

1. **Mirza, D.** and Schurgers, C., "Energy-Efficient Ranging for Post-Facto Self-Localization in Mobile Underwater Networks", IEEE Journal on Selected Areas in Communication (*IEEE JSAC*) Special Issue on Underwater Wireless Communications and Networks, Vol. 26, Issue 9, pp.1697-1707, December 2008.

2. **Mirza, D.** and Schurgers, C., “On the Maximum Likelihood Performance of Beacon-based Tracking”, to be submitted, IEEE Transactions on Communications, 2010.
3. **Mirza, D.** Lu F. and Schurgers, C., “Optimal Collaborative Tracking in Mobile Underwater Networks using Graphical Inference”, to be submitted, IEEE Journal of Oceanic Engineering, 2010.
4. **Mirza, D.** and Schurgers, C., “Collaborative Tracking in Mobile Underwater Networks”, ACM International Workshop on Underwater Networks in conjunction with ACM SenSys 2009, Berkeley, California, November 2009.
5. **Mirza, D.** and Schurgers, C., “Motion-Aware Self-localization for Underwater Networks”, ACM International Workshop on Underwater Networks in conjunction with ACM MobiCom 2008, San Franscisco, California, September 2008.
6. **Mirza, D.,** Lu F. and Schurgers, C., “Efficient Broadcast MAC for Underwater Networks”, poster at ACM International Workshop on Underwater Networks in conjunction with ACM SenSys 2009, Berkeley, California, November 2009.
7. **Mirza, D.** and Schurgers, C.,“Collaborative Localization for Fleets of Underwater Drifters," MTS/IEEE Oceans’07, Vancouver, Canada, pp. 1-6, October 2007.
8. **Mirza, D.,** Lu F. and Schurgers, C., “Analyzing the effect of 3D geometric parameters on localization accuracy for underwater sensor networks”, poster presented at ACM International Workshop on Underwater Networks in conjunction with MobiCom, Los Angeles, CA, September 2006.

9. **Mirza, D.** and Schurgers, C., "Energy-efficient localization in networks of underwater drifters", ACM International Workshop on Underwater Networks in conjunction with ACM MobiCom 2007, Montreal, Canada, pp. 73-80, September 2007.
10. **Mirza, D.**, Lu F., and Schurgers, C., "TB-MAC: Efficient MAC-Layer Broadcast for Underwater Acoustic Sensor Networks," IEEE International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP'09), Melbourne, Australia, December 2009.
11. **Mirza D**., and C. Schurgers, "Design of Self-Localization Strategies for Networks of Underwater Drifters", MTS/IEEE Oceans'08, Quebec, Canada, pp. 1-8, September 2008.
12. Lee, S., Mounzer**,** J., **Mirza, D.** and Schurgers, C., "Low Cost, Medium Range Optical Communication for Underwater Test Beds", Demo at ACM International Workshop on Underwater Networks in conjunction with MobiCom, San Franscisco, California, September 2008.
13. Lu F., **Mirza, D.,** and Schurgers, C., "D-Sync: Doppler-based time synchronization for mobile underwater sensor networks", accepted in ACM International Workshop on Underwater Networks (WUWNET), 2010.

ABSTRACT OF THE DISSERTATION

STRATEGIES IN TRACKING AND LOCALIZATION OF DISTRIBUTED UNDERWATER SYSTEMS

by

Diba Mirza

Doctor of Philosophy in Electrical Engineering

(Computer Engineering)

University of California, San Diego, 2010

Professor Bhaskar D. Rao, Chair

Professor Curt Schurgers, Co-Chair

Distributed underwater systems, consisting of multiple sensing platforms can provide critical data to better understand complex and inter-related ocean processes. However, to correctly interpret data collected by such systems, we need to know when and where samples were obtained. Since GPS is not available underwater, the position

of devices should be estimated with respect to certain known references that may reside on the surface or on the sea-bed.

The main challenge is that devices are energy-constrained. Further, low-cost solutions to both the sensing platforms and the overall system design would be critical in making these systems more prevalent and available to scientists. Existing underwater tracking techniques are not well-suited to distributed systems because they are built around stand-alone platforms. They ignore vital relative information between devices and require long range communication which is both expensive and has high energy consumption. As a result, existing techniques do not scale well in multi-vehicle systems.

To address these challenges, we take a systems perspective to underwater positioning. This means that instead of viewing each node as a separate entity, as in traditional systems, we consider the network as a whole. Therefore, we shift our focus to tracking a collective, rather than independently positioning a number of devices. This paradigm-shift allows us to use collaboration between devices to improve the performance and energy-scalability of mobile distributed systems. However, it makes the problem of tracking and localization more complex.

Our proposed solutions draw on the framework of factor graphs to optimally and jointly estimate the trajectories of multiple nodes by combining information in 4 dimensions. This framework allows us to leverage both from network density and accurate motion information, if available. In addition, we have identified node mobility as a key factor that can both impede and improve performance. We show how mobility in combination with delays in medium access is an impairment to time-

synchronization and localization and propose cross-layer and collaborative approaches to counter its effect.

Our proposed strategies are essentially aimed at more efficient localization and tracking in underwater networks where resources are constrained. We believe that our techniques in combination with existing systems would address the localization requirements of a wide range of underwater applications.

CHAPTER 1

INTRODUCTION

The oceans play a vital role in the well-being of our planet. Yet they remain vastly unknown even after decades of exploration. Further, human activity has had a significant and growing impact on ocean-systems. Scientists predict that without making a conscientious effort towards preserving our natural resources, we can potentially tip the delicate ecological balance critical to life on the planet. Collecting diverse information about ocean processes will be vital in solving some of the imminent global problems of our century.

There are many existing ways in which ocean sensing is done. Bathymetric and sonar systems map underwater terrain using acoustics, large-scale surface phenomena are observed via satellite, sensors deployed on sea-beds detect seismic activity; various mobile platforms such as AUVs, underwater robots, gliders and floats are used for surveillance, exploration and sensing. Even divers collect valuable data in the form of video-recordings and actual samples. While many oceanographic systems have been developed for specific applications, we will elaborate and focus on one class of systems, namely *mobile distributed underwater systems*.

Physical, chemical and biological processes vary over a large range of spatial and temporal scales. As a result, comprehensive scientific studies require observing these phenomena at relevant scales. A single device equipped with a number of sensors can capture the temporal variations of one or more parameters, however over a limited region. To increase the extent of spatial observation multiple devices are often

deployed in various configurations. For example sensors are arranged at different depths on moorings. A number of such moorings can then span a geographic region. By deploying multiple instruments we can concurrently sample phenomena both in time and space and deduce their correlations. Such a system of underwater sensors and vehicles, operating in a coordinated manner forms a *distributed sampling system*. These systems provide a large scale-view of processes and are especially suited for studying process-variations and inter-connections. As such they may consist of stationary or mobile platforms or a combination of both.

We are particularly interested in systems with mobile platforms. This is because the ocean environment is inherently mobile. The dynamics of ocean currents dictate how chemicals, nutrients and organisms are transported within the ocean. As a result, many ocean processes need to be observed and followed within this intrinsically mobile environment. Mobile platforms naturally lend themselves to sensing underwater phenomena that are influenced by ocean currents. For example, a collection of mobile devices is more suitable for tracking an oil-spill rather than deploying many static sensors. In this dissertation we will specifically focus on mobile distributed systems, where devices observe phenomena in their own moving frame of reference. A rendering of an example system is depicted in Figure 1.1. It consists of a large collection (a swarm) of underwater devices such as guided AUVs, ROVs, gliders and floats. A typical system under study in this dissertation can consist of both passive and self-propelled vehicles. Equipped with a multitude of sensors, each device essentially collects samples in its own local neighborhood. Devices are deployed as a collective. In essence, they form a distributed and dynamic sampling system. A more

in depth discussion on the oceanographic applications of such systems is presented in Appendix A.1.

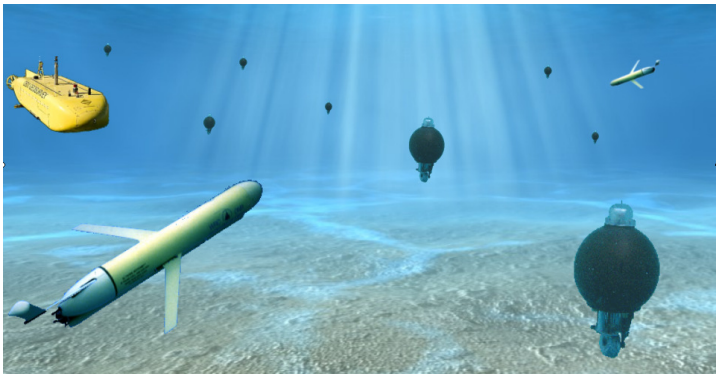


Figure 1.1: Underwater network of autonomous mobile platforms

To act as a sampling system, devices need to know their positions, so that spatial correlation of sampled data can be deduced. However, GPS service is not available underwater due to the large attenuation of radio waves. As such, the vehicles' positions must be estimated. A simple solution would entail having each device communicate with a number of known position references that may reside on the surface or at predetermined locations on the sea-bed. By communicating with a reference, the distance to the reference can be estimated which results in a geometric constraint on the (unknown) position of the device. Once enough constraints are established (generally 3), the device position can be estimated fairly accurately which is essentially how location service is provided by GPS satellites to terrestrial systems [Kap96]. However, for underwater systems one cannot rely on a pre-deployed infrastructure as with GPS satellites, instead we would have to provide this infrastructure, which can often be a costly undertaking. Even if such an infrastructure was put in place, it is impractical to assume that nodes will always be in range of fixed

localization buoys because the area occupied by the collective can be large. Further, devices may have uncontrollable mobility (as in the case of floats [Jaf06]) due to which they can drift away from surface transponders in unpredictable ways. Once devices can no longer communicate with references, they cannot obtain enough constraints to estimate their positions. However, when multiple vehicles are present, this problem can be overcome by *sharing* information.

Sharing is possible if devices were allowed to also communicate with each other, thereby operating in a networked fashion. Inter-node communication can be used to create more geometric constraints while the system is moving. We believe that these additional constraints can be used to improve the localization performance and we will dedicate most of the later chapters to exploring this overall *networked approach* to underwater localization. A broader overview of the research challenges in networking such systems can be found in [Hei06].

To sum up, just as distributed systems provide more spatially and temporally rich data using simpler platforms, networking opens the door for resource sharing and makes more useful data available for location tracking of the collective. This intuitive advantage motivates us to look at the problem of distributed underwater localization more closely.

1.1 Problem Motivation

The main motivation for addressing the problem of localization is that while navigation has been a fundamental part of underwater systems, it has been mainly designed for stand-alone platforms. In principle, such existing techniques can be extended to multi-vehicle systems, by tracking each device independently. While

accurate positioning (around a few meters) has been previously demonstrated using these techniques they rely heavily on either precise on-board sensors or frequent communication with known references [Bla03] [Kin03] [Vic98] [Whi98] [Spi76]. In fact, existing strategies essentially tradeoff device sophistication (accurate on-board sensors) [Ala02] [Yun01] [Hud98] [Whi99] [Bro97] versus infrastructure costs (well-calibrated baselines and planned deployments of references)[1][Bin06] [Lar00] [Lar02] [Jak05] [Bel91]. However, we would like to look at systems that play a different type of tradeoff, namely, many distributed nodes versus one expensive device.

The need for new solutions in such systems arises because resources are constrained. Specifically mobile platforms are severely *energy-constrained*. In addition, while *precision motion sensors and instruments* are valuable inputs to tracking algorithms, they are expensive and may be available to only a few nodes. Lastly, ensuring that each device is within the range of *position references* requires infrastructure which is costly.

As such, no single localization strategy would be suitable across all different oceanographic systems. Nevertheless, in the context of distributed systems we can improve upon existing techniques by using cooperation. While cooperative techniques are more prevalent in terrestrial networks and robotics [Zha08] [Ihl04] [Bis04] [Sha03] [Fox00], they have not been designed to specifically take advantage of the characteristics of oceanographic applications and do not address the challenges that arise in an underwater environment.

[1] An in-depth discussion on tracking multi-vehicle mobile using existing underwater tracking techniques is presented in Section A.2 of Appendix A.

1.2 Application Context

The tracking strategies that we will discuss in this dissertation are targeted for a class of underwater applications where position information is not required in real time. The main reason is that in many scientific applications data collected by devices is analyzed after it is available at a central location and real-time reporting is not needed. For example, intensive data collection is essential to building computer-models that can predict hazardous conditions such as eutrophication and hypoxia that are fatal to marine organisms. However building such models takes time. So, data can also be collected non-real time. Another example is using measurements of the flow-field to build high-resolution models of ocean currents [Fri06], [Che06], [Dal06].

If data is not strictly needed in real-time, it is not efficient to collect it via underwater communications. This is because underwater communication (which most often is based on acoustic signaling) consumes a lot of energy. So, instead many systems rely on batch data extraction using more efficient methods (not based on acoustics). For example, nodes could occasionally re-surface to send data via satellite links, or upload information to data-gathering entities via high speed optical links only when they are at close proximity [Vas05]. Lastly, data can be collected once devices have been retrieved, after the mission is over. The fact that data is not required in real-time also impacts how localization can be performed.

Running a full localization algorithm while nodes are submerged requires considerable communication between nodes. Since position information is not required until data is being analyzed, we propose that the actual tracking algorithms be run offline. While submerged, nodes measure their inter-node distances and store this

information locally along with other measurements of their motion from on-board sensors. This data is then collected at a central location and given as input to tracking algorithms. In the remainder of the dissertation, we will describe our solutions in the context of this system setup, unless specified otherwise. Our solutions are essentially targeted towards the subset of underwater applications that are *delay-tolerant*.

1.3 Problem Definition

The fundamental question that we address in this dissertation is:

How can we efficiently track the positions of a distributed system consisting of a collection of mobile underwater vehicles in the case where position information is not needed in real-time? In other words, how can we efficiently perform delay-tolerant underwater tracking of mobile vehicle collectives?

To answer this question we take a systems perspective to tracking distributed systems. This means that instead of viewing each node as a separate entity, as done traditionally, we consider the network as a whole. To track the collective efficiently we would like to *optimally* and *jointly* estimate the trajectories of all devices given all available data. This is a complex estimation problem because of the inter-dependence of a large number of unknown states. By *optimal* we imply estimating the Maximum Likelihood trajectories of devices. This problem has not been solved so far, in the most generic sense, to the best of our knowledge. Our goal is to improve upon underwater tracking strategies where data is only combined in the time-domain [Stu08] [Kin06] [Cor07] [Tri98] and also over terrestrial collaborative localization techniques where only spatial information is used [Mao07] [Gol05] [Gol06] [Doh01].

As a result of the above approach devices no longer have to be in direct communication of references and can use short to medium range acoustic modems to estimate inter-node distances which is both more energy efficient and more accurate compared to long-range systems. Therefore, these solutions would be scalable to larger networks and robust to changes in topology due to mobility. Such a collaborative technique would not only increase the geographical extent of distributed systems but also allow sharing of resources and information among devices. To solve this problem, we have laid down a probabilistic framework based on factor-graphs. This framework allows adding resources (such as Long Baseline (LBL) transponders, high resolution bathymetric sonars and gyrocompasses) as needed to meet localization accuracy requirements while operating within application and system level constraints. Since we use a probabilistic framework, the only information we need is a statistical characterization of the error in measurements obtained from motion sensors and acoustic ranging.

Another problem area is how accurately and energy-efficiently can we estimate the inter-node distances which are key inputs for tracking algorithms. While the actual tracking would be performed offline, following the setup described in the previous section, nodes have to communicate while they are submerged to perform ranging, i.e. estimate their distances with neighbors. Now, the unique characteristics of underwater acoustic communications open up a number of problems in this area. First, unlike terrestrial radio systems, the power consumed for transmitting acoustic signals underwater is very high (tens of watts) and much larger compared to the that consumed in the receive and idle states. As a result, to address energy-efficiency,

networking and localization protocols should be designed to minimize the total number of transmissions and/or the number of per-node transmissions. The second problem area that has not been given much attention is the effect of mobility on localization and time-synchronization. While in terrestrial systems, this effect is negligible because the speed at which communication occurs is much higher than the speed of devices; underwater acoustic communication is 5 orders of magnitude slower than radio. So, the effect of mobility can be much more pronounced underwater and needs to be studied in the context of localization and tracking.

1.4 Contributions

Our main contribution is system level solutions that make underwater localization and tracking more efficient for delay-tolerant applications. To this end, we have proposed optimal tracking for distributed systems, energy-efficient signaling schemes for gathering data critical to localization and the use of cross-layer design for time-synchronization.

Tracking and localization of mobile distributed systems have so far been treated as two different problems. Most existing work on collaborative localization is designed for static terrestrial networks and largely relies on heuristics to address this problem due to the highly non-linear relationship between data and unknowns. On the other hand, existing tracking techniques can at best allow for collaboration between nodes in a limited sense [Fox00] [Zha08]. We propose a generalized framework based on graphical inference that can solve both these problems and show that a number of existing techniques can be cast as special cases of this solution. We only require a

probabilistic characterization of measurement errors which can be obtained with only two devices and does not require prior information from a full-deployment.

In addition, through a combination of analysis and simulations, we have studied the role of mobility in the context of localization, tracking and time-synchronization. Interestingly we have shown that mobility can have both an adverse and beneficial effect on localization and tracking. We have analyzed and quantified this effect and have proposed powerful estimation techniques that use collaboration and cross-layer design to counter the adverse effects of mobility.

While existing localization techniques can be applied to the systems we are studying, with restrictions on device size, cost and energy, collaborative methods proposed by us become more suitable. We believe that our techniques in combination with existing systems would address the localization requirements of a wide range of applications that use multiple mobile platforms.

1.5 Dissertation Outline

Collaborative localization relies on relative information between nodes, specifically inter-node distances. So, we first look at two basic approaches to estimating inter-node distances: a traditional approach and another based on broadcast. In Chapter 2, we will discuss these two strategies, showing the need for time-synchronization for energy-efficient ranging. In Chapter 3, we study time-synchronization in depth. We highlight how node mobility deteriorates time-sync performance and propose to counter it using cross-layer design. In Chapter 4 we introduce our estimation framework based on factor-graphs that can optimally and jointly estimate the trajectories of multiple nodes using inter-node distance

measurements along with measurements of nodes' motion. In Chapter 5 we use our factor-graph framework to address the problem of mobility for systems that can only measure inter-node distances and further propose energy-optimizations for these systems. An account of related work is given at the end of each individual chapter.

CHAPTER 2

RANGING IN DISTRIBUTED SYSTEMS

2.1 Introduction

In Chapter 1, we outlined our overall objective where we intend to track a distributed system as a collective as opposed to tracking each device separately. This can be done when devices communicate with each other and in doing so establish relationships about their relative positions. Once these inter-node relationships are available we can in principle use all the measurements obtained by each node locally (about its velocity, acceleration, heading and so on) to jointly estimate the positions of all nodes in the collective. In this chapter we will talk about how we can obtain crucial relative information in the form of inter-node distances, for distributed systems. In principle relative information can also be angle measurements with neighbor nodes but this requires arrays of sensors which are often too bulky for mobile vehicles. While we can incorporate such information within our position estimation framework, we will focus on the most easy and common type of inter device information, namely, an estimate of the inter-node distances. The process of obtaining these distance measurements is generally referred to as *ranging*.

Ranging is possible by equipping each device with an acoustic modem. Due to the relatively slow speed of sound, acoustic communication lends itself naturally to derive distance from time-of-flight (TOF). Time-of-flight is an estimate of the time it takes for an acoustic signal to travel from a transmitting node to the node that receives

the signal. Since this signal travels at the speed of sound underwater, an estimate of the distance between the two nodes can be obtained from first principles: TOF is estimated by time-stamping the send time of the message and recording the time at which it is received (according to the local clocks of nodes). If both nodes had the same notion of time, ranging would be a very simple operation. However, no two clock-crystals are identical. They normally tick at slightly different rates, a phenomenon commonly referred to as clock-drift. As a result nodes go out of sync as time elapses. Time-synchronization is required to re-establish a common notion of time between any two nodes. In this chapter we will discuss how the accuracy of time-of-flight based ranging is related to the level of relative time-synchronization between nodes. In addition, we will introduce abstractions that will allow us to compare the suitability of different time-sync protocols for ranging and localization. Time-synchronization is explored in-depth in the next chapter.

Due the mobility of the system, distance-estimation has to be performed periodically. This is illustrated in Figure 2.1. Since devices are battery operated, *energy efficiency* is a crucial constraint. Traditional location estimation techniques are designed for stand-alone systems consisting of a few devices. In such systems, devices are localized independently. Consequently, ranging functionalities that are part of the typical modem software operate in a link-centric way, i.e. for an individual device the process of ranging is repeated between a node and each of its neighbors separately [Fre05a],[Fre05b]. As devices are added to the system, ranging has to be repeated between each node pair. Therefore, the energy consumed by collecting distance estimates in this way doesn't scale well with the number of devices. This motivates us

to rethink the way distance-estimation is performed underwater for large networked systems.

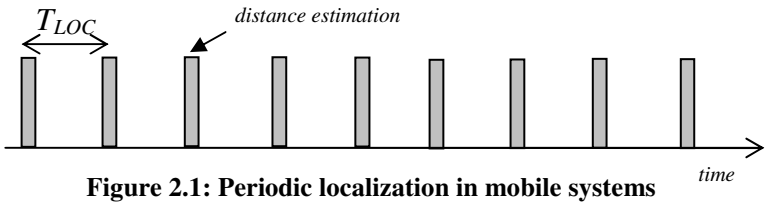


Figure 2.1: Periodic localization in mobile systems

To come up with a better approach, we need to first understand what exactly the cost of ranging is. To achieve this, we have experimentally characterized the energy consumed in communication for an off-the-shelf acoustic modem. We have also characterized the intrinsic sources of error. These characterizations will set the stage to better understand the severity of the problems related to energy consumption and performance of ranging which we will address in subsequent sections.

2.2 Experimental Characterization

The power consumption characteristics of underwater acoustic modems are widely different from terrestrial radio systems. This affects the way ranging must be performed in distributed systems as well as how other important protocols such as time-synchronization and MAC must be designed. We have measured the power consumed in signaling underwater for the WHOI micro-modem which is short to medium range (~ 1km). Figure 2.2 shows the power profile obtained during send and receive of a periodic, bi-directional ping exchange, illustrating the high power consumption for transmission compared to receive and idle modes. Therefore, the amount of signaling needed for ranging can significantly diminish the energy budget of nodes. The power numbers for the micro-modem along with other parameters are

summarized in Table 2.1. Using these parameters, we will later obtain the energy consumed for traditional roundtrip ranging as a function of network density (Section 2.3).

Table 2.1: Parameters of the Micro-modem

Data rate R_{data}	80 bps
Transmit power P_{Tx}	35 W
Receive power P_{Rx}	0.3 W
Message length, L	10 bytes
Transmission range R	≈1000 m

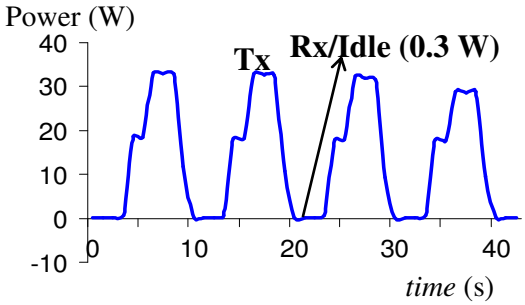


Figure 2.2: Measurement of power consumption of the micro-modem

Figure 2.3: Set up of experiments in Mission Bay, San Diego, CA

In addition, we characterized the ranging performance of the micro-modem by conducting two sets of experiments with a modem pair in Mission Bay, San Diego. Our experimental set up is as in Figure 2.3, showing the locations where the measurements were obtained, in the bay and by the dock.

In a first set of experiments conducted in the bay, one of the modems was kept stationary and the other was moved to different anchored buoys as shown in Figure 2.3. At each distance, the modems exchanged a number of bi-directional ping messages, which we repeated for different orientations of the transducers. Estimates of the true distances between buoys were obtained from GPS fixes. Figure 2.4 shows the error in estimated distances from these experiments, calculated as the standard deviation of measurements from the true distances. Here we have captured the cumulative effect of *intrinsic* errors on the ranging-performance. These errors are due to uncertainties in send and receive time, access time, interrupt handling, byte alignment and transmission and reception time. We estimated the speed of sound empirically by measuring temperature. Error in our estimate of the speed of sound is part of range estimates. Since Mission Bay is relatively shallow, our results also include the effect of multi-path on the performance of the modem. The ranging error we obtained is within GPS accuracy. Further, the performance does not vary much with the orientation of the transducers.

We also performed experiments at the dock in Mission Bay. Here we measured the exact distances between the modems. While in the first set of experiments (described earlier) the modems were subject to the small motion of the buoys, at the

dock they were completely static. This is reflected in the error distribution of the measurements shown in Figure 2.5.

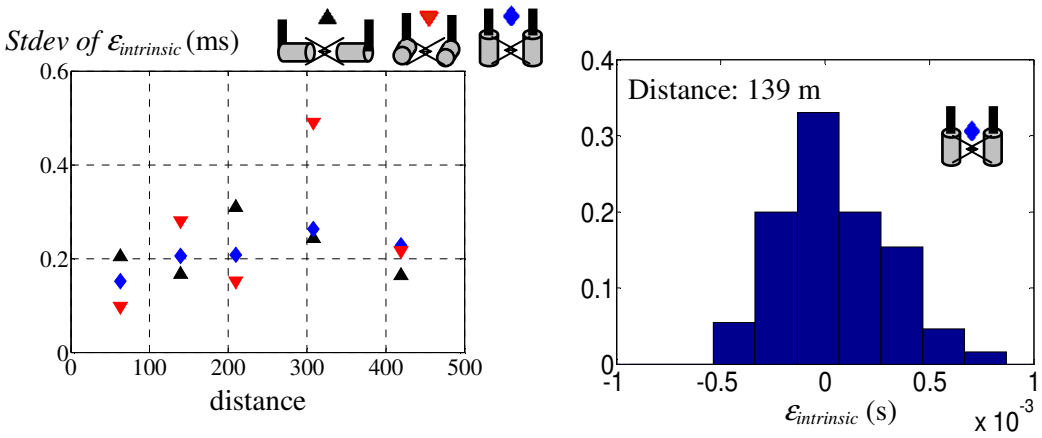


Figure 2.4: Ranging performance from experiments in Mission Bay

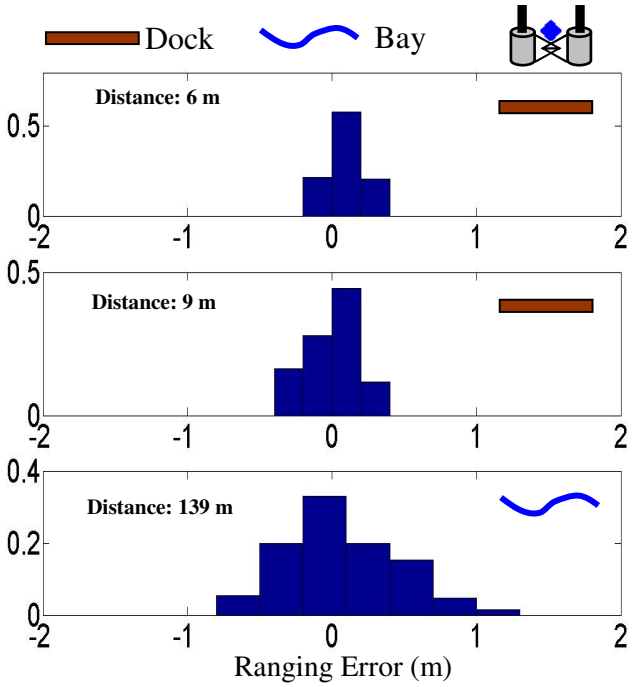


Figure 2.5: Error distribution of range measurements.

In general our results show that the errors intrinsic to the modem are small (of the order of a meter). However, as we will see, the effect of other factors such as

time-synchronization, node motion and delays during channel access can potentially have a larger effect on the ranging performance. We will analyze these effects in depth in the next chapter. The experimental results presented above will be referred to in later sections and in other chapters.

In our experimental results we have observed that the energy consumed to transmit is fairly high. Next we investigate the performance and energy-efficiency of two basic ranging schemes: a traditional link-based method where time-of-flight is determined from a bi-directional message exchange; and an approach based on broadcast. We will first investigate the traditional approach to ranging via bi-directional message exchange, a bit more closely in the next subsection, before detailing the principles behind broadcast ranging in the following subsection.

2.3 Performance Analysis of Link-Based Ranging

In general, distance between any two nodes can be deduced by measuring the time-of-flight of a packet exchange between them. Suppose that a message was sent by node **A** at time t_1 and received by node **B** at time t_2. With knowledge of the speed of sound c, the distance between the position of node **A** at time t_1, $P_A(t_1)$ and the position of node **B** at time t_2,$P_B(t_2)$ which is denoted by $d_{AB}(t_1,t_2)$, can be found as:

$$\left\| P_A(t_1) - P_B(t_2) \right\| = d_{AB}(t_1,t_2) \ = c \cdot (t_2 - t_1) \qquad (2.1)$$

Where t_1 and t_2 are the send and receive times of a message according to a global clock, respectively. However, in practice, devices do not know the exact time *t1* and *t2*. Instead they only have a local notion of time which is measured by counting ticks generated by their clock crystal. Since no two crystals are exactly alike, the rate

at which they tick is also different. Therefore, over time, the local time measured by each device will deviate from the absolute global time, a phenomenon known as clock drift. This drift, which we denote by δ. (δ - 1) is measured in ppm (parts per million). Due to the cumulative effect of clock drift, at any point in time, t, the local time of a node is offset from the global time. We denote this offset by $\Gamma(t)$.

Now, let's reconsider the message sent from node **A** to node **B**. The send and receive times of the message at nodes **A** and **B** according to their local time is denoted by $t_A(t_1)$ and $t_B(t_2)$ respectively. Further for any node **A**, $\Gamma_A(t_0)$ is the clock offset at time t_0 with respect to a global clock and δ_A is the clock skew. The local time of any node **A** at time t is related to the true global time as:

$$t_A(t) = \delta_A \cdot t + \Gamma_A \tag{2.2}$$

Therefore the send and receive times as recorded by nodes are related to the global times, t_1 and t_2 as:

$$\begin{aligned} t_A(t_1) &= t_0 + \Gamma_A(t_0) + \delta_A \cdot (t_1 - t_0) \\ t_B(t_2) &= t_0 + \Gamma_B(t_0) + \delta_B \cdot (t_2 - t_0) \end{aligned} \tag{2.3}$$

In link-based (also known as roundtrip) ranging, each node pair **A**-**B** executes a bi-directional ping: **A** first sends a time-stamped ping packet to **B**, immediately followed by a response back from **B** to **A**. By averaging the time-of-flight measurements, the clock offset is effectively canceled out. This strategy is, for example, implemented in the standard software of the micro-modem. The estimated distance, neglecting the effect of clock drift during the message exchange, is given by:

$$\hat{d}_{AB}^{RTRIP} = c \cdot \frac{\left[t_B(t_2) - t_A(t_1)\right] + \left[t_A(t_4) - t_B(t_3)\right]}{2} \qquad (2.4)$$

Where t_3 and t_4 are respectively the global times at which the reply message was sent by node **B** and received at node **A**. However, the above equation is only an approximation. Equation (2.4) can be written exactly as the average of the distances $d_{AB}(t_1,t_2)$ and $d_{BA}(t_3,t_4)$ plus an error term because of neglecting the effect of clock skew during the message exchange. We obtain this in equation (2.5) by substituting for local times $t_A(t_1)$, $t_B(t_2)$, $t_A(t_4)$, $t_B(t_3)$ in equation (2.4) as per equation (2.3):

$$\begin{aligned}
& c \cdot \frac{\left[t_B(t_2) - t_A(t_1)\right] + \left[t_A(t_4) - t_B(t_3)\right]}{2} \\
&= \frac{c}{2} \cdot \left[\Gamma_B(t_0) - \Gamma_A(t_0) + \delta_B \cdot (t_2 - t_0) - \delta_A \cdot (t_1 - t_0)\right] + \\
&\quad \frac{c}{2} \cdot \left[\Gamma_A(t_0) - \Gamma_B(t_0) + \delta_A \cdot (t_4 - t_0) - \delta_B \cdot (t_3 - t_0)\right] \\
&= \frac{c}{2} \cdot \left[\delta_B \cdot (t_2 - t_3) + \delta_A \cdot (t_4 - t_1)\right] \\
&= \frac{d_{AB}(t_1,t_2) + d_{BA}(t_3,t_4)}{2} + \tilde{\varepsilon}_{RTRIP}
\end{aligned} \qquad (2.5)$$

Where, $\tilde{\varepsilon}_{RTRIP} = \frac{c}{2} \cdot \left[(\delta_A - 1) \cdot (t_4 - t_1) - (\delta_B - 1) \cdot (t_3 - t_2)\right]$

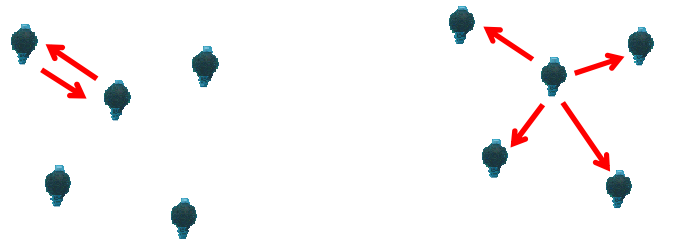
From equation (2.4) and equation (2.5), we observe that the distance-estimate from the roundtrip approach does not correspond to the distance between nodes at any one time instance. There is an ambiguity because of node mobility and the effect of clock skew during the message exchange. We will explain these effects more in the next chapter when we discuss synchronization using the same request and reply message exchange.

Since link-based ranging relies on a bi-directional message exchange, in a networked setting, a ping-exchange would have to be performed on each link individually. We assume that nodes overhear messages transmitted by neighbor nodes[2]. So, for a network where nodes have on average λ neighbors, each transmission results in λ message receptions. The total energy consumed as a result of each transmitted message is the transmit energy plus λ times the receive energy. Therefore, the energy consumed per node per distance-estimation event is:

$$E_{RTRIP} = \lambda \cdot \left(E_{Tx} + \lambda \cdot E_{Rx} \right) \tag{2.6}$$

Where, E_{Tx} is the energy required to transmit a ping message,

E_{Rx} is the energy required for message reception.



(a) Roundtrip ranging **(b) Broadcast ranging**

Figure 2.6: Comparison of two basic ranging strategies

Since nodes communicate over a shared medium, they can reach all their neighbors via a single broadcast message. Broadcast ranging takes advantage of the fact that multiple nodes need to estimate distances at around the same time. Consequently a single message is used to simultaneously deduce time-of-flight (TOF) to neighbor nodes, see Figure 2.6(b). This is as opposed to link-based ranging where

[2] Even if nodes were not overhearing, this result holds because the energy consumed to transmit is dominant in underwater acoustic communication.

distance between each node pair is estimated separately, see Figure 2.6(a). We will explore broadcast ranging more in the next section.

2.4 Network Centric Ranging

We propose an alternative to ranging which uses a broadcast message instead of a ping exchange. Here the problem of ranging is addressed from a network perspective by allowing multiple nodes to estimate their distance to a neighbor node using a single broadcast. But nodes have to be explicitly synchronized at a prior time. Inter-node distances can then be estimated from a single broadcast message rather than having to combine two messages (one in each direction) as in link-based ranging.

We denote the relative clock offset, say between two nodes *A* and *B,* by $\Psi = \Gamma_B - \Gamma_A$ and the relative clock drift $\gamma_{BA} = (\delta_B - \delta_A)$ The process of determining the offset Ψ and/or the relative clock drift γ_{BA} between node pairs is known as time-synchronization. Suppose that nodes were synchronized at time t_0, where only the clock offset was estimated, denoted by $\hat{\Psi}(t_0)$. Then, the inter-node distance estimate, $\hat{d}_{AB}(t_1)$ can be obtained from the one-way time of arrival of a broadcast message as:

$$\begin{aligned}
&\hat{d}_{AB}(t_1,t_1) = c\cdot\left[t_B(t_2) - t_A(t_1) - \hat{\Psi}(t_0)\right] \\
&= c\cdot\left[\Psi(t_0) + \delta_B\cdot(t_2 - t_0) - \delta_A\cdot(t_1 - t_0) - \hat{\Psi}(t_0)\right] \\
&= c\cdot\left[\delta_B\cdot(t_2 - t_1) + (\delta_B - \delta_A)\cdot(t_1 - t_0) + \Psi(t_0) - \hat{\Psi}(t_0)\right] \\
&= c\cdot(t_2 - t_1) + c\cdot\left[(\delta_B - 1)\cdot(t_2 - t_1) + (\delta_B - \delta_A)\cdot(t_1 - t_0) + \Psi(t_0) - \hat{\Psi}(t_0)\right] \\
&= d_{AB}(t_1,t_2) + c\cdot\left[(\delta_B - 1)\cdot(t_2 - t_1) + (\delta_B - \delta_A)\cdot(t_1 - t_0) + \varepsilon_\psi\right] \\
&= d_{AB}(t_1,t_1) + c\cdot\left[(\delta_B - 1)\cdot(t_2 - t_1) + \gamma_{BA}\cdot(t_1 - t_0) + \varepsilon_\psi\right] + \varepsilon_{motion}
\end{aligned} \tag{2.7}$$

Where, ε_ψ is the error in the offset estimate $\psi(t_0)$.

ε_{motion} is the error because of the motion of node **B** during the interval $|t_2 - t_1|$ and is upper-bounded as follows:

$$\left|\varepsilon_{motion}\right| = \left|d_{AB}(t_1,t_2) - d_{AB}(t_1,t_1)\right| \le v_B \cdot \frac{R}{c} \tag{2.8}$$

Where, v_B is the speed of node **B** and R is the maximum transmission range.

So, for the case where only the clock offset is estimated by the synchronization protocol, the error in distance estimation from a one-way message is given by:

$$\begin{aligned}\varepsilon_D^{(1)} &= \hat{d}_{AB}(t_1,t_1) - d_{AB}(t_1,t_1) \\ &= (\delta_B - 1)\cdot d_{AB}(t_1,t_2) + c\cdot\left[\gamma_{BA}\cdot(t_1 - t_0) + \varepsilon_\psi\right] + \varepsilon_{motion} \\ &= (\delta_B - 1)\cdot d_{AB}(t_1,t_1) + c\cdot\left[\gamma_{BA}\cdot(t_1 - t_0) + \varepsilon_\psi\right] + \delta_B\cdot\varepsilon_{motion}\end{aligned} \tag{2.9}$$

Suppose the synchronization algorithm also estimate the relative clock drift of node **B,** denoted by $\hat{\gamma}_{BA}$. In this case, the inter-node distance estimate at time t_1 can be obtained from the one-way time of arrival of a broadcast message as:

$$\begin{aligned}&\hat{d}_{AB}(t_1) = c\cdot\left[t_B(t_2) - t_A(t_1) - \hat{\Psi}(t_0) - \left(\hat{\gamma}_{BA}\cdot(t_1 - t_0)\right)\right] \\ &= d_{AB}(t_1,t_1) + (\delta_B - 1)\cdot d_{AB}(t_1,t_2) + c\cdot\left[(\gamma_{BA} - \hat{\gamma}_{BA})\cdot(t_1 - t_0) + \varepsilon_\psi\right] + \varepsilon_{motion}\end{aligned} \tag{2.10}$$

Following equation (2.10), we can obtain the error in the distance estimate when the synchronization protocol estimates both the relative clock offset and skew as follows:

$$\begin{aligned}\varepsilon_D^{(2)} &= \hat{d}_{AB}(t_1,t_1) - d_{AB}(t_1,t_1) \\ &= (\delta_B - 1)\cdot d_{AB}(t_1,t_1) + c\cdot\left[\varepsilon_\delta\cdot(t_1 - t_0) + \varepsilon_\psi\right] + \delta_B\cdot\varepsilon_{motion}\end{aligned} \tag{2.11}$$

Equations (2.9) and (2.11) can be summarized as:

$$\varepsilon_D = (\delta_B - 1)\cdot d_{AB}(t_1,t_1) + c\cdot\left[\varepsilon_\gamma\cdot(t_1 - t_0) + \varepsilon_\psi\right] + \delta_B\cdot\varepsilon_{motion} \tag{2.12}$$

Where, ε_γ is the relative clock drift between nodes when only the clock offset is estimated during synchronization and it is the error in the relative drift when it is estimated as well.

The distance estimates are most accurate immediately after synchronization, at time t_0. From equation (2.12) and (2.8), the error in distance estimate, $\varepsilon_D(t)$ will grow in time:

$$\begin{aligned}\varepsilon_D(t) &= \varepsilon_D(\Delta t + t_0) = \hat{d}_{AB}(t,t) - d_{AB}(t,t) \\ &= (\delta_B - 1)\cdot d_{AB}(t,t) + c\cdot\left[\varepsilon_\gamma\cdot\Delta t + \varepsilon_\psi\right] + \delta_B\cdot\varepsilon_{motion}\end{aligned} \tag{2.13}$$

Where, Δt is the time elapsed since the last synchronization at time t_0.

Based on equation (2.13), we can find the upper bound for the magnitude of the distance-error as:

$$\begin{aligned}\left|\varepsilon_D(t)\right| &= \left|(\delta_B - 1)\cdot d_{AB}(t,t) + c\cdot\left[\varepsilon_\gamma\cdot\Delta t + \varepsilon_\psi\right] + \delta_B\cdot\varepsilon_{motion}\right| \\ &\le \left|(\delta_B - 1)\right|\cdot d_{AB}(t,t) + c\cdot\left[\left|\varepsilon_\gamma\right|\cdot\Delta t + \left|\varepsilon_\psi\right|\right] + \left|\delta_B\right|\cdot\left|\varepsilon_{motion}\right| \\ &\le c\cdot\left[\left|\varepsilon_\gamma\right|\cdot\Delta t + \left|\varepsilon_\psi\right|\right] + R\cdot\left[\left|\delta_B - 1\right| + \left|\delta_B\right|\cdot\frac{v_B}{c}\right] \\ &= c\cdot\left[\left|\varepsilon_\gamma\right|\cdot\Delta t + \left|\varepsilon_\psi\right|\right] + \chi_{drift}\end{aligned} \tag{2.14}$$

$$\chi_{drift} = R\cdot\left[\left|\delta_B - 1\right| + \left|\delta_B\right|\cdot\frac{v_B}{c}\right] \tag{2.15}$$

To limit the ranging error, synchronization has to be repeated periodically in addition to periodic localization as depicted in Figure 2.7. Here, the time elapsed between synchronization events is indicated as T_{SYNC}. As per equation (2.13) the error in ranging, $\varepsilon_D(t)$ grows with T_{SYNC}. Therefore, the synchronization interval is chosen to meet an acceptable limit in the error of distance estimates.

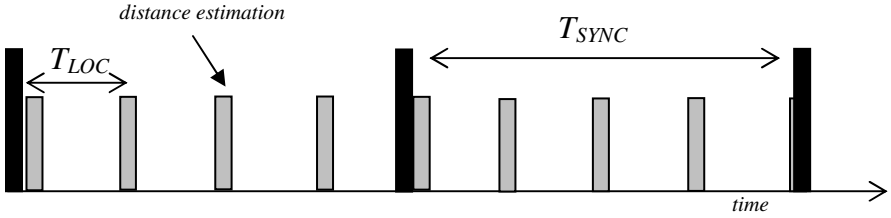


Figure 2.7: Periodic synchronization for broadcast-based localization

For a given drift or drift-estimate error, ε_γ, T_{SYNC} is determined from equation (2.14) for a target ranging error, $\varepsilon_D^{\max}$ as:

$$T_{SYNC} \geq \frac{1}{|\varepsilon_\gamma|} \cdot \left(\frac{1}{c} \cdot \left(\varepsilon_D^{\max} - \chi_{drift} \right) - |\varepsilon_\Psi| \right) \tag{2.16}$$

Now, if the system is such that nodes communicate only to localize, energy would primarily be consumed to obtain range estimates periodically as depicted in Figure 2.1. In this case, the overhead of performing synchronization for broadcast ranging should be taken into account when comparing link-based and broadcast schemes. For broadcast ranging, the average energy consumed per node per localization is given by equation (2.17).

$$E_{BCAST} = \left(E_{Tx} + \lambda \cdot E_{Rx} \right) + \alpha \cdot E_{SYNC} \tag{2.17}$$

$$\alpha = \left\{ \begin{array}{ll} \dfrac{T_{LOC}}{T_{SYNC}}, & \text{for } T_{LOC} \leq T_{SYNC} \\ 1, & \text{otherwise} \end{array} \right\} \tag{2.18}$$

Here, E_{SYNC} is the energy consumed per node for synchronization. The parameter α is the number of localization events per synchronization. If the synchronization interval required to remain within an error bound is less than a

localization period, nodes synchronize only once, right before range estimates are obtained. In this case $\alpha = 1$. Now, based on our analysis so far let us compare the performance of broadcast and link-based ranging.

2.5 Choosing a Ranging Scheme: Performance Comparison

To compare the two ranging alternatives, we express the relative energy performance of a broadcast compared to a link-based scheme by combining equations (2.6) and (2.17):

$$\frac{E_{BCAST}}{E_{RTRIP}} = \frac{1}{\lambda} \cdot (1 + \frac{\alpha \cdot E_{SYNC}}{E_{Tx} + \lambda \cdot E_{Rx}}) = \frac{1}{\lambda} \cdot \left(1 + \alpha \cdot N_{SYNC}\right) \qquad (2.19)$$

Here, N_{SYNC} is the number of messages transmitted per node for synchronization and depends on the protocol used. We will discuss different approaches to time-synchronization in the next chapter where we will look at both existing methods and enhancements proposed by us. However, as an example we will use TPSN (Timing-sync Protocol for Sensor Networks) [Gan03] to compare the performance of roundtrip and broadcast ranging.

TPSN requires two message exchanges between each node pair for synchronization. Therefore, for a network where nodes have an average of λ neighbors, the number of messages transmitted per node for synchronization, $N_{SYNC} = \lambda$. While this signaling overhead can be further reduced if a hierarchy or a spanning tree had been previously established, we will consider the base protocol. Consequently, the energy consumption of broadcast based ranging relative to roundtrip can be obtained from equation (2.20) as:

$$\left(\frac{E_{BCAST}}{E_{RTRIP}}\right)_{TPSN} = \alpha + \frac{1}{\lambda} \tag{2.20}$$

As per equation (2.20), the relative performance of broadcast and roundtrip ranging is determined by the network density λ and the parameter α. In general λ should be greater than 10 for a connected network ($1/\lambda < 0.1$) [Bet02] and $\alpha \leq 1$ (as per equation (2.18)). Since TPSN does not estimate the clock drift, the error in drift will be the actual relative clock drift. Since a full characterization of TPSN is not available we will assume that the offset estimate is perfect right after synchronization. By combining equations (2.18) and (2.16), α for broadcast ranging can be expressed in terms of clock drift, the localization period and the target ranging error as:

$$\alpha = \min\left(\frac{\gamma \cdot T_{LOC} \cdot c}{\varepsilon_D^{\max} - \chi_{drift}}, 1\right) \tag{2.21}$$

The period at which localization should be repeated is a function of the dynamics of the system, since with increased mobility node positions have to be estimated more often. Therefore, for a target ranging accuracy, the combined effect of expected mobility and the accuracy of the clock crystal used by nodes decides the preferred ranging choice. Figure 2.8 shows the variation in α as a function of the product of γ (in ppm) and T_{LOC} (in seconds) for different target ranging errors, $\varepsilon_D^{\max}$.

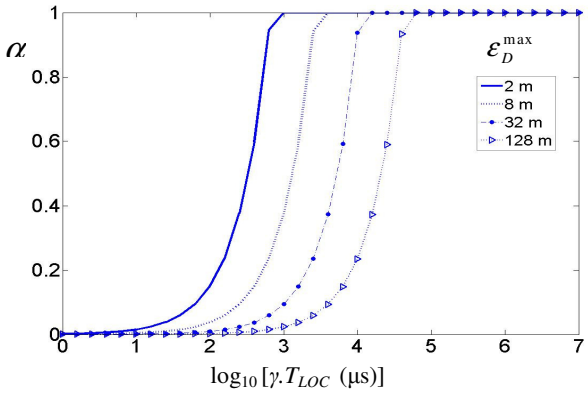


Figure 2.8: Combined effect of localization period and clock drift on the parameter α

The overall relative energy consumption of broadcast ranging vs. roundtrip is obtained by combining the results of Figure 2.8 and Figure 2.9. For the broadcast approach, the parameter α was determined as a function of various system parameters in Figure 2.8. Figure 2.9 shows the relative energy consumption of broadcast ranging vs. roundtrip as a function of network density and the parameter α. Here we observe that as long as α is below 0.9, broadcast ranging outperforms link-based ranging for all network densities.

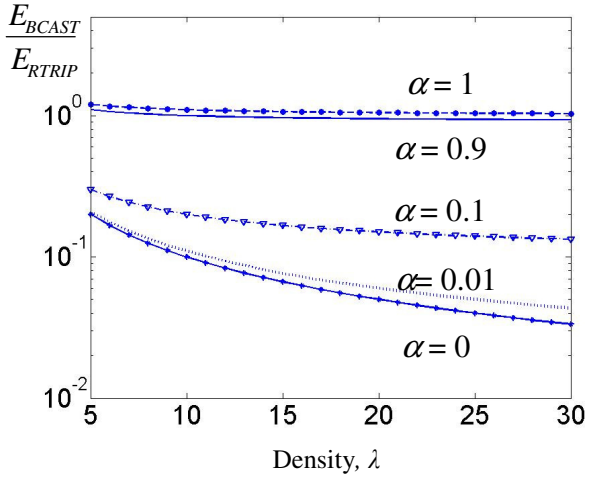


Figure 2.9: Relative energy consumption of broadcast ranging vs. roundtrip with network density and parameter, α

As shown earlier, the preferred ranging scheme and expected gains from it can be determined using equations (2.19) and (2.21) for any specific system. As per Figure 2.9 the gains from broadcast ranging can be substantial. With newly available clocks that are both cheap and precise [Eus06] [Sch08], very low values of α can be achieved, making broadcast ranging a much more energy-efficient option. As an example, Eustice et al have reported integrating a highly accurate temperate-compensated clock into the micro-modem, with a drift of only 0.02 ppm [Eus06]. Similar levels of accuracy have also been achieved through a novel and inexpensive approach using two crystals [Sch08]. However, if such accurate clocks are not available, nodes have to explicitly time-synchronized which is the topic of next chapter.

2.6 Conclusion

With an increased interest in understanding underwater processes via 3D sampling, future deployments are expected to be large and dense. Estimating positions of devices is an integral part of data-collection in such systems. Traditional link-based ranging does not scale well as we move to dense deployments. This is because the per node energy consumption compounds with the number of links. In our analysis, we contrast link-based ranging to a broadcast approach. We further quantify the accuracy of time-synchronization to the performance of ranging in underwater networks and provide a framework that can determine the preferred ranging choice given the specifics of any particular system.

2.7 Acknowledgments

This chapter is, in part, a reprint of material published in “"Design of Self-Localization Strategies for Networks of Underwater Drifters”, that appeared in the Proceedings of MTS/IEEE Oceans’08. In all cases, I was the primary researcher and author and Curt Schurgers supervised the research. I would like to thank our collaborator Jules Jaffe at Scripps, Robert Glatts, Bridget Benson and Shoubhik Mukhopadhyay for their help with the experiments at Mission Bay.

CHAPTER 3

ENERGY-EFFICIENT TIME-SYNCHRONIZATION

3.1 Introduction

A number of important network functionalities require that nodes have a common notion of time. These include time stamping of events, distributed data aggregation, MAC and localization. However, the local clock of nodes has an intrinsic drift due to which nodes go out of sync as time elapses. Therefore, time synchronization protocols are crucial to any distributed system. Although the accuracy of time synchronization required for underwater networks is much lower than that for terrestrial networks due to the fact that communication takes place at a much coarser time-granularity, still, accurate time-synchronization can alleviate the need for frequent re-synchronizations, which saves energy. Therefore, precise time-synchronization, using minimal signaling, is of primary importance to underwater networks given that underwater devices are highly energy constrained.

All network time synchronization methods rely on some kind of message exchange between nodes to establish a common notion of time. However, non-determinism in the network dynamics such as medium access time, propagation time or interrupt handling time makes synchronization challenging [Siv04]. For sensor networks, MAC layer time stamping is exploited to significantly reduce the timing uncertainty, which leaves propagation delay as the single major source of error. As the distance between two nodes is small compared to the speed of RF signals, propagation delay is often negligible in terrestrial networks.

Unfortunately, traditional time sync protocols [Els02] [Gan03] [Mar04] developed for terrestrial sensor networks cannot be applied to underwater networks because of long and unknown propagation delays of acoustic signals [LuF09]. The propagation delay, if not estimated, translates into a timing uncertainty between a pair of synchronizing nodes. This problem becomes more prominent when nodes are mobile. Because the propagation delay changes over time, measurements obtained from time-stamped messages are no longer sufficient to accurately synchronize nodes. While a few time-sync protocols [Chir08] [Sye06] have been specifically designed for underwater networks, they are more suitable for networks with very limited mobility. In our work, we focus on mobile systems instead.

Since energy is a key design constraint, we will first introduce a light-weight synchronization protocol that only estimates the clock offset. It uses hardware enhancements, namely highly accurate clocks [Eus06] to compensate for the effect of drift. However, if such clocks are not available to nodes, they have to estimate their relative drift to avoid frequent resynchronizations. So, we will also explore full sync protocols where the clock offset and drift are estimates.

There are two major impediments to underwater synchronization, namely, large delays in propagation and medium access, and substantial node mobility during the synchronization process. While existing solutions have been proposed to address these challenges, they rely on heavy signaling, which is undesirable due to high energy costs. We introduce a powerful new approach that incorporates physical layer information, namely an estimate of the Doppler shift. Large Doppler shift has been identified as a major challenge to underwater communication, and current systems

implement sophisticated solutions to estimate and track such Doppler shift for each data exchange. While an impediment to communication, we will show that the Doppler shift contains highly useful information that can be leveraged to improve time synchronization. Specifically, it provides an indication of the relative motion between nodes. Our protocol, called D-sync, strategically exploits this feature to address the timing uncertainty due to node mobility. As such, D-sync can handle substantial mobility, without making any assumptions about the underlying motion, and without extensive signaling. Simulation results show that D-sync significantly outperforms existing time synchronization both in terms of accuracy and energy. We begin by formally introducing the problem of time-sync.

3.2 Problem Definition

To formally define the time-sync problem, we will revisit the equations introduced in Chapter 2. Let t denote global time. For any node **A**, $\Gamma_A(t_0)$ is the clock offset at time t_0 with respect to the global clock and δ_A is the clock skew. The local time of any node **A** at time t is related to the true global time as:

$$t_A(t) = t_0 + \Gamma_A(t_0) + \delta_A \cdot (t - t_0) \qquad (3.1)$$

Global time-synchronization is the process of estimating the clock offset $\Gamma_A(t_0)$ and the skew δ_A. We can write equation (3.1) for another node **B**. Relative time sync implies estimating the relative clock offsets $\psi_{BA}(t_0) = \Gamma_B(t_0) - \Gamma_A(t_0)$ and the relative clock skew. In the next section we will introduce a synchronization scheme that only estimates the relative clock offset between nodes.

3.3 Post-facto Joint Ranging & Synchronization (SDME-S)

In Section 2.2, we showed via experimental measurements that transmitting data even with short to medium range acoustic modems consume considerable power. To address this problem, we propose a light-weight protocol for estimating clock offsets, which we have dubbed SDME-S (Sufficient Distance Map Estimation-Synchronization). The key idea is that nodes exchange only enough information to be able to be synchronized at a later time when the data collected by all nodes is available centrally. This is in line with our post-facto strategy of localizing nodes and so, SDME-S is designed for post-facto ranging and localization. This scheme is also built on the realization that extremely accurate clocks have become available. For example, Eustice et al have reported integrating one such temperate-compensated clock into the micro-modem, with a skew of only 0.02 ppm [Eus06]. Similar levels of accuracy have also been achieved through a novel and inexpensive approach of using two crystals [Sch08].

3.3.1 Protocol Description

The goal of SDME-S is to collect the data needed to estimate the clock offsets between each pair of nodes post-mission. As such it is more a synchronization-data-collection algorithm than a time-synchronization or ranging algorithm. We will see that the signaling of SDME-S is such the inter-node distances can also be estimated during the sync process.

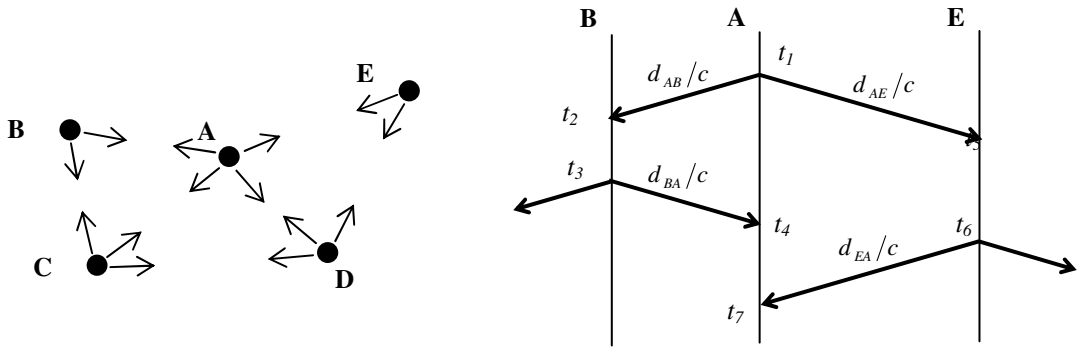


Figure 3.1: SDME-S: Synchronization-data collection using broadcasts

The operation of SDME-S is illustrated in Figure 3.1. Essentially, each node only broadcasts once, where a broadcast message contains a time stamp of when it was sent (according to the local clock of the node). Each receiving node stores this time stamp in addition to the time it received this packet. With this information, the clock offset estimates for each node pair, $\hat{\Psi}$ can be calculated post-mission from (3.2). Here, we neglected the effect of the clock skew in (3.1), since all times t_i are very close to the start of synchronization process t_0. We have also neglected the effect of node movement during the message exchange. We will study the impact of these simplifications in the next subsection.

$$\frac{\left[t_B(t_2)-t_A(t_1)\right]-\left[t_A(t_4)-t_B(t_3)\right]}{2} \cong \frac{1}{2}\cdot\left[\frac{d_{BA}}{c}+\Gamma_B-\Gamma_A\right]-\frac{1}{2}\cdot\left[\frac{d_{AB}}{c}+\Gamma_A-\Gamma_B\right]$$
$$\cong \Gamma_B-\Gamma_A=\hat{\Psi} \qquad (3.2)$$

One should note that after every node sends out a broadcast message containing the send time of the message, only two time stamps i.e. either (t_1, t_2) or (t_3, t_4) are available on each node and no node has all four timestamps for any of its links. This is not an issue if the calculation of range information (and self-localization) is

done post-mission. However, if range information needs to be extracted while nodes are submerged, extra steps need to be taken to distribute the timestamps (e.g. each broadcast could also contain the receive times of all broadcasts already heard from neighbors).

By modifying the design goal from global synchronization to post-mission relative synchronization, SDME-S reduces the number of packets exchanged by a factor of 2 (compared to TPSN (Timing-sync protocol) [Gan03] and LTS (Lightweight time synchronization protocol) [Gre03]) or more in TSHL). For a network where each node has an average of λ neighbors, the resulting energy consumption per node per synchronization event is given by:

$$E_{SDME-S} = \left(E_{Tx} + \lambda \cdot E_{Rx}\right) \tag{3.3}$$

Also note that due to the relatively slow speed of sound, we only require clock offsets to be estimated with an accuracy of about 1 ms, which corresponds to a range error of 1.5 m. Although not for ranging, nodes do need some idea of the global time while submerged, for example to make sure they gather ranging data around the same time or to support duty cycling. This, however, does not require very accurate timing, and a global time accuracy of around 1 second is sufficient. With a clock skew of 0.02 ppm and synchronization with GPS at the start of the mission, clocks remain within this limit for more than 1 year. However, if such hardware enhancements are not possible, precise time-synchronization protocols are required which is the topic of discussion in Section 3.4.

We observe that from the timing-information obtained above, the inter-node

distances can also be estimated using the same equations as link-based ranging given by equation (2.4). However, the way in which these timing measurements are obtained is different in SDME-S compared to the link based.

Although functionally equivalent to roundtrip ranging, the accuracy of SDME-S is affected by the increased time delay between messages. Unlike the roundtrip ping message (where the initial message is followed immediately by a response), the two broadcast messages that are combined to provide the same information are typically spaced in time. To avoid collisions, nodes should select different times to broadcast. As a single packet transmission may take a second or more due to low data rates (e.g. 80 bps for the WHOI micromodem, see Table 2.1), the time over which broadcasts need to be spaced can easily be as large as 200s, also depending on node density. During this time, clocks drift apart, but this effect is minor if accurate clocks are used (e.g. with a 1 ppm crystal, the drift would be 0.2 ms, or a 0.3m ranging error). However, node mobility may be a bigger issue. With relative speeds of only 0.1 m/s, nodes move up to 20m between broadcasts, causing this amount of error in range estimates. From our modem characterization, we see that this mobility induced effect is dominant compared to the intrinsic errors, which are in the order of a meter, (see Figures 2.4 and 2.5). Whether this error can be tolerated depends on the system. In short-range systems especially, it may be an issue. We tackle the problem of mobility induced error in Section 3.4.

3.3.2 Modeling Synchronization Accuracy

Next, we will evaluate the performance of SDME-S in terms of accuracy and start with its ability to estimate offsets. When we do not neglect clock skew and node

mobility during message exchanges, the perceived time of flight obtained from equation (3.1) is given by:

$$t_B(t_2) - t_A(t_1) = (t_2 - t_1)\cdot\delta_A - (\Gamma_A(t_0) - \Gamma_B(t_0)) - (\delta_A - \delta_B)\cdot(t_2 - t_0)$$

$$t_A(t_4) - t_B(t_3) = (t_4 - t_3)\cdot\delta_A + (\Gamma_A(t_0) - \Gamma_B(t_0)) + (\delta_A - \delta_B)\cdot(t_3 - t_0) \qquad (3.4)$$

The offset estimation from equation (3.2) can be more exactly expressed as:

$$\frac{[t_B(t_2) - t_A(t_1)] - [t_A(t_4) - t_B(t_3)]}{2} = \hat{\Psi}(t_0)$$
$$= \Gamma_B(t_0) - \Gamma_A(t_0) + \varepsilon_{signaling} + \varepsilon_{\text{int}rinsic} \qquad (3.5)$$

We added $\varepsilon_{intrinsic}$ in this equation to include effects such as send and receive time, access time, interrupt handling, byte alignment, and transmission and reception time. They are common to all synchronization algorithms and we have characterized this error experimentally in Section 2.2 of Chapter 2. The unique effects of skew and propagation speed in combination with mobility are captured in $\varepsilon_{signaling}$, which is further analyzed in (3.6). The inequalities follow from the graphical representation of Figure 3.2.

$$\begin{aligned}
|\varepsilon_{signaling}| &= \left|\delta_A\cdot\left(\frac{t_2 - t_1 - t_4 + t_3}{2}\right) + (\delta_B - \delta_A)\cdot\left(\frac{t_2 + t_3 - 2t_0}{2}\right)\right| \\
&= \left|\delta_A\cdot\left(\frac{d_{AB}(t_1,t_2) - d_{BA}(t_3,t_4)}{2\cdot c}\right) + (\delta_B - \delta_A)\cdot\left(\frac{t_2 + t_3 - 2t_0}{2}\right)\right| \\
&\le |\delta_A|\cdot\left(\frac{|v_B|\cdot(t_2 - t_1) + |v_A|\cdot(t_4 - t_3) + |v_B - v_A|\cdot(t_3 - t_1)}{2\cdot c}\right) + |\delta_B - \delta_A|\cdot\left(\frac{t_2 + t_3 - 2\cdot t_0}{2}\right) \\
&\le \underbrace{\left[2\cdot|\delta_A|\cdot\frac{v_{\max}}{c}\cdot\frac{R}{c}\right]}_{|\varepsilon_{motion_flight}|} + \underbrace{\left[|\delta_A|\cdot\frac{v_{\max}}{c}\cdot T_{MAC}\right]}_{|\varepsilon_{motion_MAC}|} + \underbrace{|\delta_B - \delta_A|\cdot(t_D - t_0)}_{|\varepsilon_{drift_sync}|}
\end{aligned} \qquad (3.6)$$

Where, $d_{AB}(t_1,t_2)$ is the distance traveled by a message between nodes A and B during the interval (t_1, t_2). v_A is the speed of node A. v_{max} is the maximum speed of nodes. R is the transmission range. T_{MAC} is the maximum delay in channel access, $t_D = (t_2+t_3)/2$ is approximately the time of message exchange and (t_D-t_0) is time elapsed since the last synchronization time.

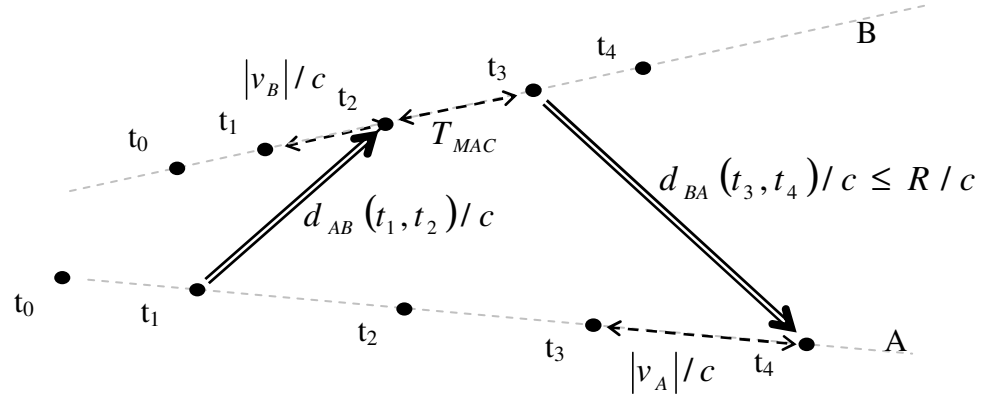


Figure 3.2: Effect of node mobility on the message exchange of SDME-S

In equation (3.6), we identify three distinct sources of error:

$\varepsilon_{motion\_flight}$: Error due to node mobility during the time of flight of the packet. This is bounded by the clock skew during the interval $(2.v_{max}.R)/c^2$, which is the maximum time of flight of a packet when nodes are mobile.

ε_{motion_MAC}: Error due to node mobility during the time between the broadcasts. To avoid excessive collisions, nodes need to space their broadcasts apart, e.g. by introducing a random back-off. The maximum change in node distances during this time is given by $(v_{max}.T_{MAC})$ which translates to a timing error, eq (10).

ε_{drift_sync}: Error due to clock drift during the time elapsed since the last synchronization, (t_D-t_0).

With the estimated offsets, the inter-node distances are found for each

localization step using the time-of-flight of a message. Since the network is localized for an instance in time t_L, distance estimates, $\hat{d}_{AB}^{t_L}$ are calculated with respect to this common time, equation (3.7). Using an analysis similar to the one in Figure 3.2, we compute this error as in equation (3.8).

$$\hat{d}_{AB}^{t_L} = c \cdot \left(t_B(t_2) - t_A(t_1) - \Psi(t_0) \right) + c \cdot \varepsilon \tag{3.7}$$

$$|\varepsilon| \le |\varepsilon_{\text{int}rinic}| + |\varepsilon_{signaling}| + \underbrace{\left[\left(\frac{|v_B|}{c} + |\delta_B - 1| \right) \cdot \frac{R}{c} + \left(\frac{|v_B - v_A|}{c} + |\delta_A - \delta_B| \right) \cdot |t_1 - t_L| \right]}_{|\varepsilon_{common\_time}|} + \underbrace{|\delta_A - \delta_B| \cdot (t_L - t_0)}_{|\varepsilon_{clock\_drift}|} \tag{3.8}$$

In this equation, $\varepsilon_{clock\_drift}$ is the error due to the clock drift between localization time t_L and the last synchronization time t_0. On the other hand, $\varepsilon_{common\_time}$ is caused by node mobility and the fact that the time of the message exchange for each link may deviate from the common localization time t_L again due to MAC back-offs. Finally, we also observe that the accuracy in distance estimation depends on precise knowledge of the speed of sound, c. However, accurate empirical models are available that predict c as a function of temperature, pressure and salinity, with a reported accuracy of 0.07 m/s [Dus93]. In our system, these parameters are being estimated by the drifter's sensors, and also stored during localization. From our analysis of SDME-S, we have shown that node mobility combined with delay in medium access is the major impediment to synchronization. We will next present a full time-sync protocol that estimates both the relative offset and clock drift while dealing with the problem of mobility.

3.4 Time Sync Using Physical Layer Inputs (D-Sync)

Two main factors contribute to the pronounced effect of mobility on the performance of underwater time synchronization. First, the distance between nodes is not a constant during the process of synchronization. It changes between message exchanges depending on the relative speed of nodes. So, while devices that are not propelled such as underwater drifters and gliders, have maximum speeds of up to 1m/s, their relative speed can be as high as 2m/s. Similarly, the relative speeds of self-propelled vehicles such as AUVs, can go up to 4 m/s. The second reason for the high impact of mobility is the considerable delay before nodes can transmit. This is because they need to randomly back off to avoid collisions which can be significant even with moderate number of contenders. With relative speeds of about 2m/s, nodes could drift tens of meters between transmissions. Given the speed of sound underwater, this translates to a timing uncertainty in the order of milliseconds. In this section we will specifically focus on designing a time-sync protocol that is robust to mobility by incorporating physical layer information, namely an estimate of the Doppler shift.

The Doppler shift due to node mobility is one of the major impairments to underwater communication. Consequently, estimating the Doppler shift and compensating for it is a well-studied problem [Sif08] [Joh97] [Lib08] [Mas08] [Par09] [Sha00]. It has been shown that the Doppler shift can be estimated from a single packet exchange between a pair of nodes without the use of any additional hardware [Sha00] [Lib08]. Mason et al. [Mas08] show that their Doppler estimation algorithm can estimate relative speeds up to 5m/s with standard deviation of 0.1 m/s using experimentally obtained data. Instead of estimating the Doppler information at the

packet level, Nathan et al. [Par09] proposed a symbol by symbol Doppler rate estimation method for highly mobile underwater systems. This can potentially give us an even better estimate of Doppler velocity. We believe that the Doppler shift estimate as provided by physical layer algorithms can be a crucial input for time synchronization. Our new protocol, called D-sync, strategically exploits this feature to address the timing uncertainty due to node mobility as discussed earlier. Further, the accuracy of D-Sync outperforms existing time synchronization protocols.

Among existing time-sync schemes, TSHL [Sye06] has the most energy efficient signaling scheme while MU-Sync [Chir08] obtains the best performance. We propose a light-weight version of D-Sync, called B-D-Sync which uses the signaling scheme of TSHL while achieving the performance of MU-Sync.

3.4.1 Protocol Description

We begin by describing at a high level the operation of D-Sync. The goal of the protocol is to synchronize nodes to a single node within their transmission range, which we refer to as the beacon. The beacon essentially takes the role of a cluster-head and is responsible for estimating the clock offset and skew for all nodes that can communicate with it. The beacon can either be a super-node with a higher energy budget or it can be periodically elected among the nodes in the network. Once nodes are synchronized to the beacon, they are also synchronized to each other and network wide synchronization is achieved.

We first describe the basic signaling block when a single node has to synchronize to the beacon. For the network case, the same signaling pattern follows. Figure 3.3 summarizes the messages exchanged between a beacon node and an

unsynchronized node **B**. We also introduce some important signaling parameters that we will refer to later in this section.

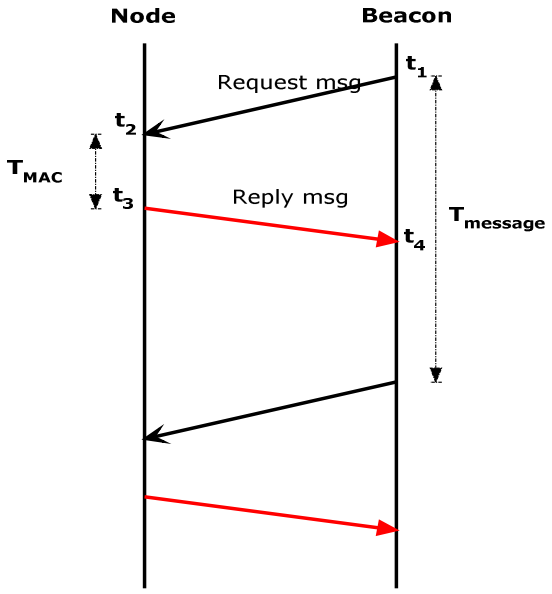


Figure 3.3: D-Sync Messaging Scheme

As shown in Figure 3.3, a beacon initiates the synchronization process by broadcasting a request message and storing the send time of the message, $t_A(t_1)$. Node **B** records the receive time of the message as per its local time, $t_B(t_2)$. As mentioned earlier, the message sent by the beacon can be used by node **B** to estimate its relative speed based on the estimated Doppler shift [Mas08] [Sha00]. Now, there is a delay in medium access, T_{MAC} before node **B** can reply message back to the beacon. We will refer to this time as the response time of node **B**. (Later in Section 3.4.3 we will discuss the effect of the response time on the performance of D-Sync and MU-Sync). Finally, node **B** transmits a reply message to the beacon at time t_3, time stamped by its local send time, $t_B(t_3)$. As before, the beacon estimates its relative speed to node **B** from the reply message and also records the receive time of this message, $t_A(t_4)$. This

process is repeated every $T_{message}$ (s) which is another signaling parameter that affects performance as we will discuss in Section 3.4.3. Given all the timing measurements and estimates of the relative node speeds obtained from Doppler, we will now derive the set of equations used by D-Sync to synchronize nodes.

We begin with a formal description and discussion of the synchronization problem, and continue to relate the unknown skew and offset of a node to the Doppler and timing measurements obtained from the messages described earlier.

As mentioned earlier, the local time of any node A is related to the true global time, t by equation (3.1). The synchronization problem is to estimate the clock offset and skew for all nodes with respect to a beacon node. To derive our estimator in the light of existing schemes, we begin with a set of basic equations that relate the propagation delay to the local time of nodes, when a request and reply message is exchanged between two nodes.

Consider a message that is sent by node **A** at time t_1 and received by node **B** at time t_2. The message is time-stamped by the local time of node **A** at time t_1 and by the local time of node **B** at time t_2. Using equation (3.1), these local times are related to the true global time as:

$$\begin{aligned} t_A(t_1) &= t_0 + \Gamma_A(t_0) + \delta_A \cdot (t_1 - t_0) \\ t_B(t_2) &= t_0 + \Gamma_B(t_0) + \delta_B \cdot (t_2 - t_0) \end{aligned} \tag{3.9}$$

The distance, $d_{AB}(t_1,t_2)$ defined as the distance between the position of node **A** at time t_1 , $P_A(t_1)$ and the position of node **B** at time t_2, $P_B(t_2)$ is related to the propagation delay, $(t_2 - t_1)$ as given by equation (3.10). We must emphasize that $d_{AB}(t_1,t_2)$ does not correspond to an actual distance between nodes at any time but is

the distance travelled by the sound wave in the interval ($t_2 - t_1$). By substituting for t_1 and t_2 in equation (3.10) in terms of the local time of nodes **A** and **B**, as given by equation (3.9) we obtain an equation that relates the skew and offset of node **B** to the unknown distance $d_{AB}(t_1,t_2)$, equation (3.11). Since node **B** is being synchronized to the clock of the beacon which is node **A**, node **A**'s clock defines the true time, i.e. $\Gamma_A = 0$ and $\delta_A = 1$. Therefore, the relative clock offset $\psi = \Gamma_B - \Gamma_A = \Gamma_B$.

$$\frac{\left\|P_A(t_1) - P_B(t_2)\right\|}{c} = \frac{d_{AB}(t_1,t_2)}{c} = t_2 - t_1 \tag{3.10}$$

$$\frac{d_{AB}(t_1,t_2)}{c} = \frac{t_B(t_2) - \Gamma_B}{\delta_B} - \frac{t_A(t_1) - \Gamma_A}{\delta_A}$$

$$t_B(t_2) = \delta_B \cdot \left(t_A(t_1) + \frac{d_{AB}(t_1,t_2)}{c} \right) + \Gamma_B \tag{3.11}$$

We can similarly obtain equation (3.12) when node **B** replies to node **A** at a later time, t_3 and node **A** receives this message at time t_4. This message exchange is summarized in Figure 3.3.

$$t_B(t_3) = \delta_B \cdot \left(t_A(t_4) - \frac{d_{BA}(t_3,t_4)}{c} \right) + \Gamma_B \tag{3.12}$$

Now, if the nodes are stationary, but the propagation delay is not negligible, as is the case in static underwater networks, then we have a total of **3** unknowns and **2** equations. If we had one more message exchange between nodes we could obtain 2 more constraints which would make the problem solvable [Sye06]. However, if nodes are mobile, the relative distance between nodes also varies with every message exchange. As a result, the number of unknowns grows with the number of equations. Specifically, for n equations we have $n+2$ unknowns. Further the equations are not linear in the unknowns. To address this problem, protocols such as MU-Sync make the

assumption that the relative node distance does not change during a message exchange and further use a coarse estimate of the skew to estimate the one-way propagation delay. We will show later in Section 3.4.3 how these assumptions affect the performance of MU-Sync. In D-Sync, we do not make the above assumptions. Instead we relate the clock offset and skew to the change in the relative distance of two nodes by adding equation (3.11) and equation (3.12) to obtain equation (3.13).

$$\begin{aligned} & t_B(t_3)+t_B(t_2) \\ &= \delta_B \cdot \left(t_A(t_4)+t_A(t_1)+\frac{d_{AB}(t_1,t_2)-d_{BA}(t_3,t_4)}{c} \right)+2\cdot\Gamma_B \\ &= \delta_B \cdot \left(t_A(t_4)+t_A(t_1)+\frac{d_{AB}(t_1,t_1)-d_{BA}(t_3,t_3)}{c} \right)+2\cdot\Gamma_B+\varepsilon_{motion} \end{aligned} \tag{3.13}$$

$$\varepsilon_{motion}=\frac{\delta_B}{c}\cdot\left(d_{AB}(t_1,t_2)-d_{AB}(t_1,t_1)+d_{BA}(t_3,t_3)-d_{BA}(t_3,t_4)\right) \tag{3.14}$$

ε_{motion} is the error due to node mobility and can be upper bounded as follows:

$$\begin{aligned} & \left|\varepsilon_{motion}\right| = \frac{\delta_B}{c}\cdot\left|d_{AB}(t_1,t_2)-d_{AB}(t_1,t_1)+d_{BA}(t_3,t_3)-d_{BA}(t_3,t_4)\right| \\ & \le \frac{\delta_B}{c}\cdot\left\{\left|d_{AB}(t_1,t_2)-d_{AB}(t_1,t_1)\right|+\left|d_{BA}(t_3,t_3)-d_{BA}(t_3,t_4)\right|\right\} \\ & \le \frac{\delta_B}{c^2}\cdot R\cdot\left(v_B+v_A\right) \end{aligned} \tag{3.15}$$

Where, R is the maximum transmission range.

v_A is the maximum speed of node A.

v_B is the maximum speed of node B

So far, we have obtained a relationship between the unknown skew and offset and the change in the distance between nodes during a request and reply message. Now if $v_{AB}(t)$ is the relative speed between nodes, the total change in distance during a request and reply exchange is given by equation (3.16). From here on we will denote

$d_{AB}(t_1,t_1)$ by $d_{AB}(t_1)$.

$$d_{AB}(t_3) - d_{AB}(t_1) = \int_{t1}^{t3} v_{AB}(t)dt \tag{3.16}$$

We now formulate the estimation problem by relating the Doppler measurements obtained at the end of each message to the relative node speed. The observed Doppler shift is the projection of the difference in the velocity of nodes on the distance between them, given by:

$$f_{doppler}(t) \cdot \lambda = \frac{\left(V_B(t) - V_A(t)\right)^T \left(P_B(t) - P_A(t)\right)}{\left\|P_B(t) - P_A(t)\right\|_2} \tag{3.17}$$

Where, λ is the wavelength of sound. The relative node speed is given by the change in the distance between nodes as:

$$\begin{aligned} v_{AB}(t) = \frac{\partial d_{AB}(t)}{\partial t} &= \frac{\partial \left\|P_B(t) - P_A(t)\right\|_2}{\partial t} \\ &= \frac{1}{\left\|P_B(t) - P_A(t)\right\|_2} \cdot \left(P_B(t) - P_A(t)\right)^T \left(\frac{\partial P_B(t)}{\partial t} - \frac{\partial P_A(t)}{\partial t}\right) \\ &= \frac{\left(P_B(t) - P_A(t)\right)^T \left(V_B(t) - V_A(t)\right)}{\left\|P_B(t) - P_A(t)\right\|_2} \end{aligned} \tag{3.18}$$

From equation (3.17) and equation (3.18), the Doppler shift is related to the relative speed of nodes as:

$$v_{AB}(t) = f_{doppler}(t) \cdot \lambda \tag{3.19}$$

By substituting equation (3.19) in equation (3.16) we obtain:

$$d_{AB}(t_3) - d_{AB}(t_1) = \lambda \cdot \int_{t1}^{t3} f_{doppler}(t)dt \tag{3.20}$$

From equation (3.20), we observe that the continuous Doppler shift is required to estimate the change in distance. However, we will only use the two Doppler

measurements obtained at the end of each message exchange. Our estimate of the change in distance is given by equation (3.21).

$$\begin{aligned} & d_{BA}(t_3) - d_{AB}(t_1) \\ & = 0.5 \cdot \left(v_{BA}(t_3) + v_{AB}(t_1) \right) \cdot \left(t_3 - t_1 \right) + \varepsilon \\ & = \hat{v}_{AB}(t_1, t_3) \cdot \left(\frac{t_B(t_3) - \Gamma_B}{\delta_B} - t_A(t_1) \right) + \varepsilon \end{aligned} \tag{3.21}$$

Substituting equation (3.21) in equation (3.13), we obtain how the clock offset and skew are related to measurements of the relative speed of nodes obtained from Doppler:

$$\begin{aligned} & t_B(t_2) + t_B(t_3) \cdot \left(1 + \frac{\hat{v}_{AB}(t_1, t_3)}{c} \right) \\ & = \delta_B \cdot \left(t_A(t_4) + t_A(t_1) \cdot \left(1 + \frac{\hat{v}_{AB}(t_1, t_3)}{c} \right) \right) + \left(2 + \frac{\hat{v}_{AB}(t_1, t_3)}{c} \right) \cdot \Gamma_B + \varepsilon + \varepsilon_{motion} \end{aligned} \tag{3.22}$$

To summarize, when a request and reply message is exchanged between a beacon and an unsynchronized node; we can obtain a linear equation in **2** unknowns as given by equation (3.22). We can use the Ordinary Least Squares (OLS) estimator to solve the set of equations obtained when *N* such messages are exchanged. The estimator of the clock skew and offset for the above problem is then given below:

$$\hat{\Lambda} = \left[\hat{\delta}_B, \hat{\Gamma}_B \right]^T = \left(W^T W \right)^{-1} W^T Y \tag{3.23}$$

$$\begin{aligned} Y[i] &= t_B(t_2^i) + t_B(t_3^i) \cdot \left(1 + \frac{\hat{v}_{AB}(t_1^i, t_3^i)}{c} \right) \\ W[i,1] &= t_A(t_4^i) + t_A(t_1^i) \cdot \left(1 + \frac{\hat{v}_{AB}(t_1^i, t_3^i)}{c} \right) \\ W[i,2] &= 2 + \frac{\hat{v}_{AB}(t_1^i, t_3^i)}{c} \end{aligned} \tag{3.24}$$

Where, **Y** is a *N* x *1* vector and **W** is a matrix with *N* x *2* entries. The elements

of **Y** and **W** are given in equation (3.24). We next present an analysis of the error in D-Sync.

3.4.2 Error Analysis

There are two main sources of error in D-sync: the error due to random noise in Doppler measurements and the error due to the fact that Doppler measurements are not available continuously. As a result, the change in distance during a message exchange can only be estimated using the average of the two measurements obtained at the end of each message transmission, as described earlier. We will refer to the resulting error as the interpolation error. The timing error in equation (3.22) can be expressed in terms of afore mentioned errors as per equation (3.25). Equation (3.25) also shows the dependence of the error on the MAC delay, T_{MAC} shown in Figure 3.3. This delay affects the performance of MU-Sync as well which we will discuss in the next section.

$$\begin{aligned} \varepsilon &= \left(\varepsilon_{noise} + \varepsilon_{interp}\right) \cdot \left(\frac{t_3 - t_1}{\delta_B \cdot c}\right) \\ &= \left(\varepsilon_{noise} + \varepsilon_{interp}\right) \cdot \left(\frac{T_{MAC} + T_{prop}}{\delta_B \cdot c}\right) \end{aligned} \qquad (3.25)$$

$$T_{MAC} = t_3 - t_2; \;\; T_{prop} = t_2 - t_1$$

We will next derive an upper bound for the interpolation error in terms of the maximum and the average relative acceleration of a pair of nodes. This result is summarized in equation (3.26).

$$\varepsilon_{interp} = \frac{1}{t_3 - t_1} \cdot \int_{t_1}^{t_3} v_{AB}(t)dt - \left(\frac{v_{AB}(t_1) + v_{AB}(t_3)}{2} \right) \leq \frac{(t_3 - t_1) \cdot \left(a_{\max}^2 - a_{avg}^2 \right)}{4 \cdot a_{\max}} \tag{3.26}$$

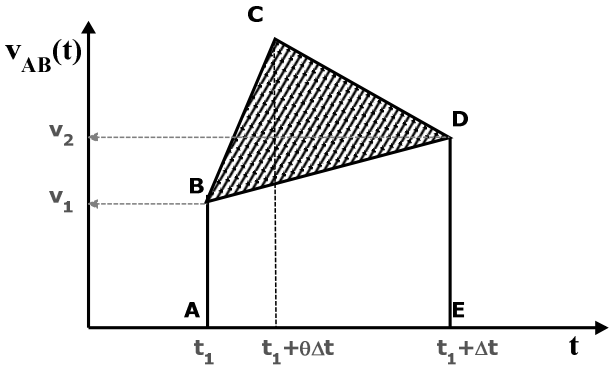


Figure 3.4: Maximum variation of relative speed with time

The interpolation error is given by:

$$\varepsilon_{interp} = \frac{1}{\Delta t} \cdot \int_{t_1}^{t_1+\Delta t} v_{AB}(t)dt - \left(\frac{v_{AB}(t_1) + v_{AB}(t_1 + \Delta t)}{2} \right) \tag{3.27}$$

The variation in the relative node speed must be such that it satisfies the two boundary data points obtained from Doppler measurements of the relative speed: $v_{AB}(t_1) = \mathbf{v_1}$ and $v_{AB}(t_1+\Delta t) = \mathbf{v_2}$.

The maximum interpolation error occurs when a node moves with maximum relative acceleration of a_{max} until sometime $t_1+\theta \cdot \Delta t$, and decelerates at a_{max} thereafter. This error is depicted in Figure 3.4 as the area of the triangular region **BCD**.

$$\max\{\varepsilon_{interp}\} = \frac{1}{\Delta t} \cdot \left(area(ABCDE) - area(ABDE) \right) \tag{3.28}$$

We first find θ so that the two boundary speed measurements are satisfied.

$$v_1 + \theta \cdot \Delta t \cdot a_{\max} = v_2 + (1-\theta) \cdot \Delta t \cdot a_{\max}$$

$$\theta = \frac{v_2 - v_1}{2 \cdot \Delta t \cdot a_{\max}} + \frac{1}{2} \tag{3.29}$$

Now area of the region ABCDE can be written as:

$$\begin{aligned} & area(ABCDE) \\ &= \frac{1}{2} \cdot \theta \cdot \Delta t \cdot \left(2 \cdot v_1 + a_{\max} \cdot \theta \cdot \Delta t\right) + \frac{1}{2} \cdot (1-\theta) \cdot \Delta t \cdot \left(2 \cdot v_2 + a_{\max} \cdot (1-\theta) \cdot \Delta t\right) \\ &= \frac{1}{4} \cdot \left(a_{\max} \cdot \Delta t^2 - \frac{\left(v_2 - v_1\right)^2}{a_{\max}} \right) + area(ABDE) \\ &= \frac{\Delta t^2 \cdot \left(a_{\max}^2 - a_{avg}^2\right)}{4 \cdot a_{\max}} + area(ABDE) \end{aligned} \tag{3.30}$$

From equation (3.30), the maximum interpolation error is given as:

$$\begin{aligned} \max\{\varepsilon_{interp}\} &= \frac{1}{\Delta t} \cdot \left(area(ABCDE) - area(ABDE)\right) \\ &= \frac{\Delta t \cdot \left(a_{\max}^2 - a_{avg}^2\right)}{4 \cdot a_{\max}} \end{aligned} \tag{3.31}$$

We next compare the performance of D-Sync and MU-Sync over a set of relevant parameters.

3.4.3 Performance of D-Sync

In this section we will compare the D-Sync and MU-Sync via simulations for a pair of nodes. For this comparison we use the same signaling scheme as MU-Sync, which we described at the beginning of Section 3.4.1. Later, we will discuss the performance of the two schemes in a network setting.

In the following simulations, nodes follow smooth curved paths with a constraint on their maximum speed and maximum instantaneous acceleration. Further,

all simulation results show the mean and standard deviation of the synchronization error 2 hours after a node is synchronized. A total of 10 request and reply messages are used for synchronization. The simulation parameters are summarized in Table 3.1 unless specified otherwise. We now describe the parameters that affect the performance of D-Sync and MU-Sync.

Response Time

As mentioned earlier, the response time is defined as the time elapsed before an unsynchronized node replies to a request message sent by the beacon node, indicated by T_{MAC} in Figure 3.3. While this time is assumed to be very small in MU-sync [Chir08], it can be quite substantial in underwater networks. Due to the low data rates of underwater acoustic modems, the duration of packet transmission is in the order of hundreds of milliseconds. As the number of nodes competing to respond to a reference message increases, nodes will need to back off tens of seconds to avoid collision, depending on the number of contenders. Consequently, the response time of nodes also increases.

Figure 3.5 shows the mean and standard deviation of the synchronization error for MU-Sync and D-sync when the response time of a node is varied between 5s and 100s. In our simulations, the maximum relative speed of the node is 2m/s, which is nominal for self-propelled vehicles and corresponds to a higher mobility regime for devices that are not propelled.

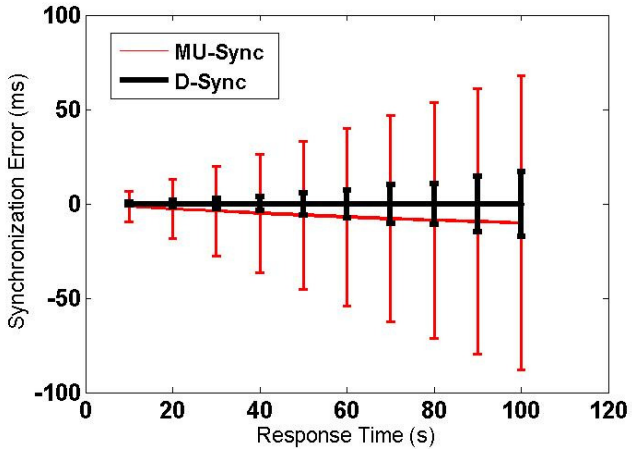


Figure 3.5: Performance with response time

We observe that both D-Sync and MU-Sync show an increasing trend with the response time, however for different reasons. In the case of MU-Sync the error increases because it assumes that nodes remain stationary during a message exchange. While in reality nodes would have moved a distance proportional to the response time. This translates to an error in the one-way propagation delay for MU-Sync. The error in D-sync increases with the response time because Doppler measurements are available only when a message is exchanged between nodes, as opposed to a continuous estimate. When the response time grows, so does the interpolation error as given in equation (3.25). However, D-Sync still outperforms MU-Sync because unlike MU-Sync, it takes node mobility into account during a message exchange. We will now compare the two schemes vs. network mobility.

Extent of Mobility

We define the extent of mobility as the relative speed of nodes. While the performance of MU-Sync is affected directly by the relative node speed, the performance of D-Sync depends on the rate of change of relative speed as we have

shown in Section 3.4.2, equation (3.31). To further illustrate this point, we compare the two protocols via simulations when the maximum relative speed of a node is varied from .01 m/s to 5m/s. The maximum relative acceleration was fixed at .04 m/s^2. All other simulation parameters are given in Table 3.1. From Figure 3.6 we observe that at low node speeds (below 1m/s), the performance of MU-Sync and D-Sync are comparable. This is expected, since the assumptions made by MU-Sync are valid at low mobility. However, with increased mobility, D-Sync significantly outperforms MU-Sync and has consistent performance across different speeds.

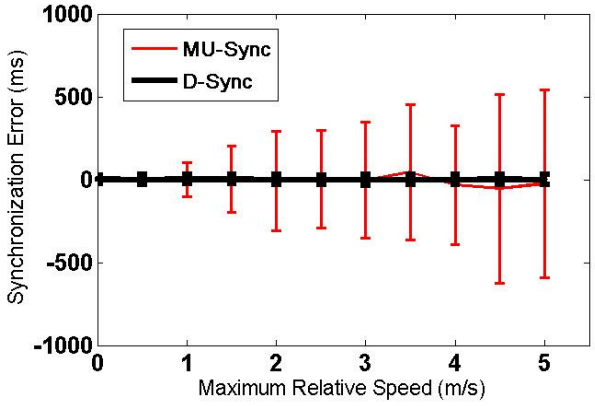


Figure 3.6: Performance with relative node speed

The accuracy of D-Sync is affected by the relative acceleration of nodes. This is shown in Figure 3.7 where the maximum relative acceleration is varied between .01m/s^2 to 0.1 m/s^2, while the maximum relative speed is kept constant at 2m/s. The error in MU-Sync had a standard deviation of around 200ms and did not vary with relative acceleration, so it is not shown in Figure 3.7. We next evaluate the robustness of D-Sync to the error in speed estimates obtained from Doppler.

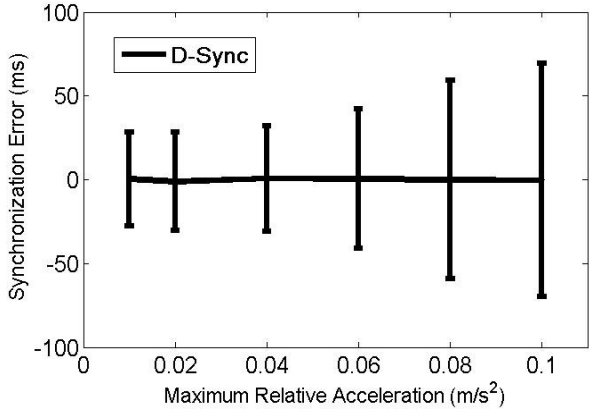


Figure 3.7: Performance with relative node acceleration

Error in Doppler measurements

The error in the estimate of relative node speeds based on Doppler measurements only affects the performance of D-Sync. Figure 3.8 shows how the synchronization error of D-Sync varies when the standard deviation of the error in speed is increased from .01m/s to 0.5 m/s. While the nominal reported error is 0.1m/s [Mas08], we observe that D-Sync still significantly outperforms MU-Sync at errors of above 0.4m/s. The error in MU-Sync was once again around 200ms and showed no variation with the Doppler error as expected.

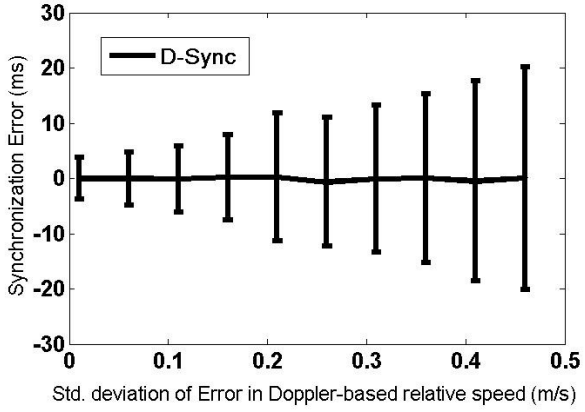


Figure 3.8: Performance of D-Sync with error in Doppler-based relative speed estimate

We will next describe why the interval at which request messages are sent by the beacon affects synchronization performance.

Interval between request messages

In its basic form MU-sync recommends that the cluster head sends out a message every 5s. Further, to achieve accurate synchronization, each node transmits a total of 25 messages. However, we recognize that estimating the skew is equivalent to estimating the slope of a line that best fits the timing data. To get a good fit, the dynamic range over which regression is performed should be far greater than the error in measurements. One way of increasing this dynamic range is to allow enough time to elapse before initiating the next request-reply exchange. Figure 3.9 shows how the performance of both MU-Sync and D-Sync can be improved by increasing the interval between request messages, while keeping the number of messages constant. The results show that increasing the message interval is a simple and effective way of improving synchronization performance without additional energy overhead.

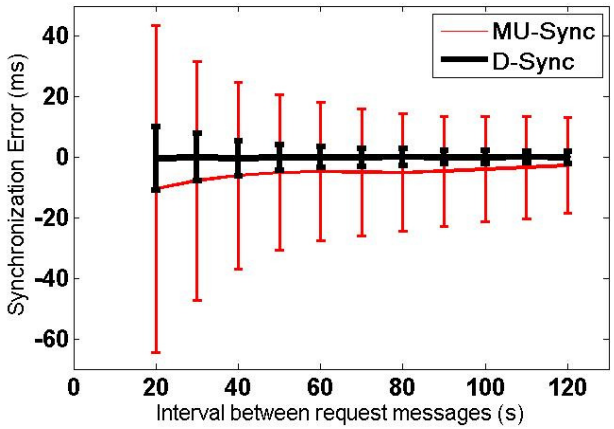


Figure 3.9: Performance with message interval

Total number of reference messages

Finally, we compare the two schemes when the total number of messages transmitted per node is increased from 5 to 45 as shown in Figure 3.10. While both protocols benefit from more messages, D-sync outperforms MU-sync especially when the number of messages transmitted per node is lower. D-Sync is also able to achieve high accuracy, i.e. 10 ms error 2 hours after synchronization, using only 10 messages.

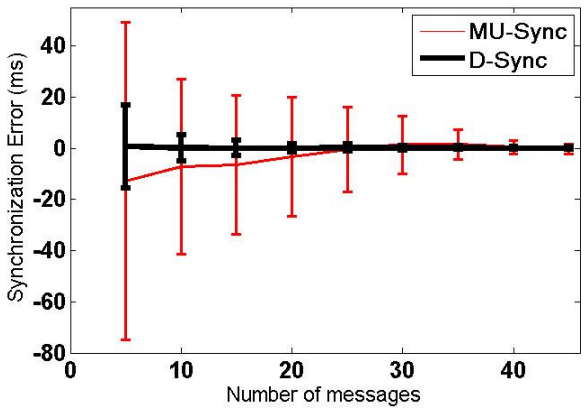


Figure 3.10: Performance with number of messages

To summarize, we have shown that D-Sync outperforms MU-Sync over a range of relevant parameters. As a next step we propose a more light-weight version of our protocol, called Broadcast D-Sync (B-D-Sync) that is especially suitable for denser networks.

Table 3.1: Simulation Parameters

Packet Length, L	60 bytes
Data Rate, R	240 bps
Max Transmission Range, D	1000 m
Max. Offset Error	.03 s
Std. Doppler Error	0.1 m/s
Max. back off Duration, T_{MAC}	30 s
Slot Length, t_{slot}	.976 s

3.5 A Variant of D-Sync: Broadcast D-Sync

We have earlier shown that D-Sync can accurately synchronize a pair of nodes by exchanging several two-way messages using broadcast signals. However, if we want to synchronize a group of nodes, then each node must transmit a reply message for every request message sent by the beacon. As a result, the total energy consumption can quickly ramp up. On the other hand, TSHL is known to be very energy efficient for synchronizing a group of nodes. Since TSHL assumes that the network is static, it can estimate the clock skew from a number of consecutive beacon broadcasts. Further, it requires only a single two-way exchange between the beacon and an unsynchronized node to obtain the unknown propagation delay and clock offset. As such, TSHL is not applicable to mobile systems because the propagation delay is no longer constant and the variation in the propagation delay is not known. However, we can strategically exploit Doppler information to overcome this problem as explained below.

Using Doppler measurements we estimate the change in the relative distance or in other words the change in the propagation delay. Instead of relying on two-way messages to obtain the propagation delay, we first obtain an accurate estimate of the one-way delay from a single two-way exchange. We then use Doppler measurements to estimate one-way delay for every broadcast message transmitted by the beacon (see Figure 3.11). Once the propagation delays for all messages are known, we apply linear regression to jointly estimate the skew and offset. Therefore, nodes need to respond to only one beacon message to synchronize.

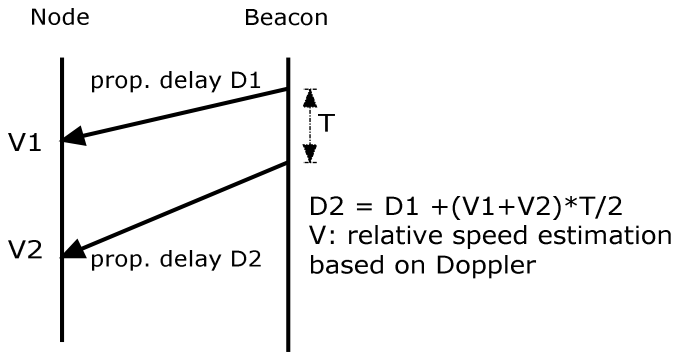


Figure 3.11: Relationship between consecutive propagation delays

We now describe the messaging scheme used by B-D-Sync which is slightly different from TSHL. Instead of dividing the synchronization into two phases, we jointly estimate the skew and offset. Here we focus on a single hop network with n nodes. We organize the network into a beacon node and n-1 regular nodes. The beacon node is in charge of initiating the synchronization process by broadcasting a number of beacon messages. In addition, the beacon node will mark its last message to which nodes respond. Each unsynchronized node records when a beacon message was received. After the last message is received, nodes send all their previous recorded timestamps to the beacon node in a response message (or store this information on their memory banks for offline computation). Finally, the beacon node computes the skew and offset for each unsynchronized node, and broadcasts the result to all nodes. To balance the energy consumption, nodes take turns in playing the role of the beacon. Figure 3.12 shows the detailed messaging scheme for B-D-Sync.

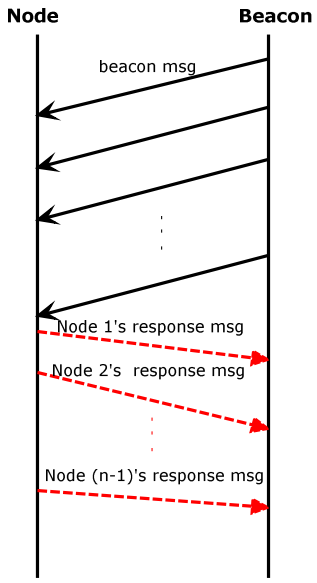


Figure 3.12: B-D-Sync Messaging

Next, we will evaluate MU-Sync and TSHL together with our proposed schemes in a network setting.

3.6 Comparison of D-Sync and B-D-Sync with Existing Protocols

In our simulation set up, nodes are placed in a 1000m by 1000m field. Each node can move on a smooth curved path in this field with a maximum relative velocity of 2m/s and a maximum relative acceleration of 0.1m/s^2. The speed of sound is set to 1500m/s and remains constant throughout the synchronization process. Nodes have a skew of 80ppm and an offset of 60 μs relative to the beacon clock. Nodes use a slotted contention based MAC protocol to gain channel access. Assuming a maximum initial synchronization offset of 0.03s, the slot length t_{slot} is calculated based on the packet size, maximum synchronization offset, maximum distance between nodes and data rate as specified in Table 3.1. Given the above parameters, for a one-hop network with n nodes, each node chooses a random back-off in the range [0, 10*(n-1)*t_{slot}] before it

responds to a beacon message. The interval between consecutive beacon messages is set to 2s for B-D-Sync and TSHL, and (10*n*t_{slot} + 5) s for D-Sync and MU-Sync, respectively. The multiplicative constant 10 is carefully chosen to ensure that all nodes can respond before the next beacon transmission. Finally, the Doppler-based relative speed measurements have a Gaussian error with standard deviation 0.1m/s, which is a typically suggested value [Mas08].

As both D-Sync and MU-Sync rely on two-way messages, they will follow the same signaling scheme as explained in Section 3.4.1. Similarly TSHL and broadcast D-Sync share the same signaling scheme as illustrated in Figure 3.12. We now compare the performance of the four protocols in terms of accuracy and energy consumption, and show how they vary with network size.

We define accuracy as the mean of the absolute timing error, two hours after synchronization. We define the energy consumption as the total number of messages sent by all nodes in the network. Figure 3.13 shows the accuracy of each scheme, while Figure 3.14 depicts the corresponding energy consumption. From Figure 3.13 and Figure 3.14, we observe that D-Sync significantly outperforms MU-Sync however with identical energy consumption.

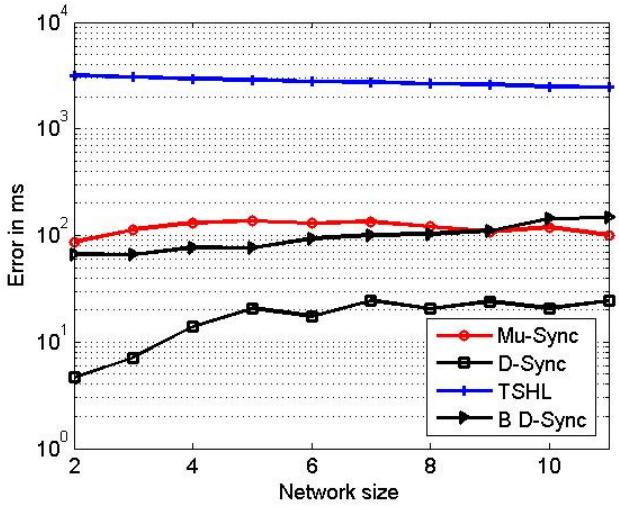


Figure 3.13: Comparison of the accuracy of protocols vs. network size

The error in D-Sync increases from 7 to 20ms when the network size grows from 2 to 11 because of the increased delay in MAC access time, T_{MAC} . The high accuracy achieved by D-Sync makes it an ideal candidate to avoid frequent re-synchronization. Further B-D-Sync has a performance comparable to MU-Sync, however using far less energy.

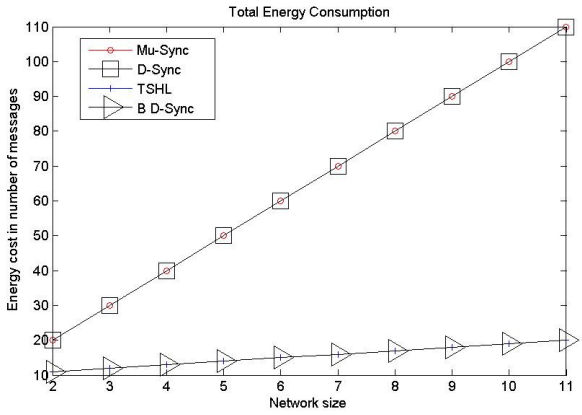


Figure 3.14: Comparison of the energy consumption of different protocols

Since D-Sync and MU-Sync use the same signaling scheme, their energy consumption curves entirely coincide. Similarly, TSHL and broadcast D-Sync have identical energy consumption. Based on Figure 3.14, it is clear that B-D-Sync and

TSHL consume far less energy as compared to MU-Sync and D-Sync. When energy consumption per synchronization is a constraint, broadcast D-Sync would be a perfect candidate.

In summary, node mobility in underwater networks combined with low data rates of acoustic modems can have a significant impact on the performance of time-synchronization protocols. While the absolute velocity of nodes is hard to obtain underwater, information about their relative speed is a key input to time-synchronization. Incidentally, the Doppler shift caused by the relative motion of nodes is a very well studied problem since it is a major impairment to underwater acoustic communication. A number of effective physical layer techniques have been developed to estimate the Doppler shift. We have shown that our proposed time-sync protocols strategically leverage Doppler information provided by the physical layer to achieve accurate time synchronization. Depending on how energy-constrained nodes are and the level of time-sync accuracy required by applications, D-Sync or B-D-Sync would be promising candidates for time synchronization in underwater networks.

3.7 Related Work

Two unique characteristics of underwater sensor networks make underwater time synchronization challenging. The first one is large propagation delays. The second relates to the inherent mobility in underwater systems. Even for static underwater systems, sensor nodes tend to experience some degree of mobility due to ocean currents or wind. As a result, the propagation delay will not remain constant.

TSHL [Sye06] is the first method specifically designed to deal with long propagation delays. Their synchronization protocol is organized in two phases. In the

first phase they perform linear regression over timing information from multiple beacon transmissions so that nodes are skew synchronized. In the second phase, the clock offset is corrected by exchanging two-way messages. The fundamental assumption is that the distance and thus the propagation delay is a constant throughout the skew estimation phase. Essentially, they assume a static network which does not hold for most underwater systems.

To account for the time variability in the propagation delay due to the relative motion of nodes, MU-Sync [Chir08] employs frequent two-way messaging to estimate both the offset and skew. In MU-Sync, the clock skew is estimated by performing linear regression twice over a set of local timing information collected via a two-way message exchange with a cluster-head. Because of using a large number of two-way messages, MU-Sync is not as energy efficient as TSHL. Furthermore, it assumes that the one-way propagation delay can be estimated as the average round trip time. However, for underwater mobile systems with nominal mobility, the estimate of the one-way propagation delay using MU-Sync becomes quickly biased. Finally, due to channel contention, nodes have to defer their transmission for random periods before responding to the cluster head. As the number of nodes increases, this time duration becomes longer which significantly deteriorates the performance of MU-Sync.

The closest work to our proposed solution is Mobi-Sync [Liu09], in which the spatial correlation of nodes' velocities is exploited to estimate the time varying propagation delay. Nodes are classified into three groups: surface buoys, super nodes, and ordinary nodes. It is assumed that surface buoys are equipped with GPS to obtain global time reference and super nodes can communicate directly with them to

maintain synchronization. An ordinary node launches time synchronization by broadcasting request messages to its neighboring super nodes. Upon receiving the request message, each super node responds with a measurement of its absolute velocity. Nodes use a correlation model to estimate their velocity given the velocity of the super-node. While being effective in estimating the time varying delay, this protocol needs to know the exact correlation model between nodes, which is very hard to obtain. Also for networks with self-propelled vehicles, there may not be any correlation among neighboring nodes. On the contrary, our proposed scheme does not make any assumption about the underlying motion model nor does it require the motion correlation statistics of nodes for time synchronization. In addition, for Mobi-Sync, the network has to be densely deployed to ensure that each ordinary node maintains connectivity to at least three or more super nodes in order to have a good estimate of velocity.

3.8 Conclusion

Node mobility in underwater networks combined with low data rates of acoustic modems can have a significant impact on the performance of time-synchronization protocols. While the absolute velocity of nodes is hard to obtain underwater, information about their relative speed is a key input to time-synchronization. Incidentally, the Doppler shift caused by the relative motion of nodes is a very well-studied problem since it is a major impairment to underwater acoustic communication. A number of effective physical layer techniques have been developed to estimate the Doppler shift. We have shown that our proposed time sync protocols strategically leverages Doppler information provided by the physical layer to achieve

accurate time synchronization. Depending on how energy constrained nodes are, we believe D-Sync or B-D-Sync would be promising candidates for time synchronization in underwater networks.

3.9 Acknowledgments

This chapter is, in part, a reprint of material published in: “Energy-Efficient Ranging for Post-Facto Self-Localization in Mobile Underwater Networks”, IEEE Journal on Selected Areas in Communication (JSAC) Special Issue on Underwater Wireless Communications and Networks, Vol. 26, Issue 9, pp.1697-1707, December 2008 and in part a reprint of material accepted for publication in the Proceedings of ACM International Workshop on Underwater Networks (WUWNET), 2010 under the title “D-Sync: Doppler-based time synchronization for mobile underwater sensor networks”.

In the first case, I was the primary researcher and author and Curt Schurgers supervised the research and the latter case was done in collaboration with Feng Lu and Curt Schurgers supervised the research.

CHAPTER 4

COLLABORATIVE TRACKING: SPATIO-TEMPORAL COMBINING

4.1 Introduction

As mentioned before, in our overall system setup, nodes collect measurements of their motion and their distances to neighbor nodes while they are submerged. The specific algorithms that operate on this data to estimate node trajectories are run offline after this data is available at a central location. In Chapters 2 and 3, we investigated how inter-node distance measurements could be obtained in a network setting and the parameters that affect performance. In this chapter we will present the specific tracking algorithms that can optimally combine both inter-node distances and measurements of nodes' motion obtained from on-board sensors. The framework that we present in this chapter is applicable to a variety of *mobile instruments*. These include vehicles that are self-propelled (such as remotely operated underwater vehicles or ROVs), derive mobility from vertical motion (such as gliders) or passively float with the underwater currents (such as subsurface drifters). In the case of these mobile underwater platforms, location determination is not a one-shot event; instead positions have to be tracked over time. While traditional approaches either start from a temporal or spatial view, in this chapter, we propose an integrated spatio-temporal solution. Our collaborative tracking provides robust positioning of mobile underwater networks with limited resources, relying on both low-cost navigational sensors and inter-node distance estimates.

Independent tracking: the temporal view

Underwater navigation has long been an integral part of submersibles. When dealing with a group if vehicles, devices are treated as a number of independent entities rather than a collective in motion: it essentially tracks each vehicle separately. This is achieved by equipping a vehicle with its own navigational unit consisting of a suite of sensors to measure depth, attitude, velocity and acceleration and algorithms to merge these measurements into a running fix of the device's position [Bla03] [Rom05] [Kin06].These algorithms also rely on periodic external position updates to compensate for the cumulative effect of dead-reckoning errors. These external inputs are often obtained via acoustic range measurements to surface transponders or arrays of them (yielding either three independent range measurements for long baseline systems [Bin06], or a combination of range and angular data for short baseline [Aud04]).

While effective for stand-alone devices, this traditional approach has a number of key drawbacks when considering a collective of them. First, it does not scale well as the number of vehicles increases. A larger network typically occupies a larger region in space. With a limited number of surface beacons, the spacing between them therefore increases and the frequency of external position updates goes down. Also, in mobile networks in general, devices may move occasionally out of range from surface transponders. As expected, when external position updates become sparse and dead-reckoning errors grow, this approach starts breaking down. The problem is that information is really only considered in one dimension - *time*. The trajectory of each vehicle is essentially viewed as an independent string, as illustrated in Figure 4.1a. It

completely ignores the spatial dimension, i.e., the information that could be obtained by also looking at the other vehicles.

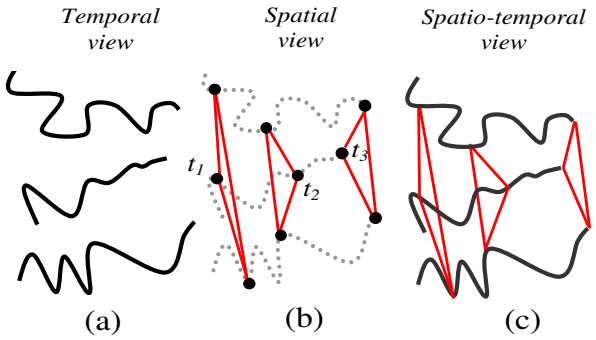


Figure 4.1: Views of multi-vehicle trajectory estimation.

Collaborative self-localization: the spatial view

Networked submersible platforms are envisioned to be equipped with short to medium range communication capabilities [Jaf06] [Hei06]. In this case, inter-node distance estimates can be readily obtained via acoustic ranging. Leveraging inter-device measurements has been the key component of so-called collaborative localization [Cha06]. Devices communicate with each other, and resolve their positions by manipulating the network-wide geometric constraints (only a few beacons are needed in the network). In essence, this collaborative localization exploits the *spatial* dimension. To extend it towards mobile scenarios, the network is periodically re-localized at a succession of time snapshots [Mir08], as illustrated in Figure 4.1b. However, this approach ignores the temporal dimension, as each node's trajectory is now viewed as a set of uncorrelated points in time.

As we will see later, this solution breaks down when the network becomes spatially sparse, i.e., when each device only has few neighbors. Collaborative localization was first proposed for static networks, such as terrestrial sensor networks.

However, unlike these terrestrial sensor networks which can be very dense due to extremely low cost mote-like platforms, underwater networks are still expected to operate in more sparse regimes. Furthermore, the unpredictable dynamics of ocean currents can result in highly varying densities over the network lifetime. This problem of network sparsity will have a detrimental effect on collaborative localization, as we will discuss in the next section.

Collaborative tracking: the spatio-temporal view

Instead of the above approaches, in a mobile networked system, the problem should really be viewed as a *4-dimentional* one. As shown in Figure 4.1c, node trajectories form a set of interrelated strings. Both temporal (dead-reckoning using navigational instruments) and spatial (inter-device range measurements) dimensions should be exploited together. The most natural approach is to merge traditional tracking with collaborative localization. Specifically, position estimates obtained with collaborative localization can serve as external updates to tracking. Or equivalently, data from navigational instruments can be used to interpolate between periodic collaborative position estimates. While intuitively appealing, we will show that this *combined strategy does not work*. It still suffers from exactly the same intrinsic limitations discussed before. The problem is that this natural merging of schemes does not provide a truly 4-dimensional view. Instead, we will propose a novel approach that incorporates both time and space equally. Essentially, it operates on the mobile network as a set of interacting strings. This approach will provide real *collaborative tracking* (as opposed to of enhanced collaborative localization). Furthermore, a truly collaborative scheme that optimally combines all available measurements will be able

to operate well even into regions where network connectivity becomes sparse and surface beacons are few and far between.

The collaborative framework that we introduce in this chapter relies on range estimates from acoustic modems (inter-device angular information is much harder to obtain as array-style communication is typically impractical on individual devices) and cheap navigational instruments. While accurate navigational systems have been used in stand-alone vehicles, they may be too expensive for these new networked platforms consisting of many vehicles. However, compact and less expensive alternatives are available. e.g., pressure sensors made from strain gages or quartz crystals can measure depth; magnetic compasses provide heading information with 1-10 degrees of precision; inertial sensors such as MEMs-based accelerometers, magnetometers and gyros with moderate accuracy are available at low cost (tens of dollars). Note that on mobile platforms, velocity information is hard to obtain, as direct measurements only yield values relative to currents. While ADCPs (acoustic Doppler current profilers) can measure absolute velocity if bottom-lock can be obtained, these instruments have a high cost and large size.

4.2 Why Collaborative Tracking?

Before introducing our collaborative tracking solution, we briefly look at the more straightforward technique of directly combining traditional stand-alone tracking with collaborative localization. First, networked collaborative localization is run periodically, with inter-device distance measurements obtained from acoustic ranging. Next, the resulting position estimates are used as external inputs to the tracking algorithm, which uses dead-reckoning from cheap navigational sensors.

Unfortunately, this solution suffers from an inherent flaw. In the first step, i.e. the collaborative localization, each time instance of the network is treated as an independent estimation problem. The performance of this localization greatly depends on the network density, as shown in Figure 4.2. It depicts the mean RMS localization error and related standard deviation for a network of 20 nodes and 3 beacons as a function of density (expressed as the average number of neighbors per node; transmission range is 200m). The key point is the distinct transition effect (error jumps as density decreases beyond a certain point – note the scale on the graph). The reason is that as the network becomes sparser, we fairly rapidly shift to a situation where the network becomes *non-localizable*. This means that node positions can no longer be uniquely resolved (errors of several hundred meters). E.g., with only one measurement, the node can be anywhere on a ring (with the thickness of the 2D ring determined by the ranging uncertainty; depth is available from pressure sensors).

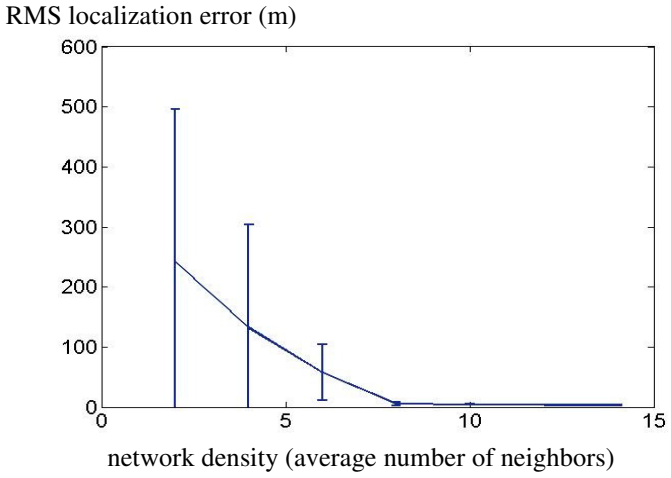


Figure 4.2: Performance of collaborative localization.

The problem is that for such a sparse network, collaborative localization essentially is unable to provide any useful position information to the tracking algorithm. Therefore, for sparse networks, almost all nodes are non-localizable and we expect the performance of tracking basically to default back to the case where no external position information is available. Essentially, all the information from inter-node distance measurements is lost. However, even if the network is non-localizable at each point in time, there is still a lot of useful information in the distance information between nodes that can be leveraged. The key idea is that navigational data and ranging information should be used jointly, in a true 4-dimentional estimation problem.

4.3 Problem Formulation

In its basic form, the collaborative tracking problem that we will talk about in this section can be seen as a complex multi-dimensional spatio-temporal estimation problem. The goal is to find the maximum likelihood estimate for the overall probability distribution function that simultaneously captures the locations of all devices in the spatial and temporal dimensions. The constraints are given by the set of inter-device range estimates collected at discrete time instants, as well as the inputs from the individual navigational sensors. We will now formally define this problem.

We consider a network with N nodes with unknown positions during the tracking interval. The unknown nodes are indexed by the set $U = \{1, ...N\}$. There are M position references (with known positions), indexed by the set $Q = \{N+1,...M+N\}$. Nodes can obtain inter-node distance estimates only with neighbors that are in their communication range. The data available for tracking which is derived from

measurements made while devices are submerged, during the tracking period, (0, T) can be any combination of the following:

(1) The position of reference nodes during the tracking interval, $\mathbf{Q} = \{P_i(t_k)\}$.

Where, i = N+1... N+M; k = 1… K; t_k ϵ (0,T)

(2) The set of inter-node distance estimates obtained between node-pairs, $\mathbf{z_D} = \{ \tilde{d}_{ij}(t_k) \}$

Where, i ϵ {1,… N}, j ϵ {1,…M+N}. The set { t_k }represents the time instances when measurements were made.

$$\tilde{d}_{ij}(t_k) = \left\| P_i(t_k) - P_j(t_k) \right\| + \tilde{\varepsilon}_d \qquad (4.1)$$

$\tilde{\varepsilon}_d$ is random error in distance estimates which we characterized in Chapter 2.

(3) Measurements of the unknown nodes' instantaneous translational velocity, $\mathbf{z_V} = \{ \tilde{v}_i(t_v) \}$

$$\tilde{v}_i(t_v) = \left. \frac{\partial P_i(t)}{\partial t} \right|_{t=t_v} + \tilde{\varepsilon}_v \qquad (4.2)$$

Where, i ϵ {1,…N};{t_v }ϵ (0,T); $\tilde{\varepsilon}_v$ is random error in velocity measurements.

(4) Measurements of the unknown nodes' acceleration, $\mathbf{z_A} = \{ \tilde{a}_i(t_k) \}$

$$\tilde{a}_i(t_k) = \left. \frac{\partial^2 P_i(t)}{\partial t^2} \right|_{t=t_k} + \tilde{\varepsilon}_a \qquad (4.3)$$

Where, i ϵ {1,…N} ;{ t_a }ϵ (0,T); $\tilde{\varepsilon}_a$ is random error in acceleration measurements.

(5) Measurements of the unknown nodes' heading, $\mathbf{z_\theta} = \{\theta_i(t_j)\}$

Where, $i \in \{1,\ldots N\}$;$\{ t_j \} \in (0,T)$

Given all the above measurements, the problem is to obtain the Maximum Likelihood estimates of the position of all nodes during the tracking interval (0, T) at a time-granularity of Δt, equation (4.4). The position of a node i at time t in 2-D is denoted by $P_i(t)$. The problem is to find $\mathbf{P}^*$as defined in equation (4.4). Since nodes know their depth from measurements of pressure sensors [Kin06], we estimate their positions in 2-D, by looking at the projection of a 3-D network on a 2-D plane.

$$\mathbf{P}^* = \{P_1^*(k \cdot \Delta t), P_2^*(k \cdot \Delta t), \ldots P_N^*(k \cdot \Delta t)\} \forall k = 1,..K$$

$$P_i^*(k \cdot \Delta t) = \arg\max \iiint_{\Theta_{i,k}} p(\mathbf{P}, \mathbf{V} \mid \mathbf{Q}, z_D, z_V, z_A, z_\theta) \cdot d\Theta_{i,k} \qquad (4.4)$$

$$\Theta_{i,k} = \{\mathbf{P}, \mathbf{V} \setminus P_i(t_0 + k \cdot \Delta t)\}$$
$$\mathbf{P} = \{P_i(k \cdot \Delta t)\} \forall i = 1,\ldots N; k = 1,\ldots K$$
$$\mathbf{V} = \{V_i(k \cdot \Delta t)\} \forall i = 1,\ldots N; k = 1,\ldots K$$

The above described collaborative tracking is a complex estimation problem. The complexity essentially is due to distance measurements made between unknown nodes. This introduces a great deal of inter-dependence between large numbers of unknown states. As such as direct approach to evaluating equation (4.4) is infeasible. However, the general problem of computing the distribution of individual variables from a global function defined over many variables is frequently encountered in coding theory. Solutions for particular instances of this problem (i.e. particular structures of the joint distribution) have been previously proposed under different names. Some examples are the Forward /Backward algorithm or BCJR, iterative turbo

decoding and decoding of LDPC codes. However, it has been shown that all these algorithms and many others (Pearls belief propagation and even Kalman filters) are all instances of a single generic message passing algorithm, the sum-product algorithm that operates on a 'factor-graph' [Ksc01].

These factor-graphs offer a way to represent any global function (in this case the multi-dimensional probability distribution) in terms of simpler local functions that depend only on a subset of variables. The sum-product algorithm can operate on this graph and exploit these simple relations to estimate the global function via iterative message passing. We will first present a brief overview of factor-graphs and the sum-product algorithm based on excellent tutorials by Kschischang et al [Ksc01] and Loeliger [Loe04]. We refer the reader to these tutorials for an in-depth review of these topics.

4.4 Application of Factor Graphs to Probabilistic Inference

The sum-product algorithm is the solution to the following generic problem. Given a global function $g(x_1, x_2, x_3,\ldots, x_n)$, where each x_i takes values in the discrete domain A_i, the sum-product algorithm computes simultaneously and efficiently the *summary* for each x_i where the summary function is defined in equation (4.5)

$$\sum_{\sim x_1} g(x_1, x_2, x_3) = \sum_{x_2 \in A_2} \sum_{x_3 \in A_3} g(x_1, x_2, x_3) \tag{4.5}$$

When the global function is a joint distribution, the algorithm naturally computes the marginal distributions for each x_i. To do this, it exploits the way the global function $g(.)$ factorizes. Since computing the summary of any state-variable for the global function is computationally intensive, the structure of the global function is

exploited by representing it in terms of the product of a number of functions that only depend on a subset of the variable. An example of such a representation is given by equation (4.6).

$$g(x_1, x_2, x_3, x_4) = f_A(x_1, x_2) f_B(x_3, x_4) f_C(x_1, x_4) \tag{4.6}$$

Where, f_A ,f_B, f_C are functions of a subset of the variables and show the structure of the global function $g(.)$ by capturing the inter-dependence between different state-variables, x_i.

Now, factor-graphs provide a means of representing the structure of the global function graphically. However, their significance is more than just an alternate representation. This is because once the factor-graph representation of the global function is obtained; the sum-product algorithm can operate on this graph to efficiently compute the summary function given in equation (4.5). It is noteworthy, that the sum-product algorithm can *jointly* determine the summary of a number of variables. The factor-graph form of the example function, given by equation (4.6) is shown in Figure 4.3.

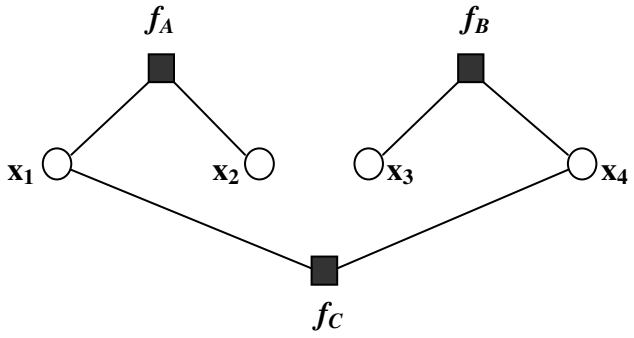


Figure 4.3: An example factor-graph

As mentioned earlier, the sum-product algorithm can now operate on the example factor-graph in Figure 4.3, to jointly estimate the summary of variables x_1, x_2, x_3, x_4. During the operation of the sum-product algorithm, messages are generated by state-variables and function nodes. We will next briefly describe these messages.

Let $\boldsymbol{\mu}_{x\text{-}f}(x)$ denote the message sent from a state-variable, $\boldsymbol{x}$ to a function node f (X), where X is the set of arguments of f. Let $\boldsymbol{\mu}_{f\text{-}x}(x)$ be the message sent from a function node to a state-variable. Also, let $n(w)$ denote the set of neighbors of a given node w on the graph. Messages are computed by each node as per the following rule [Ksc01] [Loe04].

$$\mu_{x-f}(x) = \prod_{h \in n(x) \setminus \{f\}} \mu_{h-x}(x) \tag{4.7}$$

$$\mu_{f-x}(x) = \sum_{\sim\{x\}} \left(f(X) \prod_{y \in n(f) \setminus \{x\}} \mu_{y-f}(y) \right) \tag{4.8}$$

The algorithm operates as follows. Per iteration, nodes compute outgoing messages on all their links based on the (latest) messages that had arrived on those links in a previous iteration. Nodes initiate message-passing by assuming that a unit message has arrived on each of their links. Further the algorithm has to iterate a number of times to converge. The presence of cycles in the factor-graph affects the convergence time.

The estimate of a state-variable's distribution is the product of all its incoming messages. To obtain a physical interpretation of the operation of the sum-product algorithm, it is sufficient to examine a message passed from one state-variable to another via a function node as shown in Figure 4.4. In step 1, the message, $\mu_{x\text{-}f}(x)$ is

sent from state-variable $\boldsymbol{x}$ to function-node f with $f \in \{\boldsymbol{f_1}, \boldsymbol{f_2}\}$. This message is an estimate of the probability mass distribution of $\boldsymbol{x}$. Next, in step 2, node $\boldsymbol{f}$ computes an estimate of the distribution of the position of $\boldsymbol{y}$ based on the likelihood of any information that relates $\boldsymbol{x}$ and $\boldsymbol{y}$.

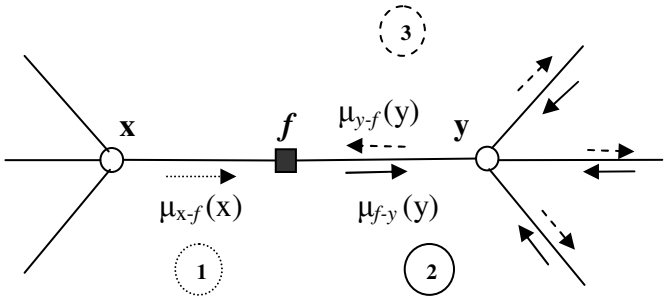


Figure 4.4: Messages passed between two state-nodes

We observe in step 2, that a number of messages can simultaneously arrive at $\boldsymbol{y}$, each being an individual estimate of its position-distribution. Node $\boldsymbol{y}$ intersects all these individual distributions to obtain an estimate of its distribution. This is the product step of the algorithm. In step 3, $\boldsymbol{y}$ sends out the most recent estimate of its distribution to all its neighbors.

4.5 Maximum Likelihood Estimation Based on Factor Graphs

As a first step towards solving our collaborative tracking problem, we have come up with the appropriate factor-graph description given in Figure 4.5. It gives a graphical representation of the interdependencies between the unknown positions of the devices in space and time. Mathematically, this graph describes the joint distribution of nodes' positions given all distance estimates and motion measurements.

Velocities are added has hidden variables, allowing us to relate accelerations to positions (although we did not intend on estimating velocities to begin with).

The structure of the graph in Figure 4.5 can be viewed in terms of a number of basic sub-graphs, which is the key to solving the complex estimation problem in a computationally efficient way. First, one observes three main chains stacked vertically. Each chain captures the trajectory of an unknown node (we only showed three to not overload the figure), where the horizontal dimension represents time. The circles indicate the state-variables that have to be estimated, which are the node positions (P_{ij} for the position of node i at time t_j) and velocities (V_{ij}). Various types of square blocks link the state-variables together. These blocks, known as function-nodes, not only indicate which state-variables are related but also how they are related.

Two main types of interdependencies are captured by the graph: temporal and spatial. Temporal interdependencies (defined by function nodes of type f_1, f_2 and f_3) relate the instantaneous positions and velocities within each chain. Spatial interdependencies (defined by functions of type f_4), on the other hand, are between chains and are dictated by inter-node distance measurements. Function f_1 describes how the instantaneous velocity of each node is related to its position at consecutive time steps, according to the first principles definition of instantaneous velocity. Function f_2 relates the instantaneous velocity of nodes over time given acceleration measurements, while f_3 describes velocity given measurements of a node's heading. In principle speed measurements can also be included, however we do not use these here, since this information is difficult to obtain underwater.

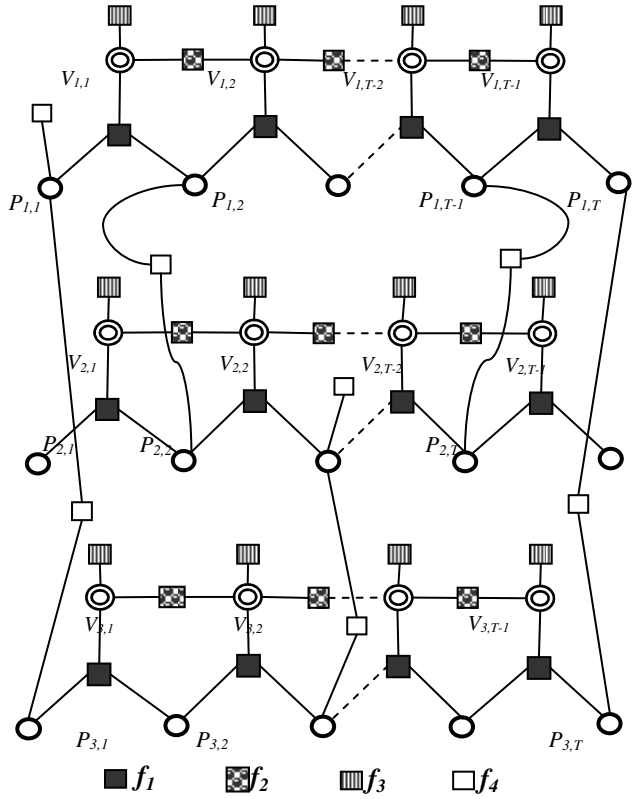


Figure 4.5: Factor graph representation of 4D tracking

Function nodes can take any generic form that best describes the likelihood of measurements given state-variables [Ksc01] [Loe04]. For our specific problem, their formal definition is given in equation (4.9).

$$
\begin{aligned}
f_1(P_{i,s+1}, P_{i,s}, V_{i,s}) &= p(P_{i,s+1} \mid P_{i,s}, V_{i,s}) = I(P_{i,s+1} - P_{i,s} - V_{i,s} \cdot \Delta t = 0) \\
f_2(V_{i,s}, V_{i,s+1}) &= p(\tilde{a}_{i,t} \mid V_{i,s}, V_{i,s+1}) \\
f_3(V_{i,s}) &= p(\tilde{\theta}_{i,s} \mid V_{i,s}) \cdot p(\tilde{\upsilon}_{i,s} \mid V_{i,s}) \\
f_4(P_{l,k}, P_{m,k}) &= p(\tilde{d}_{lm}^{t_k} \mid P_{l,k}, P_{m,k})
\end{aligned}
\qquad (4.9)
$$

Where, Δt is the time granularity at which motion measurements are obtained. All other variables as previously defined.

We will later present the details of our final algorithm, including the exact definitions of these likelihood functions. However, we will postpone this discussion until Section 4.5.2 when we address the complexity of the algorithm. The reason is that the exact definitions of the likelihood functions follow from how the distributions of state-variables are represented. To reduce the computation overhead we have made a set of specific choices regarding these representations, which we will discuss in Section 4.5.2. Specifically, each state-variable should first define the space over which its probability density will be estimated by the algorithm. While both position and velocity are most naturally defined over a continuous space, the sum-product algorithm has to operate on discrete sample points. As a result, we will choose piece-wise constant distributions that are uniformly weighted over 2D grids to approximate the true and unknown densities of state-variables. As a consequence of this choice of distributions, the definitions of the function nodes described by equation (4.9) have to be modified: since a sample point now represents an area or grid in 2D, each function node, in a strict sense, has to be integrated over the areas represented by its arguments. Furthermore, how the discretization is optimized and how many grids are chosen, is addressed by the adaptive sampling algorithm given in Section 4.5.2.

We next describe the overall operation of the sum-product algorithm on the factor-graph presented in Figure 4.5 and develop an intuitive understanding of the estimation process. These intuitions will later help us to analyze the performance of the algorithm.

4.5.1 Algorithm Operation

Once the factor graph is defined, the estimation problem is solved by running the sum-product algorithm on it. As mentioned in Section 4.4, the sum-product algorithm is essentially a message passing algorithm. Nodes begin by composing messages and sending them out over the links of the factor graph. There are two main types of messages exchanged during a single iteration of the algorithm. Messages from a function node f to a state-variable x, denoted as $\mu_{f-x}(.)$ and messages from state-variables to function nodes, denoted as $\mu_{x-f}(.)$. These are computed as per equations (4.7) and (4.8)

The two types of messages essentially set up the internal machinery of the graph that operates as follows. State-variables send out their most current estimate of their distributions to all their neighbor functions, while function nodes send out their estimates of the distribution of neighbor state-variables. These outgoing messages are generated by combing the incoming messages in a past iteration as per equations (4.7) and (4.8). A state-variable estimates its distribution (or outgoing messages) by intersecting the individual estimates of its distribution provided by its function-node neighbors, equation (4.7). A function node generates its message to a neighbor state-variable by performing a marginalization of its local likelihood function, equation (4.8). These messages, operating at a micro-level, provide a way of modifying node distributions by carrying information across the graph.

For tracking, one view of what takes effect as a result of the sum-product algorithm is follows. Consider the experience of a state-variable in the graph that represents a node's position at a certain time. Messages that flow to it from the

temporal dimension essentially translate the position distribution of the node at all other instances in the past and future to a common point. Messages that flow into it from the spatial dimension (between nodes) transform the position estimates of its neighbors to a distribution for the node. These individual estimates are then intersected to obtain more accurate estimates.

One important implication of this framework is that we have unified the notions of localization and tracking in a collaborative setting (although other attempts have been made in the past for single robot systems [Fox98] or in the case of multiple platforms when both distance and angle are available [Zha08] [Fox00]). As such, tracking has long implied estimating the position of a node in time, starting with a position fix at some initial time (with Gaussian error). This is especially true for tracking based on Kalman filters. As time progresses, if the position estimate becomes very inaccurate, tracking is no longer possible. This means new measurements cannot be correctly fused to update the position of the node. Instead the device has to be localized from scratch by combining multiple measurements. This is often called the *wake-up robot* problem or *global positioning problem* in robotics [Neg03] [Web10]. While, tracking based on particle-filters [Aru02] has been proposed to address this problem [Fox00] [Fox98], how these particle-filters would communicate in a collaborative setting is not addressed so far. In our factor-graph framework, each function-node essentially acts as a particle filter, modifying and propagating generic distributions that are fed into it in the form of messages. However, the key advantage of our factor-graph solution is how these particle filters are connected so that they act in unison for estimating the trajectories of all nodes, utilizing all available information.

4.5.2 Complexity Reduction and Analysis

While the sum-product algorithm offers an elegant way of solving the complex multi-dimensional tracking problem, its computational complexity is still prohibitively high. We will explore a number of ways to considerably reduce this complexity with little compromise on performance. Consider a situation where we are tracking N nodes over a period of T (s) (with ΔT (s) as the time-granularity of tracking). This would result in K time steps over which nodes are tracked with $K = \lceil T / \Delta T \rceil$. Assume during those K time steps, there are L distance estimates between unknown nodes and L_B with beacon nodes (i.e. nodes with known positions on the surface). In that case, the total computational cost (in number of operations) is given by the total number of iterations until convergence (the sum-product algorithm is based on iterative message passing) multiplied by the cost per iteration N_C. Equation (4.10) shows the expression for N_C which is derived from equations (4.6) and (4.7) for one iteration of the algorithm.

$$N_C \approx \alpha \cdot M^3 \cdot N \cdot K + \beta \cdot M^2 \cdot (N \cdot K + L) + \gamma \cdot M^2 \cdot L_B + \rho \cdot L \cdot M \qquad (4.10)$$

Where, $\alpha = 3 \cdot (C_3 + 3);\ \beta = 2 \cdot (C_2 + 2);\ \gamma = (C_1 + 1);\ \rho = 2$

In this expression, C_n is the cost of evaluating a function with n arguments. These functions are the ones needed to calculate the messages passed between nodes, from equations (4.7) - (4.8). This will be discussed more in detail later. Finally, the message length M captures how state information (e.g. node position P_{ij}) is numerically represented in the algorithm.

As can be observed from equation (4.10), the message length, M has a large impact on the overall complexity, and an efficient representation is therefore crucial. Secondly, while the natural state-space representation of position and velocity is in a continuous domain (and so the likelihood functions are also defined in this domain), the sum-product algorithm operates most efficiently on discrete representations. As a result the new likelihood functions are obtained by integrating the original functions over a hyper-volume of n x 2 dimensions where n is the number of neighbors of the function node. The cost of calculating these integrals can be quite high. Lastly, the granularity of the tracking, ΔT determines the number of state-variables that have to be estimated. A smaller time-granularity results in a more time-steps, K and proportionally increases the number of unknown states. This effect is also captured by equation (4.10). We will next address these problems by making specific choices about how distributions are represented and also show how the factor-graph in Figure 4.5 can be modified to reduce the number of estimated states.

4.5.2.1 Efficient Representation

Consider the node position estimates P_{ij}, which are 2D probability distributions (depth is known from pressure sensors). One way of representing these distributions numerically is to split the space into a grid and work with values at these grid points (essentially considering a sampled version of the continuous distribution). The number of samples (i.e., number of grid points) is what defines message length M. The problem is finding the appropriate sampling. First, the node keeps track of its allowable region, which is the smallest square region where it can reside. This region is then split in a number of grids or samples. At the onset, the node has no

information: its allowable region is the entire space and fine grids are useless and extremely wasteful. However, as the algorithm progresses, nodes should adapt their sampling and their allowable region to adequately represent their improved distribution estimates. As information percolates in time and space through the factor graph, the algorithm should therefore adapt its sampling representation at run time.

We base our solution on the following theorem that provides a simple relationship between the quality of piece-wise constant probability distributions (also called 'weighted discretizations') of $p(x)$ and its entropy $H(\cdot)$, defined in equation (4.11) [Isa09]:

Theorem 4.1: *Among any collection of weighted discretizations of $p(x)$, the minimum KL (Kullback–Leibler) divergence to $p(x)$ is achieved by a discretization that has minimum entropy H.*

$$H(q) = -\sum_{k=1}^{K} w_k \cdot \log\left({}^{w_k}\!/\!_{|V_k|} \right) \tag{4.11}$$

In this equation, w_k represents the weight and V_k is the vector dimensionality. The KL divergence is a measure of the discrepancy between the estimated distributions and the actual distribution and cannot generally be computed directly because the actual distribution is unknown. However, Theorem 4.1 says that for the particular class of approximating distributions, namely piecewise-constant distributions, which we are using in our scheme, minimizing the entropy of the approximating distribution is the same as minimizing the KL divergence. While this is a known result, Isard et al. used this theorem only to obtain the best non-uniform discretization with a fixed number of samples [Isa09]. On the other hand, we will use

it to adaptively adjust the number of samples and the granularity. By computing the entropy of our estimated distributions at run time, we obtain a measure of their quality as a function of the number and granularity of samples. We also use the entropy as a stopping condition for the algorithm. Specifically, we stop the message passing once the change in entropy computed by all state-variables is approximately zero. Our approach starts with very coarse representations, a single sample of large area representing the entire geographic region (~ kms). It then iteratively splits the space into equal sized grids until the change in entropy of the distribution is arbitrarily small. From Theorem 4.1, this entropy gradient approach allows us to decide what number of samples is sufficient to adequately represent a node's distribution. Note that this is done for each node independently. Figure 4.6 presents the pseudo code of this entire procedure. It enables us to adaptively select the minimum value of M (message size) in the computational complexity equation (4.10).

The second important factor that determines the complexity of the sum-product algorithm is the calculation of the messages passed between nodes. These were lumped together according to their number of arguments. The cost C_n for a function with n arguments is that of numerically computing an n^{th} order integral. To reduce this cost, we will discretize the weights of our distribution to one bit only. Essentially, we are only keeping track of which grids have a non-zero probability of the device residing there, without storing which grid is more likely. This allows us to convert all computation to binary logic: summations become a logical-OR and multiplications become a logical-AND**.** The resulting operations, which completely describe the sum-product algorithm we run on our factor graph, and thus our complete estimation

algorithm, are presented below. We first present the function node definitions when position and velocities are discrete random variables.

$$p\left(\tilde{d}^{t}_{(l,m)} \mid p_1, p_2\right) \approx g^{coarse}_{dist}\left(\left\|(p_1, z_1)^T - (p_2, z_2)^T\right\| - \tilde{d}^{t}_{(l,m)}\right) = U\left(-\varepsilon_d^{\max}, \varepsilon_d^{\max}\right) \quad (4.12)$$

$$p\left(\tilde{a}_{i,t} \mid v_1, v_2\right) \approx g^{coarse}_{accel}\left(v_2 - v_1\right) = U\left(-\varepsilon_a^{\max} \cdot \Delta T, \varepsilon_a^{\max} \cdot \Delta T\right) \quad (4.13)$$

$$p^{o}\left(v\right) \approx g^{coarse}_{speed}\left(\|v\|\right) = U\left(-s_{\max}, s_{\max}\right) \quad (4.14)$$

$$p\left(\tilde{\theta}_{i,s} \mid v\right) \approx g^{coarse}_{heading}\left(\angle v\right) = U\left(-\varepsilon_\theta^{\max}, \varepsilon_\theta^{\max}\right) \quad (4.15)$$

As mentioned earlier to reduce the computation overhead, we convert all operations to binary logic. We first restrict the weights of node distributions to be binary valued. Since the functions being estimated are binary, the likelihood functions can also have more coarse representations. Given bounds on measurement errors we define the likelihood distribution functions by:

$$f_1(P_{i,s+1}, P_{i,s}, V_{i,s}) = \iiint_{p_2 \in P_{i,s+1}, p_1 \in P_{i,s}, v \in V_{i,s}} I(p_2 - p_1 - v \cdot \Delta t = 0) dp_1 \cdot dp_2 \cdot dv \quad (4.16)$$

$$f_2\left(V_{i,s}, V_{i,s+1}\right) = \iint_{v_1 \in V_{i,s}, v_2 \in V_{i,s+1}} p\left(\tilde{a}_{i,t} \mid v_1, v_2\right) \cdot dv_1 \cdot dv_2 \quad (4.17)$$

$$f_3\left(V_{i,s}\right) = \int_{v \in V_{i,s}} p\left(\tilde{\theta}_{i,s} \mid v\right) \cdot p^0\left(v\right) \cdot dv \quad (4.18)$$

$$f_4\left(P_{l,k}, P_{m,k}\right) = \iint_{p_1 \in P_{l,k}, p_2 \in P_{m,k}} p\left(d^{k}_{(l,m)} \mid p_1, p_2\right) \cdot dp_1 \cdot dp_2 \quad (4.19)$$

Now, instead of computing the function nodes by evaluating the integrals in equations (4.16) to (4.19), we evaluate the functions f_1 , f_2 and f_3 to give a binary likelihood as follows:

We denote $\aleph(G,a)$ as the operation of expanding grid G by a in all dimensions. $c(G)$ as the set of all corners of grid G. $m(G)$ is the grid centre and $s(G)$ is the size of the grid. The '+' operation on a grid denotes translation in 2D.

$$f_1(P_{i,s+1},P_{i,s},V_{i,s}) \approx \left\{ \begin{matrix} 1, & if \quad \aleph\left(P_{i,s}+m\left(V_{i,s}\right)\cdot\Delta T, s\left(V_{i,s}\right)\cdot\Delta T\right)\cap P_{i,s+1}\neq\varphi \\ 0, otherwise \end{matrix} \right\} \tag{4.20}$$

$$f_2(V_{i,s},V_{i,s+1}) \approx \left\{ \begin{matrix} 1, & if \quad \aleph\left(V_{i,s}+\tilde{a}_{i,t}\cdot\Delta T, \varepsilon_a^{\max}\cdot\Delta T\right)\cap V_{i,s+1}\neq\varphi \\ 0, otherwise \end{matrix} \right\} \tag{4.21}$$

$$f_3(V_{i,s}) \approx \left\{ \begin{matrix} 1, & if \ \exists\ \Theta \ s.t.\ \Theta\in\left\{\angle c\left(V_{i,s}\right)\right\}\cap\left(\tilde{\theta}_{i,s}-\varepsilon_\theta^{\max}, \tilde{\theta}_{i,s}+\varepsilon_\theta^{\max}\right) \\ 0, otherwise \end{matrix} \right\} \tag{4.22}$$

$$f_4(P_{l,k},P_{m,k}) \approx \left\{ \begin{matrix} 1, if \left\{ \begin{matrix} \min\left(\max\left(\left\{\left\|c(P_{l,k})-c(P_{m,k})\right\|\right\}\right), \tilde{d}^{\,t}_{(l,m)}-\varepsilon_d^{\max}\right)- \\ \max\left(\min\left(\left\{\left\|c(P_{l,k})-c(P_{m,k})\right\|\right\}\right), \tilde{d}^{\,t}_{(l,m)}+\varepsilon_d^{\max}\right) \end{matrix} \right\} > 0 \\ 0, otherwise \end{matrix} \right\} \tag{4.23}$$

Based on the above binary-valued function-node definitions, the messages from equations (4.7) and (4.8) are computed using binary logic as:

$$\mu_{f-x}(x) = \underset{\{\sim x\}}{OR}\left[f(X) \underset{y\in n(f)\setminus\{x\}}{AND}\left[\mu_{y-f}(y)\right]\right] \tag{4.24}$$

$$\mu_{x-f}(x) = \underset{h\in n(x)\setminus\{f\}}{AND}\left[\mu_{h-x}(x)\right] \tag{4.25}$$

The resulting solution is a light-weight algorithm that can efficiently solve the complex multi-dimensional collaborative tracking problem. Running times on a 2 GHz Pentium Pro machine varied between 4 to 6 hours for a network of 20 nodes and a tracking interval of 1 hour, where positions were estimated every 20s. For networks of N nodes, this will scale as O(N), as follows from equation (4.10). The running-time

gains from adaptive representation and binary discretization were observed to be about two orders of magnitude.

```
while (algorithm_converged == false) {

   // Adapt the sample representation for all nodes i and time step t_j
   ∀i,t , repeat {
      split_grid(P_ij);      // split 2D grid of P_ij into 4 equal grids
      ΔH = H_i,t^old − H_i,t^new    // compute change in entropy
   } until ( ΔH < epsilon)   // discretization adequate

   // Run the sum-product algorithm
   sum_product_algorithm();
   ∀i,t , prune(P_ij);       // remove samples with zero belief

   // Stop when there is no change compared to previous iteration
   algo_converged = update();
}
```

Figure 4.6: Adaptive sample representation algorithm

4.5.2.2 State Reduction via Measurement Combining

The problem that we wish to address in this section arises because measurements of the instantaneous acceleration of nodes can generally be obtained at a much higher rate than the granularity at which nodes have to be tracked. To correctly arrive at the factor-graph representation, the tracking interval would have to be reduced to match the sampling rate at which acceleration measurements are observed, if the measurements are not modified. As a result, the number of states that have to be estimated would substantially increase and so would the computation overhead, as equation (4.10) indicates.

We will show in this section how measurements and the factor-graph shown in Figure 4.5 would have to be modified, in order to achieve a desired tracking granularity. Let Δt be the interval at which acceleration measurements are obtained.

As mentioned before ΔT is the tracking interval. If $\Delta T = N \cdot \Delta t$, then N is the number of extra states that are estimated during the tracking interval. We have so far denoted the position of a node i at time t_{N+1} by $P_{i,N+1}$. However, since acceleration measurements can be modified independently for each node, we will use a simplified notation P_k to denote the position of a node at time t_k . Using this notation, the position of a node at time t_{N+1} is related to its position at time t_1 as follows:

$$\begin{aligned} P_{N+1} &= P_1 + \sum_{j=1}^{N} v_j \cdot \Delta t = P_1 + \frac{\Delta T}{N} \cdot \sum_{j=1}^{N} v_j \\ &= P_1 + \overline{v}\left[1,\ N\right] \cdot \Delta T \end{aligned} \tag{4.26}$$

Where, $\overline{v}\left[i, j\right]$ is the average velocity in the interval (t_i, t_j). Similarly, we relate the position at time t_{2N+1} to the position at time t_{N+1}as:

$$P_{2N+1} = P_{N+1} + \overline{v}\left[N + 1,\ 2N\right] \cdot \Delta T \tag{4.27}$$

Using the similar procedure as above, the instantaneous velocity at time t_k denoted as v_k is related to v_1 and acceleration measurements as:

$$v_k = v_1 + \sum_{j=1}^{k-1} a_j \cdot \Delta t \tag{4.28}$$

We can now relate the average velocity to the instantaneous acceleration measurements as:

$$\overline{v}\left[1,\ N\right] = \frac{1}{N} \cdot \left(\sum_{k=1}^{N} v_k \right) = \frac{1}{N} \cdot \left\{ v_1 + \sum_{k=2}^{N} v_k \right\} \tag{4.29}$$

Substituting for v_k in the above equation using equation (4.28) we obtain:

$$\bar{v}\left[1,\ N\right] = \frac{1}{N}\cdot\left\{v_1 + \sum_{k=2}^{N}\left(v_1 + \sum_{j=1}^{k-1} a_j \cdot \Delta t\right)\right\}$$
$$= \frac{1}{N}\cdot\left\{N\cdot v_1 + \sum_{k=2}^{N}\left(\sum_{j=1}^{k-1} a_j \cdot \Delta t\right)\right\}$$
$$= v_1 + \frac{\Delta t}{N}\cdot\sum_{j=1}^{N-1}\left\{(N-j)\cdot a_j\right\} \qquad (4.30)$$

Following the form obtained in equation (4.30), we can obtain the average velocity in the interval (t_{N+1}, t_{2N+1}) as:

$$\bar{v}\left[N+1,\ 2N\right] = v_{N+1} + \frac{\Delta t}{N}\cdot\sum_{j=N+1}^{2N-1}\left\{(N-j)\cdot a_j\right\}$$
$$= v_1 + \frac{\Delta t}{N}\cdot\left\{N\cdot\sum_{j=1}^{N} a_j + \sum_{j=N+1}^{2N-1}\left\{(N-j)\cdot a_j\right\}\right\} \qquad (4.31)$$

Subtracting equation (4.30) from equation (4.31) and substituting acceleration measurements instead of actual acceleration, we relate the average velocities over two time epochs for any node *I* as:

$$\bar{v}_i\left[N+1,\ 2N\right] - \bar{v}_i\left[1,\ N\right] = \hat{a}_i\left[1,\ N; N+1,\ 2N\right]\cdot\Delta t + \tilde{\varepsilon}_\alpha.\Delta t$$

$$\text{Where, } \hat{a}_i\left[1,\ N; N+1,\ 2N\right] = \frac{1}{N}\left\{\sum_{j=1}^{N} j\cdot\tilde{a}_{i,j} + \sum_{j=N+1}^{2N-1}\left\{(N-j)\cdot\tilde{a}_{i,j}\right\}\right\} \qquad (4.32)$$

For a more short-hand notation we will denote:

$$\bar{V}_{i,s} = \bar{v}_i\left[(s-1)\cdot N+1,\ s\cdot N\right]\ ;\ \hat{a}_{i,s} = \hat{a}_i\left[(s-1)\cdot N+1,\ s\cdot N; s\cdot N+1,\ (s+1)\cdot N\right]$$

The factor graph in Figure 4.5 will now have to be modified as follows. Instead of estimating the instantaneous velocities, we will estimate the average velocities over

the tracking interval. Since the basic structure of the graph remains the same we will only show how a single temporal chain is modified to change the tracking granularity.

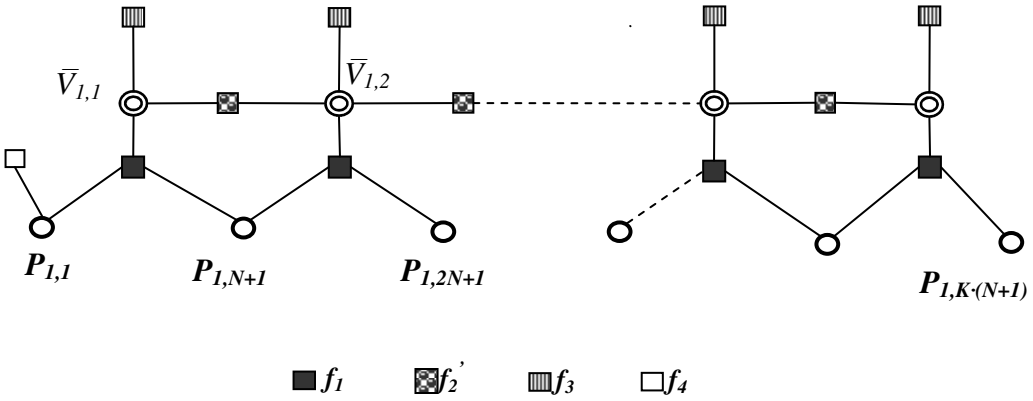


Figure 4.7: Modified factor-graph chain after state-reduction

Naturally the function f_2 has to be modified to f_2'. Following equation (4.32) and the grid operations defined for equation (4.21) we obtain f_2' as:

$$f_2'(\bar{V}_{i,s},\bar{V}_{i,s+1}) = \begin{Bmatrix} 1, & if \quad \lambda\left(\bar{V}_{i,s} + \hat{a}_{i,s}\cdot\Delta T, \varepsilon_{\alpha}^{\max}\cdot\Delta T\right) \cap \bar{V}_{i,s+1} \neq \varphi \\ & 0, otherwise \end{Bmatrix} \quad (4.33)$$

4.5.3 Performance Analysis

So far, we presented how collaborative Maximum Likelihood tracking can be performed based on factor-graphs. We would now like to analyze the performance of our estimator. In this section we will derive the covariance of the position error for the ML estimate of the position of a device at any time *t*, given all range and velocity measurements during a period (0, *T*). We derive the performance for one node in a network, whose neighbors know their positions perfectly. We will refer to the neighbor nodes as references. The ML estimation is performed under the condition that the measurement (distance and velocity estimate) errors have been characterized.

We will denote the error in the distance estimate obtained at time t_k as ε_k^R, which is Gaussian distributed with zero mean and standard deviation σ_R. Measurements of the node's velocity are obtained with period Δt. The error in velocity measurements is also zero-mean Gaussian with standard deviation σ_v.

Each distance measurement, $r(t_k)$ between an unknown node **U** and a reference node **R** at some time t_k results in an estimate of the pdf of the position of node **U** at time t_k. If the error in the distance measurements follows a zero-mean Gaussian distribution with standard deviation σ_R, the pdf of the position of node **U** as a result of a distance measurement made with a reference node is a *Gaussian ring*, depicted in Figure 4.8. The dark red regions represent the positions of maximum probability and the dark blue that of lowest probability. The Gaussian ring is formally defined as follows:

Definition 4.5.3.1: A *Gaussian ring* is the pdf of a 2 dimensional random variable $\boldsymbol{P} = [x, y]^T$ whose distance to a known center point $\boldsymbol{C} = [c_x, c_y]^T$ follows a Gaussian distribution with known mean and variance.

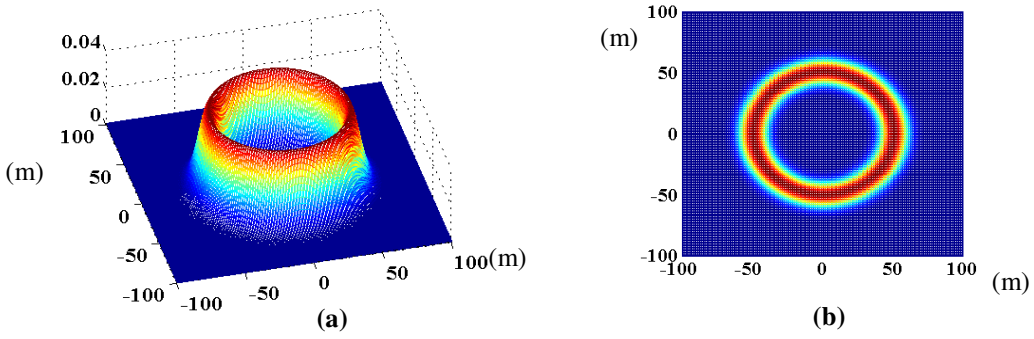


Figure 4.8: (a) pdf of position of node U as a result of a distance measurement made with a reference node (b) The Gaussian Ring: pdf of position of node U as seen from above in a 2D plane

The velocity measurements allow a way of translating a number of such Gaussian rings obtained at different points in time to a common time-instance. By intersecting these rings the position distribution of the node at any time t_0 can be obtained.

Since velocity measurements are not precise, the error in the radius of each ring after a temporal translation will grow. Further, the errors in rings that are translated through the same time period are correlated. Therefore, the intersection operation has to take this correlation into account. Conceptually, this is what the factor-graph algorithm is doing. However, a mathematical analysis of the above described steps is complex because of the non-linearity that arises with the rings. The exact final distribution of the position estimate of a node at any time instance can only be obtained numerically (as done by the sum-product algorithm). For a mathematical analysis we will make some simplifying approximations. Nevertheless, we arrive at very accurate error results as long as the ranging error is much smaller than the true distances.

As mentioned earlier, a distance measurement taken at any time t_k, results in a pdf of the unknown position at that time described by the Gaussian ring. To make our analysis tractable, we approximate the Gaussian ring by its tangent passing through the true position of the node at time t_k,P_U (t_k). The reason why we believe that such an approximation would hold is that while for a single measurement the tangent-line is a very inaccurate representation of the ring, the intersection of a number of such lines would be a fairly good approximation to the intersection of the corresponding rings.

To characterize our line representation of the Gaussian rings we introduce the following parameters.

The perpendicular to the tangent-line is at an angle α_k. Its thickness is ε_k^R, where, ε_k^R is the ranging error of the k^{th} distance estimate. $\varepsilon_k^R \sim N(0, \sigma_R)$. Our line approximation and the relevant parameters are illustrated in Figure 4.9. The darkest region of the line represents that of maximum probability. The ring shown in the figure is same as that in Figure 4.8, although the gray-scale distribution is not shown.

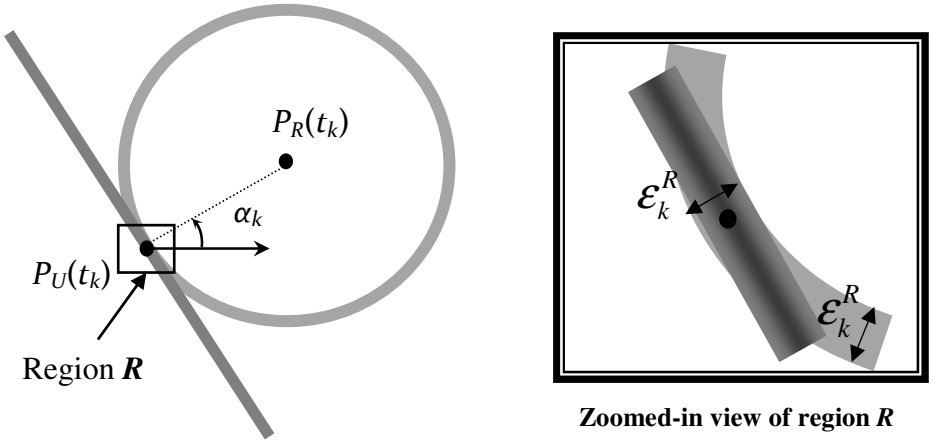


Figure 4.9: Line approximation to the Gaussian Ring

Using the line-approximation in place of the Gaussian-rings, we will now mathematically describe the operations explained earlier in this Section. Each velocity measurement taken at time ζ_i has an error $\varepsilon_i^v = \left[\varepsilon_i^{v_x}, \varepsilon_i^{v_y}\right]^T$. The errors in the velocity measurements that are used to translate any range measurement to a reference time t_0 will cumulatively add to the ranging error, ε_k^R. Since the same set of velocity measurements are added to range measurements that are translated through a common

time epoch, the errors in the translated measurements are correlated. As a first step, our goal is to obtain the covariance matrix of the error in the *N* range measurements after they have been translated to a common reference time t_0.

Let there be a total of *N* range measurements taken at time instances $\{t_1, t_2, ..., t_N\}$. From Figure 4.9, the corresponding line widths at the time each measurement was obtained are given by $\xi_R = \left[\varepsilon_1^R, \varepsilon_2^R, ..., \varepsilon_N^R\right]^T$. Note that each ε_k^R is in the direction α_k. Let us denote the sequence $E_{t_0} = [e_1, e_2, ..., e_N]^T$ as the widths of the *N* lines after they have been translated to the common reference time t_0. Also, let $\{\zeta_1, \zeta_2, ..., \zeta_M\}$ be the time-instances velocity measurements were obtained. For every velocity measurement added to a range measurement, only the components of the velocity measurements along ε_k^R would add to the line width. As a result, by projecting each velocity error onto the vector, ε_k^R we obtain the cumulative effect of velocity errors during the interval [t_k ,t_0). This is graphically represented in Figure 4.10.

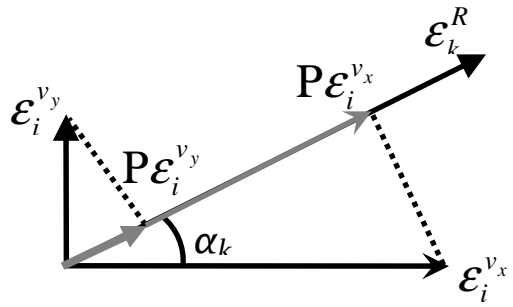


Figure 4.10: Contribution of velocity errors to error in range measurements

Following our prior discussion and from Figure 4.10, the equation that relates the error in each translated range measurement, e_k to the ranging error ε_k^R as well as the set of relevant velocity errors is given by:

$$e_k = \varepsilon_k^R + \cos\alpha_k \cdot \sum_{j:\zeta_j \in [t_k, t_0)} \varepsilon_j^{v_x} \cdot \Delta t + \sin\alpha_k \cdot \sum_{j:\zeta_j \in [t_k, t_0)} \varepsilon_j^{v_y} \cdot \Delta t \qquad (4.34)$$

Let $\xi_X = \left[\varepsilon_1^{v_x} \cdot \Delta t, \varepsilon_2^{v_x} \cdot \Delta t, ..., \varepsilon_M^{v_x} \cdot \Delta t\right]^T$ and $\xi_Y = \left[\varepsilon_1^{v_y} \cdot \Delta t, \varepsilon_2^{v_y} \cdot \Delta t, ..., \varepsilon_M^{v_y} \cdot \Delta t\right]^T$.

We rewrite equation (4.34) in matrix form to obtain the error in the translated measurements, E_{t_0} as:

$$E_{t_0} = \xi_R + \Lambda_c \Upsilon \xi_X + \Lambda_s \Upsilon \xi_y \qquad (4.35)$$

Where, Υ is a N x M matrix with elements $\Upsilon[i, j]$ defined as follows:

$$\Upsilon[i, j] = \begin{cases} 1, & if\ \ t_i \le \zeta_j \prec t_0 \text{ or } t_0 \prec \zeta_j \le t_i \\ 0, & otherwise \end{cases} \qquad (4.36)$$

Λ_c and Λ_s are each diagonal matrices of dimension N defined as follows:

$$\Lambda_c = diag\left(\cos\alpha_1, \cos\alpha_2, ... \cos\alpha_N\right),$$

$$\Lambda_s = diag\left(\sin\alpha_1, \sin\alpha_2, ... \sin\alpha_N\right)$$

From equation (4.35) and using the fact that the error vectors: ξ_R, ξ_X and ξ_Y are zero mean and statistically independent of each other, we now obtain the covariance error matrix of the N range measurements after they have been translated to a common reference time, t_0 as:

$$\begin{aligned}\operatorname{cov}\left(E_{t_0}\right) &= \mathrm{E}\left[\!\left[E_{t_0} {E_{t_0}}^T \right]\!\right] \\ &= \mathrm{E}\left[\!\left[\xi_R {\xi_R}^T \right]\!\right] + \Lambda_c \Upsilon \mathrm{E}\left[\!\left[\xi_X {\xi_X}^T \right]\!\right] \Upsilon^T \Lambda_c + \Lambda_s \Upsilon \mathrm{E}\left[\!\left[\xi_Y {\xi_Y}^T \right]\!\right] \Upsilon^T \Lambda_s \end{aligned} \tag{4.37}$$

Since all the distance measurements (prior to translation) are independent of each other and also the velocity measurements are independent, we have the following conditions:

$$\mathrm{E}\left[\!\left[\varepsilon_i^R \varepsilon_j^R \right]\!\right] = \begin{Bmatrix} 0, & \text{if } i \neq j \\ \sigma_R^2, & \text{otherwise} \end{Bmatrix} \tag{4.38}$$

$$\mathrm{E}\left[\!\left[\varepsilon_i^{v_x} \varepsilon_j^{v_x} \right]\!\right] = \begin{Bmatrix} 0, & \text{if } i \neq j \\ \left(\sigma_v \cdot \Delta t\right)^2, & \text{otherwise} \end{Bmatrix} \tag{4.39}$$

$$\mathrm{E}\left[\!\left[\varepsilon_i^{v_y} \varepsilon_j^{v_y} \right]\!\right] = \begin{Bmatrix} 0, & \text{if } i \neq j \\ \left(\sigma_v \cdot \Delta t\right)^2, & \text{otherwise} \end{Bmatrix} \tag{4.40}$$

Using equations (4.38) to (4.40), we can further simplify equation (4.37) as:

$$\begin{aligned} C_E = \operatorname{cov}\left(E_{t_0}\right) &= \sigma_R^2 \cdot \mathrm{I}_{MxM} + \left(\sigma_v \cdot \Delta t\right)^2 \cdot \left(\Lambda_c \Upsilon \Upsilon^T \Lambda_c + \Lambda_s \Upsilon \Upsilon^T \Lambda_s\right) \\ &= \sigma_R^2 \cdot \mathrm{I}_{MxM} + \left(\sigma_v \cdot \Delta t\right)^2 \cdot \left(\Upsilon \Upsilon^T \odot \left({R_\alpha}^T R_\alpha\right)\right) \\ &= \sigma_R^2 \cdot \mathrm{I}_{MxM} + \left(\sigma_v \cdot \Delta t\right)^2 \cdot \left(\Upsilon \Upsilon^T \odot \mathrm{B}_\alpha\right) \end{aligned} \tag{4.41}$$

Where,

$R_\alpha[1,i] = \cos\alpha_i$ and $R_\alpha[2,i] = \sin\alpha_i$, for all i = 1, 2, ..., N.

$\mathrm{B}_\alpha = {R_\alpha}^T R_\alpha$

$A \odot B$ is the Hadamard product of matrices A and B defined as:

$(A \odot B)[i,j] = A[i,j] \cdot B[i,j]$

We now write the joint distribution of the N translated-measurements as:

$$p\left(E_{t_0}\right) = \frac{1}{\sqrt{\left(2\pi\right)^N} \cdot \det C_E} \cdot \exp^{-\left(\frac{1}{2} E_{t_0}{}^T C_E^{-1} E_{t_0}\right)} \tag{4.42}$$

Once all the range measurements have been translated to the common time t_0, we obtain a number of lines passing through the true position of the unknown node at time t_0. Each line L_k has an error e_k in the direction α_k from our previous discussion. This is represented in Figure 4.11.

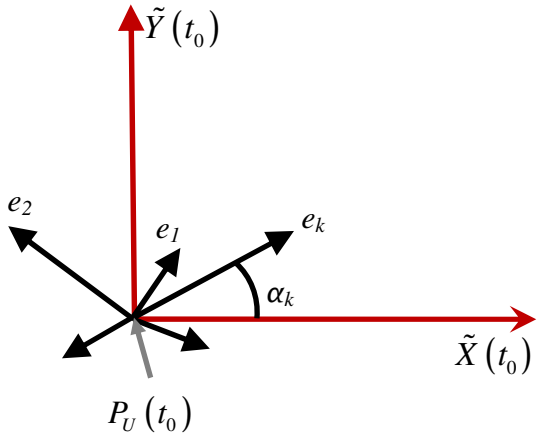


Figure 4.11: Error vectors of range measurements after translation to a common time, t_0

Now, intersecting all the lines should give us the approximated distribution of the unknown position. If the errors $\{e_1,..., e_N\}$ were independent, intersecting all the lines would be equivalent to multiplying the pdfs of all $\{e_k\}$. However, since these errors are correlated, with the covariance matrix given by equation (4.41), we would perform the intersection (or combining) operation as follows:

We will treat the joint pdf of the error in the N lines as a form of constraint on how the position distribution of the unknown node is described around the true position P_U (t_0). Using this perspective, if the position error in the X-direction, $\tilde{X}\left(t_0\right) = \tilde{x}$ and the position error in the Y-direction, $\tilde{Y}\left(t_0\right) = \tilde{y}$, then the projection of

$\tilde{p} = [\tilde{x}, \tilde{y}]^T$ on each of the vectors $\{e_k\}$ must follow the joint distribution given by equation (4.42). This can be mathematically written as follows:

$$p\left(\tilde{P}_U(t_0) = \tilde{p} = [\tilde{x}, \tilde{y}]^T\right) = p\begin{pmatrix} e_1 = \tilde{x}\cdot\cos\alpha_1 + \tilde{y}\cdot\sin\alpha_1, \\ e_2 = \tilde{x}\cdot\cos\alpha_2 + \tilde{y}\cdot\sin\alpha_2, \\ \vdots \\ e_N = \tilde{x}\cdot\cos\alpha_N + \tilde{y}\cdot\sin\alpha_N \end{pmatrix}$$

$$= p\left(E_{t_0} = R_\alpha{}^T\tilde{p}\right)$$

$$= \frac{1}{\sqrt{(2\pi)^N}\cdot\det C_E}\cdot\exp^{-\left(\frac{1}{2}\tilde{p}^T R_\alpha C_E^{-1} R_\alpha{}^T\tilde{p}\right)}$$

$$= \frac{1}{\sqrt{(2\pi)^N}\cdot\det C_E}\cdot\exp^{-\left(\frac{1}{2}\tilde{p}C_P{}^{-1}\tilde{p}^T\right)}$$

(4.43)

Where, R_α is defined in equation (4.41). From equation (4.43), we obtain the covariance error matrix of $P_U(t_0)$ denoted as C_p:

$$C_P = \left(R_\alpha C_E{}^{-1} R_\alpha{}^T\right)^{-1} \tag{4.44}$$

We also provide the following alternate proof which holds under the condition that the covariance matrix C_E should be invertible and $R_\alpha\, R_\alpha{}^T$ must be full rank (rank 2). From equation (4.41) since C_E is the sum of two matrices, one of which is the diagonal matrix and so, full rank, C_E is invertible. If there are at least two non-identical measurements, $R_\alpha\, R_\alpha{}^T$ would be full rank. As mentioned before, the vector $\tilde{P}_U(t_0)$ projected onto each vector $\{e_k\}$ should follow the joint distribution of E_{t_0}. In other words:

$$
\begin{aligned}
&E_{t_0} = R_\alpha{}^T \tilde{P}_U\left(t_0\right) \\
&\Rightarrow C_E = \operatorname{cov}(E_{t_0}) = R_\alpha{}^T \operatorname{cov}\left(\tilde{P}_U\left(t_0\right)\right) R_\alpha \\
&\Rightarrow I = C_E{}^{-1} R_\alpha{}^T \operatorname{cov}\left(\tilde{P}_U\left(t_0\right)\right) R_\alpha \\
&\Rightarrow I = R_\alpha C_E{}^{-1} R_\alpha{}^T \operatorname{cov}\left(\tilde{P}_U\left(t_0\right)\right) I \\
&\Rightarrow C_P = \operatorname{cov}\left(\tilde{P}_U\left(t_0\right)\right) = \left(R_\alpha C_E{}^{-1} R_\alpha{}^T\right)^{-1} \\
&\Rightarrow R_\alpha = R_\alpha C_E{}^{-1} R_\alpha{}^T \operatorname{cov}\left(\tilde{P}_U\left(t_0\right)\right) R_\alpha \\
&\Rightarrow R_\alpha R_\alpha{}^T = R_\alpha C_E{}^{-1} R_\alpha{}^T \operatorname{cov}\left(\tilde{P}_U\left(t_0\right)\right) R_\alpha R_\alpha{}^T \qquad (4.45) \\
&\Rightarrow I = R_\alpha C_E{}^{-1} R_\alpha{}^T \operatorname{cov}\left(\tilde{P}_U\left(t_0\right)\right) R_\alpha R_\alpha{}^T \left(R_\alpha R_\alpha{}^T\right)^{-1}
\end{aligned}
$$

The RMS error is given by the square-root of the covariance matrix, C_p :

$$RMS\left[\tilde{P}_U\left(t_0\right)\right] = \sqrt{tr\left(C_p\right)} \qquad (4.46)$$

Let $\Omega = R_\alpha C_E{}^{-1} R_\alpha{}^T$. From equation (3.44):

$$tr(C_p) = tr\left(\Omega^{-1}\right) = tr\left(\frac{adj\left(\Omega\right)}{\det\left(\Omega\right)}\right) = \frac{tr\left(adj\left(\Omega\right)\right)}{\det\left(\Omega\right)} \qquad (4.47)$$

Since Ω is a 2 x 2 matrix, from the properties of the trace operation, $tr(adj(\Omega))$ = tr(Ω). Also, det(Ω) = (tr(Ω)^2 – tr(Ω^2))/2. Using these facts we further simplify equation (4.47) as:

$$tr(C_p) = \frac{2 \cdot tr\left(\Omega\right)}{\left\{tr\left(\Omega\right)\right\}^2 - tr\left(\Omega^2\right)} \qquad (4.48)$$

Again, using trace properties:

$$
\begin{aligned}
&tr\left(\Omega\right) = tr\left(R_\alpha C_E{}^{-1} R_\alpha{}^T\right) = tr\left(C_E{}^{-1} R_\alpha{}^T R_\alpha\right) = tr\left(C_E{}^{-1} \mathrm{B}_\alpha\right) \\
&tr\left(\Omega^2\right) = tr\left(R_\alpha C_E{}^{-1} R_\alpha{}^T R_\alpha C_E{}^{-1} R_\alpha{}^T\right) = tr\left(C_E{}^{-1} R_\alpha{}^T R_\alpha C_E{}^{-1} R_\alpha{}^T R_\alpha\right) = tr\left(\left\{C_E{}^{-1} \mathrm{B}_\alpha\right\}^2\right)
\end{aligned} \qquad (4.49)
$$

To state our final result: We have derived the RMS error in the unknown position of a node at any time t_0 as:

$$RMS\left[\tilde{P}_U(t_0)\right] = \sqrt{tr\left(C_p\right)}$$

$$tr(C_p) = \frac{2 \cdot tr(\Phi)}{\left\{tr(\Phi)\right\}^2 - tr\left(\Phi^2\right)}$$

$$\Phi = \left(\sigma_R^2 \cdot \mathrm{I}_{MxM} + \left(\sigma_v \cdot \Delta t\right)^2 \cdot \left(\Upsilon\Upsilon^T \odot \mathrm{B}_\alpha\right)\right)^{-1} \mathrm{B}_\alpha \qquad (4.50)$$

Where, Υ is defined in equation (4.36)

$\mathrm{B}_\alpha[i, j] = \cos\left(\alpha_i - \alpha_j\right)$. All other parameters as defined earlier.

Next we want to evaluate the accuracy of our RMS error prediction.

4.5.4 Accuracy of Performance Prediction

To test the accuracy with which we can predict the performance of our algorithm as discussed in the previous section, we first considered a scenario with one beacon and one unknown node. The unknown node took periodic range measurements with the beacon. The distribution of the position of the unknown over time as computed by our Factor Graph tracking algorithm is shown in Figure 4.12.

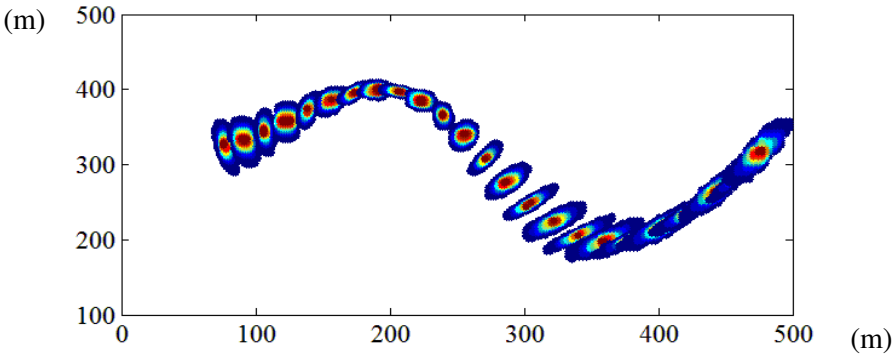


Figure 4.12: Scenario 1- Position-estimate distribution of a mobile node as computed by Factor Graph tracking algorithm

We computed the RMS error in the position estimate at each point in time based on the distribution obtained from simulations. We also computed the RMS error using our theoretical result, equation (4.50). Figure 4.13 shows both the error results over the tracking time. The percentage error in our estimate of accuracy is shown at the bottom section of the same figure. We observe that when the constraints are such that the true position distribution is actually Gaussian, we can predict the accuracy of estimates with less than 2% error.

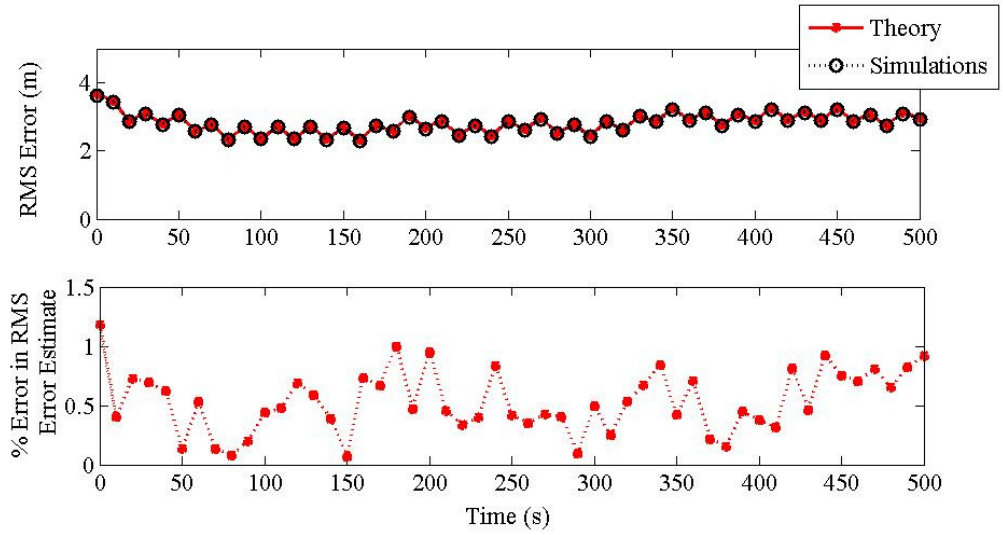


Figure 4.13: RMS error predicted vs. RMS error from simulations for Scenario 1

We next repeated this procedure for a second scenario consisting of 4 beacons and 1 mobile node. The mobile node broadcasts periodically to obtain distance estimates with all four beacons. The period of broadcast is set to double that of the tracking granularity. Once again Figure 4.14 shows the position estimate distribution as computed by our FG algorithm.

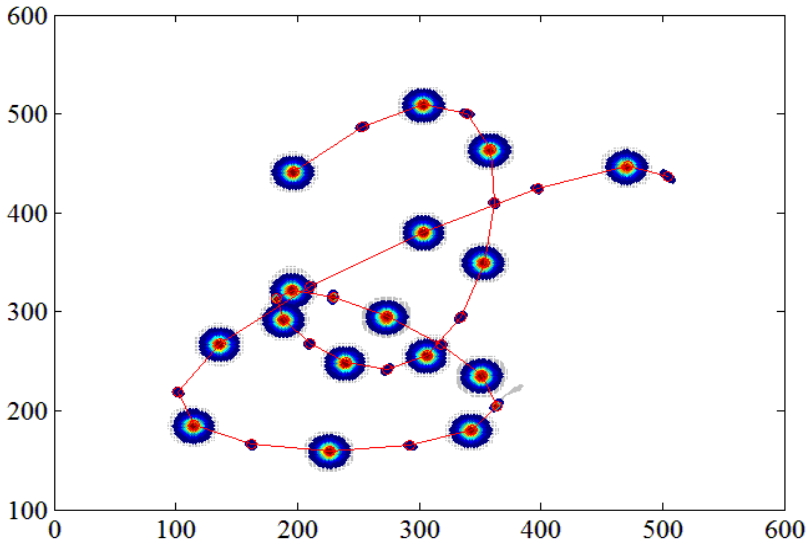


Figure 4.14: Scenario 2- Position-estimate distribution of a mobile node as computed by Factor Graph tracking algorithm for concurrent measurements

For the above scenario we first compared the RMS Error obtained from simulations to the theoretical prediction as per equation (4.50). This is shown in Figure 4.15(a). We then re-derived our theoretical result ignoring the correlation in the translated distance estimates (refer previous section). The deviation in our predicted error when this correlation is ignored was as high as 35 percent in this scenario. This is shown in Figure 4.15(b). We also observe that in Figure 4.15(b) that the true RMS error is consistently larger than our predicted error when the translated measurements are assumed to be independent. This is expected because independent distance measurement would carry more information than when they are correlated.

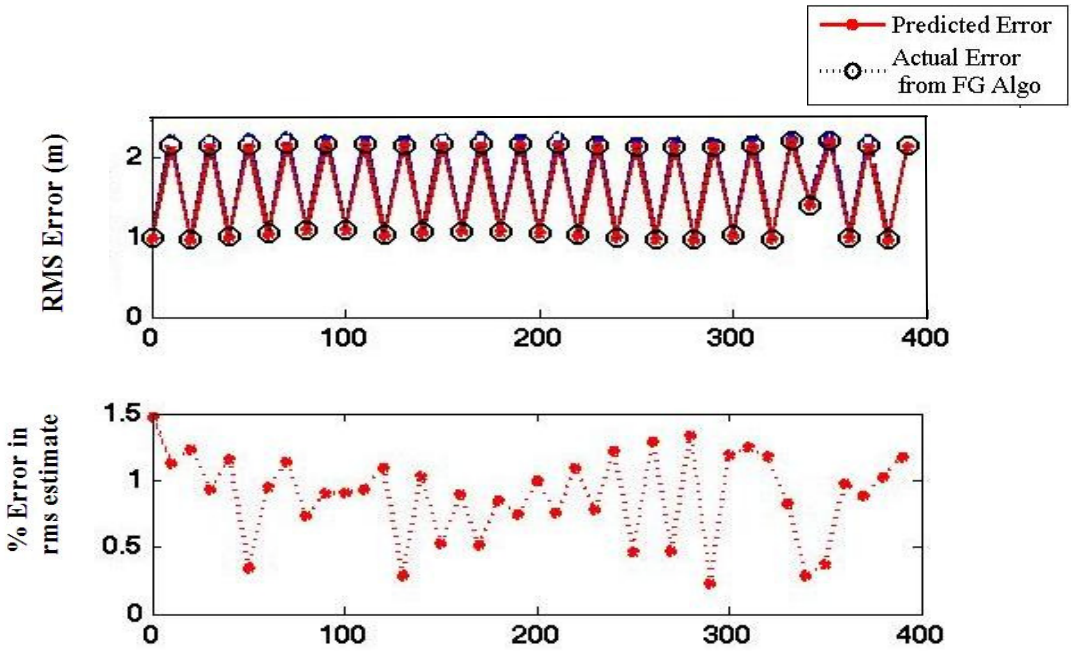


Figure 4.15 (a)

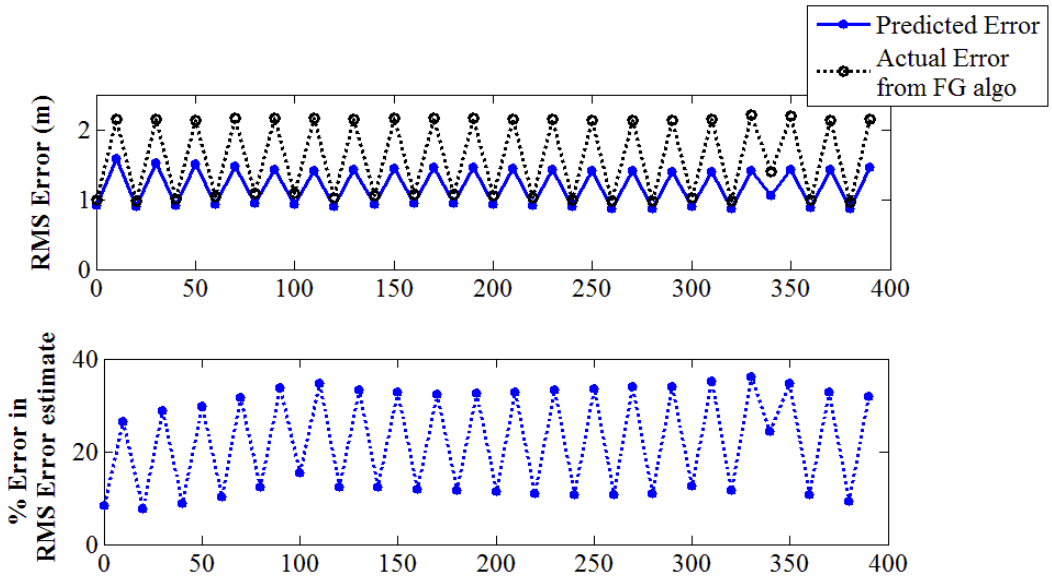


Figure 4.15 (b)

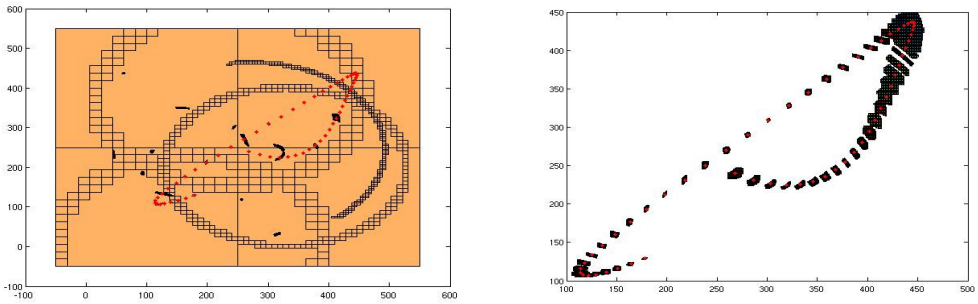
Figure 4.15: RMS Error predicted vs. RMS error from simulations for Scenario-2 taking when (a) Measurement correlation taken into account (b) Measurement correlation ignored

4.5.5 Evaluations in Simulation

Next, we evaluated the performance of our algorithm, with respect to a number of relevant parameters, through simulations. For these simulations, we considered networks consisting of a varying number of submerged vehicles, combined with a few surface beacons. These beacons are elements such as buoys that constantly track their absolute geo-position via GPS. The submerged vehicles and beacons are also assumed to be equipped with acoustic modems. This allows them to estimate their distances with their neighbors via a bi-directional ping message exchange. In accordance with our own experimental results, we select a maximum ranging error of 2m (Chapter 2, Section 2.2). Furthermore, each device carries navigational instruments, yielding heading information with 2 degree accuracy [Kin06] and acceleration estimates with an accuracy of .04 cm/s^2. In our simulations, the nodes are assumed to be loosely synchronized (±0.1s), and repeat this distance estimation every 20 seconds. As a result, each node (beacon or submersible) receives ranging information with all its neighbors at 20 second intervals. For our deployment, the collection of underwater vehicles resides in a volume of 500 m x 500 m with a maximum depth of 20 m. They move on randomly generated smooth paths (generated using spline-interpolated way-points) with constantly-varying velocities between 0 and 2 m/s.

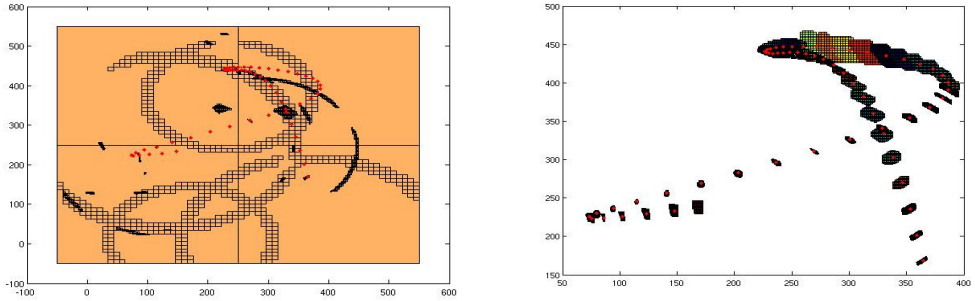
First, we compared the performance of collaborative tracking to that of repeated snapshot localization. In snapshot localization, nodes only use the range estimates available at a particular time instance, to determine their positions collaboratively. This is repeated at each instance to estimate the positions over the tracking interval. We considered a scenario with very few beacons, two in this case.

The network itself consists of 6 independently moving underwater vehicles, and the transmission range of the acoustic modem is chosen as 200m.



(a) **(b)**

Figure 4.16: Trajectory of node1 estimated using: (a) Snapshot Localization (b) Collaborative Tracking



(a) **(b)**

Figure 4.17: Trajectory of node2 estimated using: (a) Snapshot Localization (b) Collaborative Tracking

Figure 4.16(a) shows the distribution of the node position over the tracking interval for one of the unknown nodes when periodic snapshot localization was used, while Figure 4.16(b) does the same for our collaborative algorithm. Similarly, Figure 4.17(a) and (b) show the distribution of the trajectory of a second unknown node using snapshot localization and collaborative tracking respectively. In these figures, the trail of small (red) diamond-shaped markers shows the actual positions of the devices at the different times. The square grids indicate the collection of possible locations the

respective algorithm calculates as the devices residing in. In Figures 4.16(b) and 4.17(b), these grids are actually so small they appear as dark areas; it is clear our collaborative tracking approach is able to provide good overall estimation results. On the other hand, in Figures 4.16 (a) and 4.17(a), for most time instances, there is only one grid occupying the entire area of observation, i.e. the algorithm has no idea where the device is. The arc-like collections of grids are for those times when the node happens to be in communication range of a beacon or another node that has resolved its position. As can be observed, periodic snapshot localization is completely unable to track the devices as they move. The reason is that this network is sparse (at each time, the average density is less than 4) and non-localizable at each specific time instant. Our collaborative tracking, however, is able to efficiently leverage all distance information, even in a network with very few beacons.

We next perform more comprehensive simulations some important parameters. We will compare three different approaches to network tracking: One method is using snapshot localization, explained previously. The second method is tracking each unknown node independently using measurements of the node's motion and only the distance estimates obtained with reference (or beacon nodes). We will refer to this method as beacon tracking. The third strategy is our proposed collaborative scheme that combines all available data in 4 dimensions, which we refer to as collaborative tracking.

We begin by comparing the three schemes when the average number of neighbors of nodes (or network density) is increased. Since network density is hard to precisely quantify in a mobile network, because nodes may not always be Poisson

distributed, we will present our results vs. number of unknown nodes. To increase the 'density' of the network we increase the number of unknown nodes deployed in a fixed geographic region. Specifically, in our simulations nodes are deployed in a cuboid of dimension 600m x 600m x 150m where the maximum depth is 150m. The transmission range of nodes was fixed at 150m. Node moves in smooth curved trajectories within the area of deployment. Three beacons were used to track the network. We increased the number of unknown nodes from 6 to 31 which on an average increased the number of neighbors of any node from 1 to 6. By increasing the number of nodes in this scenario, more information can be used from inter-node distance estimates. In Figure 4.18 we show how the three schemes we consider vary with the number of nodes. As expected, beacon-tracking does not benefit from increasing the number of unknowns since each node is tracked independently and inter-node distances are not used. Snap-shot localization benefits from increasing the inter-node distance estimates but its performance is still quite bad. We observe that our collaborative tracking scheme substantially outperforms the other strategies and can operate well into regimes of sparse connectivity.

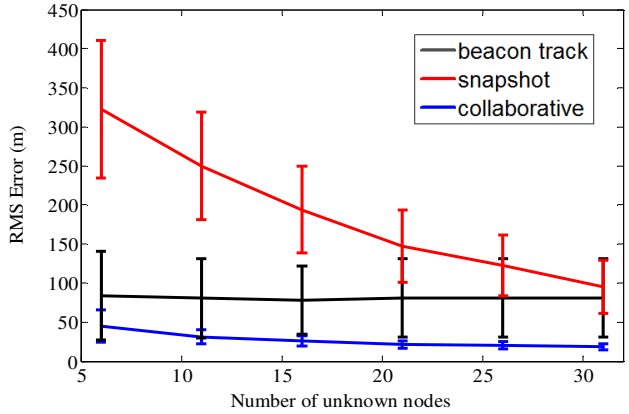


Figure 4.18: Comparative Performance vs. number of nodes

Next we compared the three schemes vs. error in velocity measurements (normalized by the maximum speed of nodes). The simulation scenario is same as before except that we consider only two deployment densities. In the first the number of unknown nodes is 11. This is represented by $\lambda = 2$ in Figure 4.19 and in the second deployment the number of unknowns was 21, represented by $\lambda = 4$ in Figure 4.19. We present our results vs. velocity error/ speed because we observed that if nodes followed the same set of trajectories with double the speed, we would get the same results as if the velocity error was halved. The only condition is that nodes should pass through the same spatial points when distance measurements are obtained.

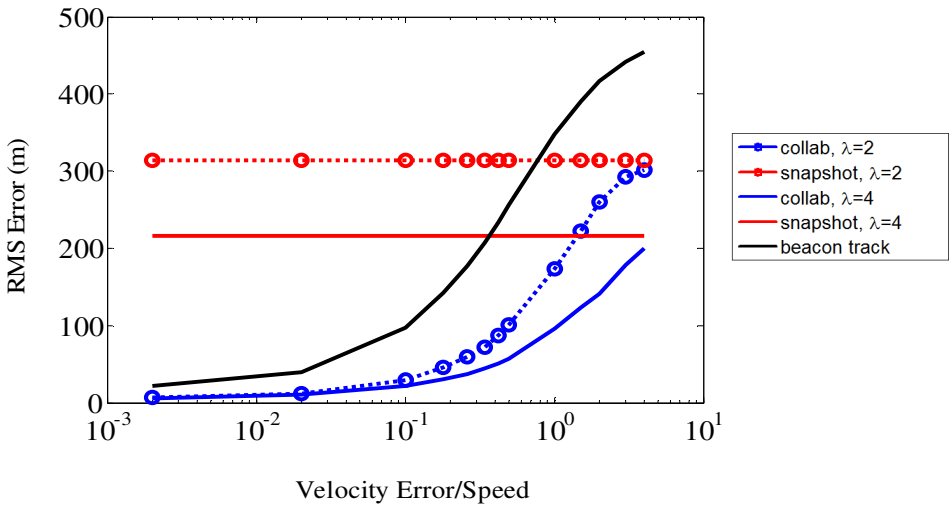


Figure 4.19: Comparative Performance vs. Normalized Velocity Error

Figure 4.19 shows that the performance of snapshot localization does not vary with the ratio of velocity error and speed. This expected because snapshot localization does not use measurements of nodes' motion to estimate position. We also observe that for both node densities, $\lambda = 2$ and $\lambda = 4$, the performance of collaborative tracking comes very close to that of snapshot localization when the ratio of velocity error by

speed is around 5. This essentially shows that if the error in velocity measurements is more than the speed of nodes, velocity measurements do not provide very useful information and we have to solely rely on distance-estimates. We also observe that once this ratio exceeds 0.2, the performance of beacon-only tracking deteriorates quickly. Overall, collaborative tracking is able to leverage from density, mobility and precise motion measurements depending on what is available.

Finally, we compare the three different tracking strategies when the maximum depth of the network is varied. Figure 4.20 shows the localization error averaged over both the duration of tracking and the nodes. We observe that compared to all nodes residing on the surface, both beacon-based tracking and snapshot localization are heavily affected when the maximum depth is increased to 150m. However, collaborative localization is much more robust.

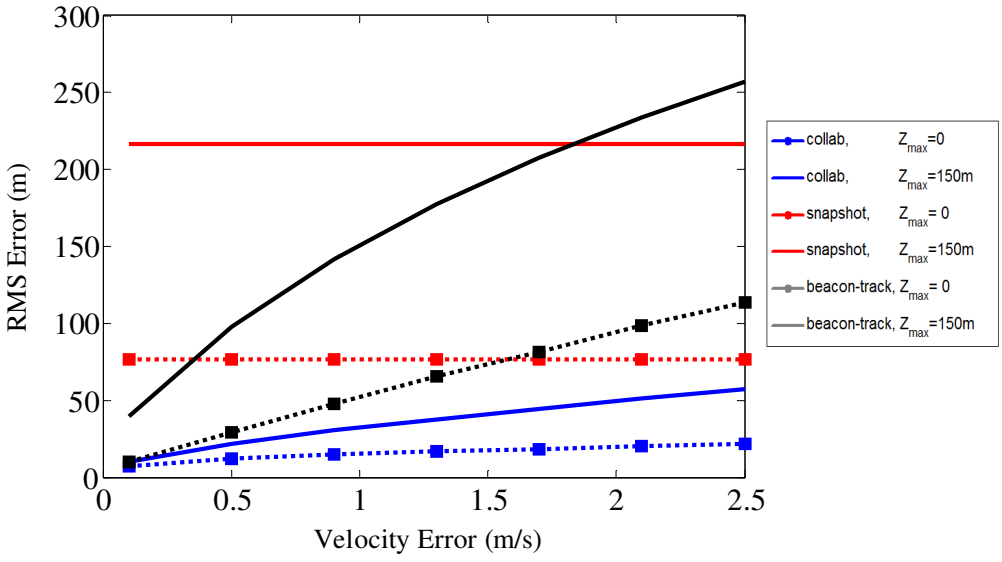


Figure 4.20: Comparative Performance vs. Velocity Error and Max. Depth

4.5.6 Results from Experimental Data

We have tested our tracking algorithm using data obtained from an experimental test-bed. Since conducting such an experiment underwater is quite difficult because the true position of nodes is hard to establish, we performed controlled experiments on a terrestrial network. The goal was to ensure that our algorithm gave us expected results when real data was used.

Our experimental setup consists of a number of static beacon nodes and one mobile node. We used the Telos mote platform for our experiments. The motes were programmed to obtain distance estimates acoustically from one-way time-of-flight measurements. The on-board accelerometer was used as the only measurement of the node's motion.

All data collected by the mobile node, namely the inter-node distances and measurements of acceleration were communicated to a central base-station for processing. The acceleration measurements which were obtained every 1ms and were aggregated over a period of 2s using the procedure explained in Section 4.5.2.

We will present two experimental scenarios. The first one is a static scenario where the node, **U** with unknown position is kept stationary at position (0, 3). Four beacons were placed at positions (1, 1), (-1, 2), (-1, 4) and (1, 5). Node **U** measured its acceleration periodically. It performed ranging by broadcasting a time-stamped message every 8s. The receive time of the message was recorded by beacons that were within range and were used to obtain an estimate of distance. The acoustic range of the motes is about 3m.

Figure 4.21 shows the position estimate of node **U** over 20 time steps when only the range estimates at each time were used to localize the node. Here only spatial combining of data was performed. As before, we refer to this method as snapshot localization. In Figure 4.21, the time instances for which no estimate is available corresponds to the times where ranging was not performed.

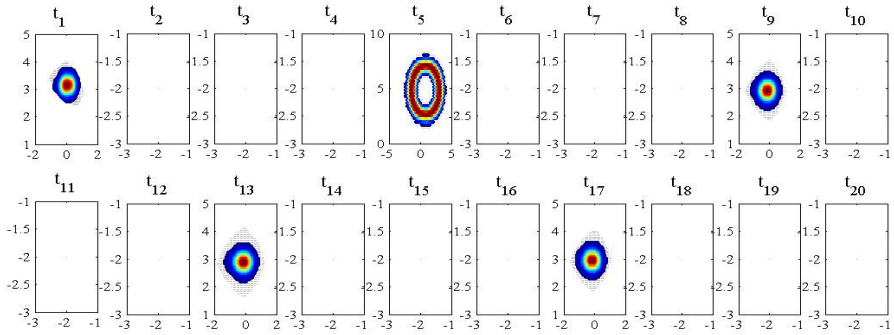


Figure 4.21: Static Scenario- Snapshot Localization

To counter the effect of outliers, we use the signal strength of the received signal to annotate each range estimate with a confidence. We used three levels of confidence (1 being the lowest and 3 the highest). For the results presented in Figure 4.21, only measurements with confidence greater or equal to a level 2 were used. Consequently, we observe that at time t_5, only one measurement was available and the node could be anywhere on a Gaussian Ring (refer Definition 4.5.3.1 of Section 4.5.3).

Next, we used all the data obtained from the experiment, namely, the inter-node distance estimates in the interval (t_1, t_{20}) as well as the acceleration measurements aggregated over 2s intervals within the same period. We ran our factor-graph algorithm on this data. The results are shown in Figure 4.22. The figure shows that the position of the node could be estimated for all time instances. Further, our

estimate at time t_5 is much better than in Figure 4.21 where only snapshot localization was performed.

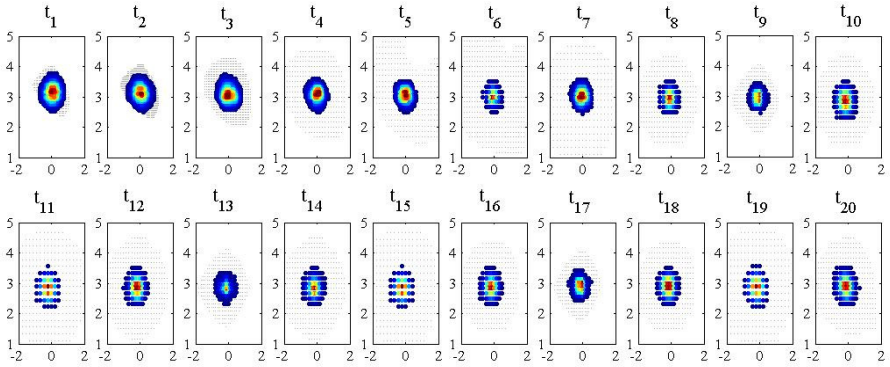


Figure 4.22: Static Scenario – Spatio-temporal Combining

We also performed experiments for a mobile scenario. The beacon positions were same as in the static case, however, this time node **U** moved in a straight line from position (0, 0) to position (0, 5). Range estimates were obtained, while the node was moving, by having the mobile node send out a broadcast message every 8s. We conducted multiple runs of this scenario and our results were consistent in all runs. Here we will present one such run. Figure 4.23 shows the estimated distribution of the position of node **U** for nine time instances using snapshot localization. In this case, the node had enough measurements at times t_5 and t_9 to localize itself with Gaussian error. Figure 4.23 also shows that distance estimates were available only at those two time instances.

We next used only the acceleration measurements and applied dead-reckoning to estimate the position of the mobile node in the interval (t_1, t_9). These results are shown in Figure 4.24. To apply dead-reckoning we need to have an initial position fix

which is (0, 0) in this case. The acceleration measurements were used to estimate the positions at subsequent times.

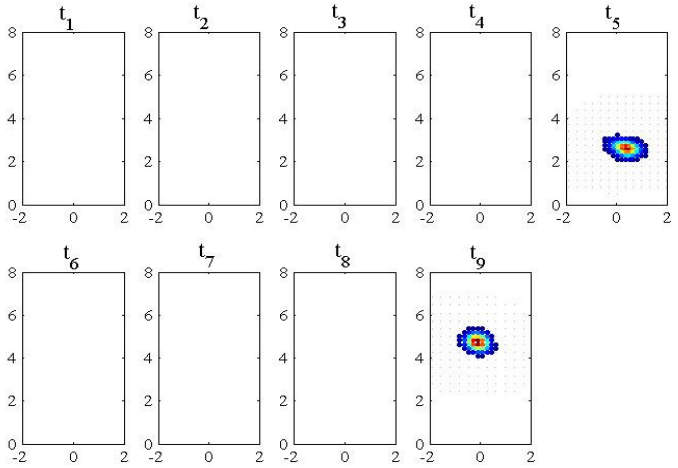


Figure 4.23: Mobile Scenario- Snapshot Localization

The accelerometer on the motes is not really designed for location tracking applications and gives us fairly erroneous estimates of the true acceleration. This can be observed in Figure 4.24 where dead-reckoning results in position estimates with large uncertainty. We also observe in the same figure that the final position of the node as estimated by dead-reckoning is at (0, 2) while the true position at that time was (0, 5).

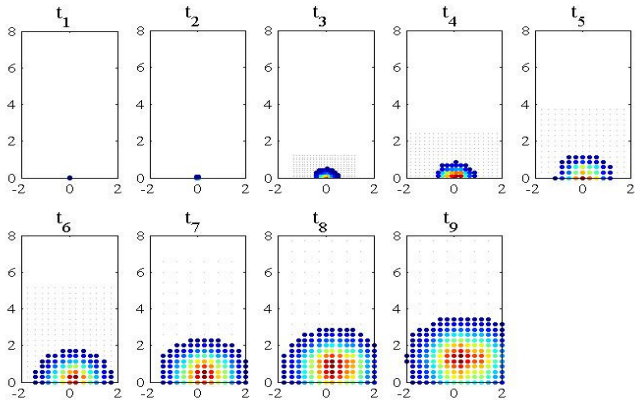


Figure 4.24: Mobile Scenario- Tracking using Dead Reckoning

Finally, we used our collaborative algorithm that combines both distance estimates and acceleration measurements. The results are shown in Figure 4.25. We observe that the estimate of the node position has substantially improved using our tracking algorithm. Although the estimate of the motion of the node was quite inaccurate, as shown in Figure 4.24, this was corrected when our collaborative algorithm was used.

The results of the mobile scenario essentially demonstrate that our proposed tracking algorithm can estimate positions accurately even with low-cost platforms that may have imprecise motion sensors. A few accurate measurements give us much better estimates than many measurements that contain outliers. For best performance the error in measurements should be well characterized. For example, we could obtain fairly good estimates of position when we knew that the acceleration measurements were more inaccurate than the distance-estimates.

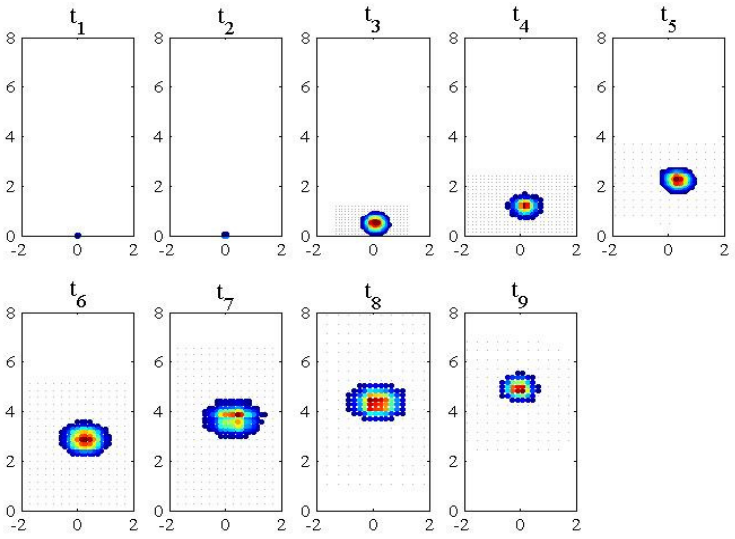


Figure 4.25: Mobile Scenario – Spatio-temporal Combining

4.6 Leveraging Mobility in Single Beacon Systems

In this section we will specifically investigate the effect of relative motion between communicating nodes on the performance on tracking. We would like to find out how much error reduction is possible when measurements are made with the same beacon node, when there is relative motion between nodes. We will use entropy of the distribution of our final position estimate as the performance metric. The entropy of the distribution of the position estimate of a node is a measure of the extent of uncertainty in the estimate. The reason for choosing entropy as our performance metric as opposed to the RMS error is that for the scenarios under consideration we are likely to be operating in a regime where the position estimate distributions are highly non-Gaussian and multi-modal.

To answer the question about the benefit of self-measurements, namely, repeated measurements between the unknown and the same neighbor node, we will quantify the reduction in entropy as a result of additional measurements.

4.6.1 Quantifying the Effect of Mobility

We will first derive the entropy for the position estimate at some time instance when one distance measurement is made with a neighbor node that knows its position (also called as a beacon) and also for the case when an additional distance measurement is obtained with the same neighbor node, at a later time. Since the entropy is a measure of uncertainty, the reduction in entropy quantifies how much we have gained (in terms of performance) from the additional measurement.

For any generic distribution of the position estimate of the unknown, $p_U(r,\theta)$ the entropy can be obtained as:

$$\mathrm{H}_U = \iint_{r,\theta} -p_U(r,\theta)\cdot\ln\left(p_U(r,\theta)\right)dr\ d\theta \qquad (4.51)$$

Let the first distance estimate, s_1 be obtained with the neighbor node at time t_1. The error in the distance measurement is zero mean Gaussian with standard deviation σ_R. Without loss of generality, we set the position of the beacon at time t_1 to be (0, 0). From measurement s_1, our estimate of the pdf of the position of the unknown at time t_1 is a Gaussian ring with center point (0, 0) (Refer to Section 4.5.3 for definition of the Gaussian ring). We can describe the non-normalized probability distribution of the position estimate of the unknown node at time t_1 in polar coordinates as:

$$f_1(r,\theta) = \frac{1}{2\pi\cdot\sigma_R}\exp^{-g_1(r,\theta)} \qquad (4.52)$$

Where, $\quad g_1(r,\theta) = \dfrac{(s_1 - r)^2}{2\cdot{\sigma_R}^2}$

To obtain the normalized distribution $f_1(r,\theta)$ must be divided by N_0 defined as:

$$N_0 = \iint_{r,\theta} f_1(r,\theta)\ dr\ d\theta \qquad (4.53)$$

From equation (4.51) to (4.53), the entropy of the position estimate at t_1 when only the distance measurement at t_1 is available is given by:

$$
\begin{aligned}
\mathrm{H}_1 &= \iint_{r,\theta} -\frac{1}{N_0} \cdot f_1(r,\theta) \cdot \ln\left(\frac{f_1(r,\theta)}{N_0}\right) dr\, d\theta \\
&= \iint_{r,\theta} -\frac{1}{N_0} \cdot f_1(r,\theta) \cdot \left\{\ln\left(\exp^{-g_1(r,\theta)}\right) - \ln\left(2\pi\sigma_R N_0\right)\right\} dr\, d\theta \\
&= \iint_{r,\theta} -\frac{1}{N_0} \cdot f_1(r,\theta) \cdot \left\{-g_1(r,\theta) - \ln\left(2\pi\sigma_R N_0\right)\right\} dr\, d\theta \qquad (4.54) \\
&= \frac{1}{2\pi\sigma_R N_0} \cdot \iint_{r,\theta} \exp^{-g_1(r,\theta)} \cdot g_1(r,\theta)\, dr\, d\theta + \frac{\ln\left(2\pi\sigma_R N_0\right)}{N_0} \cdot \iint_{r,\theta} f_1(r,\theta) dr\, d\theta \\
&= \frac{1}{2\pi\sigma_R N_0} \cdot \iint_{r,\theta} \exp^{-g_1(r,\theta)} \cdot g_1(r,\theta)\, dr\, d\theta + \ln\left(2\pi\sigma_R N_0\right)
\end{aligned}
$$

We denote the position of node **U** at any time t as $P_U(t)$. Now, let the second distance measurement, s_2 be obtained at time $t_2 = t_1 + \Delta t$. The distribution of $P_U(t_2)$ as a result of s_2 is another Gaussian ring which we denote as G(t_2). If the beacon is displaced by ξ_B in the interval Δt, G(t_2) is centered at ξ_B and has a radius with mean s_2 and std. deviation σ_R. Now, to study the effect of measurement, s_2 on the position estimate of the node at time t_1 we need to translate this measurement to time t_1 given measurements of the velocity of the known node.

Suppose node **U** was displaced by ξ_U in the interval Δt. We apply a spatial translation, $-\xi_U$ to G(t_2) to translate it to time t_1. As a result, the measurement s_2 when translated to time t_1 is a Gaussian ring with center ξ_{UB}, where ξ_{UB} is the relative displacement of the beacon and unknown in the interval Δt:

$$\xi_{UB} = (\Delta x, \Delta y)^{\mathrm{T}} = \xi_B - \xi_U \qquad (4.55)$$

Now ξ_B is known since the neighbor node knows its position at all times. However, ξ_U can only be estimated from velocity measurements that have some uncertainty. This uncertainty adds to the ranging error. So, the error in the second

measurement after it is translated to t_1 is zero mean Gaussian with standard deviation σ_δ. If k independent velocity measurements were used to translate G(t_2) to time t_1, σ_δ is defined as follows:

$$\sigma_\delta{}^2 = \sigma_R{}^2 + \frac{1}{k} \cdot \left(\sigma_v \cdot \Delta t\right)^2 \tag{4.56}$$

Now, the non-normalized probability distribution for P_U (t_1) in polar coordinates given the distance measurement made at t_2 alone is:

$$f_2\left(r,\theta\right) = \frac{1}{2\pi \cdot \sigma_\delta} \exp^{-\frac{\left(s_2 - z(r,\theta)\right)^2}{2\cdot\sigma_\delta{}^2}} \tag{4.57}$$

$$z\left(r,\theta\right) = \sqrt{\left(r\cos\theta - \Delta x\right)^2 + \left(r\sin\theta - \Delta y\right)^2}$$

Since the two distance measurements are independent, the distribution of P_U (t_1) as a result of both measurements s_1 and s_2 is given by the product of its distribution given each measurement. From equations (4.52) and (4.57) we obtain the distribution of the unknown position at time t_1 given both s_1 and s_2 :

$$f\left(r,\theta\right) = f_1\left(r,\theta\right) \cdot f_2\left(r,\theta\right) = \frac{1}{4\pi^2 \cdot \sigma_\delta \cdot \sigma_R} \exp^{-g(r,\theta)} \tag{4.58}$$

$$g\left(r,\theta\right) = \frac{1}{2} \cdot \left\{ \frac{\left(s_1 - r\right)^2}{\sigma_R{}^2} + \frac{\left(s_2 - z\left(r,\theta\right)\right)^2}{\sigma_\delta{}^2} \right\}$$

Let N_1 be the normalization factor for f (r, θ) defined as:

$$N_1 = \iint_{r,\theta} f\left(r,\theta\right)\, dr\; d\theta \tag{4.59}$$

From equations (4.51), (4.58), (4.59), the entropy of the position estimate of $P_U(t_1)$ when both measurements s_1 and s_2 are used is:

$$\begin{aligned} \mathrm{H}_2 &= \iint_{r,\theta} -\frac{1}{N_1} \cdot f(r,\theta) \cdot \ln\left(\frac{f(r,\theta)}{N_1}\right) dr\, d\theta \\ &= \iint_{r,\theta} -\frac{1}{N_1} \cdot f(r,\theta) \cdot \left\{\ln\left(\exp^{-g(r,\theta)}\right) - \ln\left(4\pi^2\sigma_R\sigma_\delta N_1\right)\right\} dr\, d\theta \qquad (4.60) \\ &= \frac{1}{4\pi^2\sigma_R\sigma_\delta N_1} \cdot \iint_{r,\theta} \exp^{-g(r,\theta)} \cdot g(r,\theta)\, dr\, d\theta + \ln\left(4\pi^2\sigma_R\sigma_\delta N_1\right) \end{aligned}$$

The change in entropy of the position estimate of node **U** at t_1 due to the additional measurement, s_2 is:

$$\Delta\mathrm{H} = \mathrm{H}_2 - \mathrm{H}_1 \qquad (4.61)$$

Where, H_1 and H_2 are defined in equations (4.54) and (4.60) respectively.

We will now use this result to study how a beacon should move with respect to an unknown node so as to maximize the reduction in entropy (-ΔH).

4.6.2 Beacon Motion Strategy

In this section we will answer the following question: *If a beacon had control over its mobility, how must it move relative to the unknown node so that the information obtained from consecutive measurements with the same beacon is maximized?*

To answer this question we will use the results obtained in the previous section. Specifically, using equations (4.51) - (4.61), we will plot the reduction in entropy for all possible relative displacements of nodes in the interval between two consecutive measurements.

Suppose at time t_1 the beacon is at a distance D from the unknown when a measurement is obtained. Let the second distance measurement be obtained at time t_2. Suppose the beacon moves such that the norm of the relative displacement of the

nodes in the interval (t_1, t_2) is L and the direction of displacement is β. Due to the time elapsed between the two measurements, translating the second measurement to t_1 results in an error growth σ_δ (See equation 4.56). For a fixed value of σ_δ we will observe the reduction in entropy when both L and β are varied. Figure 4.26 shows the change in entropy as a result of the second measurement, ΔH (obtained from equation (4.61)) vs. the direction of relative displacement β. Maximum reduction in uncertainty (or entropy) is obtained when ΔH is lowest. We have adopted the following convention: $\beta = 0$ corresponds to a relative displacement where the nodes had moved away from each other and $\beta = 180$ when the nodes had moved towards each other. Figure 4.26 shows that the larger the distance L, the more we gain from consecutive measurements with the same beacon. The results also show that as long as the distance moved by the beacon is small relative to the initial distance between nodes, D, the maximum reduction in entropy is observed when the beacon moves either towards or away from the unknown. However, when L is close to D, the optimal motion is when the beacon moves towards the unknown. In fact, the results in Figure 4.26 suggest that the best strategy is that the beacon moves closer and closer to the unknown until it has attached itself to the unknown.

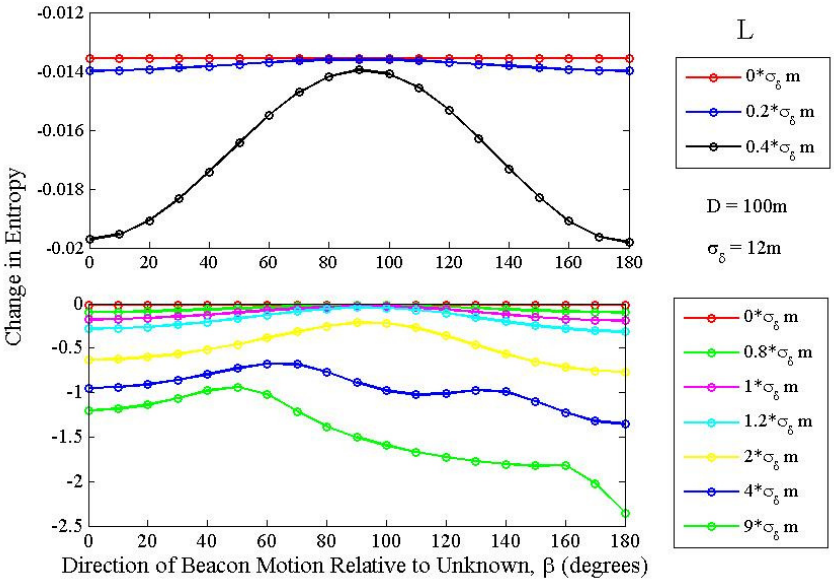


Figure 4.26: Entropy reduction vs. Relative Beacon Motion

4.6.3 Enhancing Performance in Networks with Static Beacons

It has been shown in earlier work that in a static network, introducing mobility in the beacon nodes improves localization performance [Sic04]. We will look at a slightly different problem in this section. Suppose that we had a system where the beacons were stationary. Examples would be LBL systems or even cellular systems where the base-station is used as a position reference. Could we improve the localization performance by introducing mobility in the unknown nodes? This is an interesting problem because unlike systems with mobile beacons, nodes with unknown positions do not have a perfect estimate of their motion which results in an error growth in distance measurements when translated to a common time. However, if the nodes are known to be static, the error in measurements does not grow with elapsed time. Suppose there is only one stationary reference in the system. If the nodes were static and this was known, measurements with the beacon would result in identical

Gaussian rings. Combining these measurements would only reduce the thickness of the ring (because the measurements are independent), however, an unknown node would only be able to know that it is somewhere on a ring even when multiple measurements were combined. We can obtain the reduction in entropy as a result of two such measurements for a static network where nodes are known to be static. In Figure 4.27, this shown by a solid red line. We compare this case to the case where mobility is introduced in the unknown node. Specifically the unknown moves a distance L between consecutive measurements. Since nodes measure their velocity with some error, the error growth between consecutive measurements is σ_δ (as defined by equation 4.56). The rest of the curves in Figure 4.27 correspond to this mobile case, showing the change in entropy vs. the direction of motion of the unknown relative to the beacon for different values of L. We observe that once the unknown moves more than 2.3 times the error growth between consecutive measurements, the uncertainty reduction will be better than the static case where nodes are known to be static. Figure 4.27 essentially shows under this condition we can benefit from introducing mobility in the unknown nodes, even if the velocity of the nodes cannot be measured precisely.

To validate our analytical expressions for the change in entropy, equations (4.51) - (4.61), we also obtained the reduction in entropy via simulations. The results from simulations are shown as circles in Figure 4.27. To generate the simulation results we used our factor-graph solution to obtain the distribution of the positions. From Figure 4.27, our simulation results closely match the analytical expressions for the change in entropy.

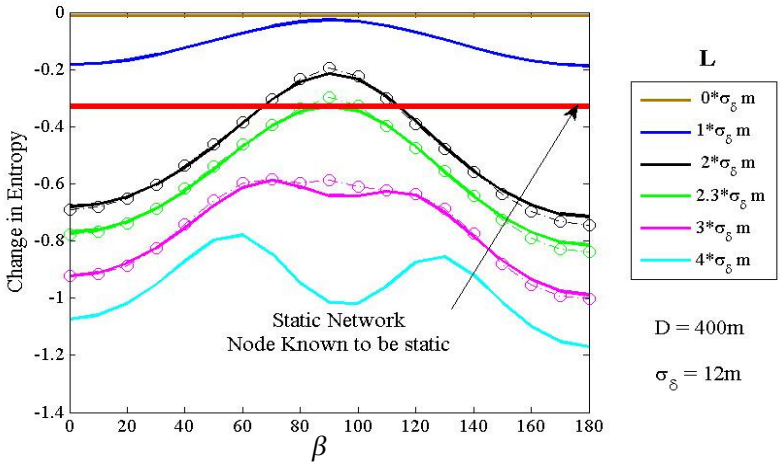


Figure 4.27: Performance improvement by introducing mobility in unknowns

4.7 Related Work

There is a vast body of existing work on underwater navigation techniques for tracking individual vehicles (e.g., [Bla03] [Rom05] [Sto02]). These often rely on stochastic estimators such as (extended) Kalman filters, which are a special instantiation of factor graphs. While some methods rely on pure dead-reckoning, others also leverage direct links to static anchor nodes, albeit in a non-networked fashion [Cor07]. It has also been proposed that nodes can periodically resurface to obtain position fixes [Ero07]. Recently, localization has been studied for networked underwater vehicles, but reported work either does not consider the effects of mobility [Cha06] or uses a distance based correlation model for the movement of nodes [Zho07].

In the realm of terrestrial systems, there exists much work on collaborative self-localization for static sensor networks, which suffers from the same sparsity problems. On the other hand, for non-networked tracking of mobile robots, techniques such as particle filters have been proposed [She05] [Fox98]. However, integration of

networking and tracking into collaborative techniques for jointly tracking a collection of mobile vehicles has received considerably less attention. One of the most relevant works is LOCALE, which is designed for collaboratively tracking nodes in sparse mobile terrestrial networks (ZebraNets) [Zha08]. It allows merging of motion estimates with information obtained from encounters with other unknown nodes. However, it relies on each node also measuring angle of arrival (as it relies on Gaussian statistics), which is not realistic in mobile underwater platforms (hydrophone arrays are expensive and bulky). As such, this work cannot be applied to networks of submersible vehicles that can only measure distance.

Finally, the use of factor-graphs for localization was also proposed by Wymeersch et al. [Wym08] [Wym09], independently from our own prior work [Mir08] [Mir09], but for terrestrial UWB networks. However, it does not include generic information from navigational instruments, but only velocity estimates (which are not available on typical submersible platforms). In our work, velocities are introduced as hidden variables in the factor-graph and are used to incorporate different types of motion information into the framework. Also, none of this prior work on localization has our analytical results on performance.

4.8 Acknowledgements

This chapter is, in part, a reprint of material published in the Proceedings of the ACM International Workshop on Underwater Networks in conjunction with ACM SenSys 2009 under the title “Collaborative Tracking in Mobile Underwater Networks”. I was the primary researcher and author and Curt Schurgers supervised the research.

CHAPTER 5

RANGE-ONLY LOCALIZATION

5.1 Introduction

In this chapter we will investigate position estimation for systems where devices are not equipped with motion sensors and solely rely on acoustic ranging. In addition, applications that use such systems do not require continuous tracking. Position information is only needed when data-samples are collected and the error in position estimates must be minimized only at these particular times. We will focus on the problems that arise when a concurrent map of node positions (or a snapshot) is required at the times samples are collected.

At each localization time T_{loc}, self-localization operates on inter-node distance estimates. Now, the intrinsic mobility of the underwater environment creates some very specific challenges with respect to range estimates collected around T_{loc}, which is illustrated in Figure 5.1. Ideally, all range estimates should be acquired at the target localization time T_{loc}. However, because communication occurs over a shared channel, medium access control (MAC) has to ensure that excessive collisions are avoided. As a result, the gathering of ranging information actually occurs over a short time epoch T around the target localization time. The problem is that mobility causes the node positions to change significantly during this time epoch. This is illustrated in Figure 5.1 by showing the positions of three nodes, at the three time instances at which the range for each pair is estimated. We observe that ranging (and topology)

becomes ambiguous and inconsistent, which results in error in position estimates obtained from self-localization.

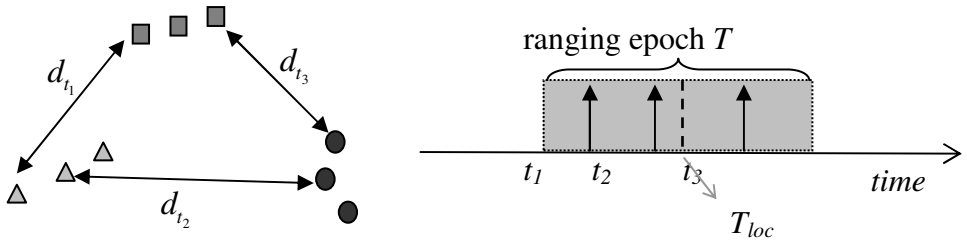


Figure 5.1: Position uncertainty due to measurement delay

In traditional terrestrial systems, this effect hardly ever comes into play. However, as we will illustrate shortly, in underwater networks, it can be very significant and the reason is the very nature of the underwater communication environment: low data rate, long propagation delays and inherent mobility. The first goal of this chapter is to show how we can limit the resulting error on self-localization performance via better data-combining or estimation techniques. However, as the network density increases, the problem can become more acute. This is because the delays in medium access grow substantially with the number of neighbors.

It has been shown that contender-counting can be performed in underwater networks using tone-based techniques [Sye08]. By strategically reducing the number of contenders, we could both reduce the large delays in medium access and also the energy-consumption of nodes. Our second solution, which we have dubbed *Sufficient Distance Map Estimation* (SDME), achieves this goal using the concept of graph rigidity. SDME protocol finds which device needs to collect what data, while minimizing the negotiation between devices.

5.2 Effect of Channel Characteristics on Localization Performance

Communication underwater has long been known to be challenging. Acoustic channels are characterized by long propagation delays and acoustic modems can achieve relatively low data rates [Hei06]. For example, the micro-modem developed by WHOI [WOO] transmits at 80 bps, and even short data packets take around a second or more to complete. With speed of sound underwater being around 1500 m/s, propagation delays can also be in the order of a second. As such, to avoid collisions in time, MAC protocols end up spacing competing transmissions over multiple tens to hundreds of seconds. For example, with packets of 10 bytes and distances of 500 m, CSMA backoff needs to be around 300s to limit collisions to less than 5% in moderately dense networks[3]. This means that the ranging epoch T is in the order of 100 seconds. On the other hand, current speeds vary between 0.1 to 1 m/s, while relative speeds of guided AUVs may even exceed this. As a result, displacement in node positions during localization can range from a few tens to up to a hundred meters.

This displacement (essentially a ranging error due to ambiguity) is much larger than the intrinsic ranging errors of the system. From the experimental results presented in Chapter 2, when stationary, the ranging error is only a few meters, and consistent across various scenarios. Therefore, and as we will show in Section 5.3, mobility causes significant degradation of self-localization performance in underwater networks, if traditional techniques are utilized.

[3] Although CDMA could allow simultaneous transmissions, it is difficult to assign orthogonal codes in a mobile network with low overhead. Also it does not allow sending and receiving at the same time (and therefore does not allow concurrent ranging either). We assume a CSMA or TDMA style MAC protocol, as is common for most acoustic modems.

The problem of node mobility during the ranging epoch is especially relevant to such short range systems since errors are large enough to skew our estimate of the network topology. Even if nodes move, it is important to have consistent position estimates across sampled data, as these sensor networks are data-centric (i.e. the correlation between data samples is important, not the identity of the specific device that collected the data; as such tracking individual devices is not important).

Our first goal is to devise a collaborative localization scheme that effectively compensates for node motion within the ranging epoch, around the specific localization time of interest T_{loc}. In the next section, we formally define the localization problem and introduce a solution strategy.

5.3 Combating the Drawbacks of Low Rate Modems

As described earlier, nodes perform ranging with their neighbors during the localization epoch, T. We denote the set of measurements taken between node pairs (i, j) at any time t as d_{ij}^{t}. The collection of all distance measurements obtained in the interval $(0, T)$ is denoted by $\mathbf{z_D} = \{d_{ij}^{t}\}_{t\in(0,T)}$. By taking into account the fact that nodes are moving during the period these distance measurements are collected, we intend to improve the localization performance. The problem is to determine the ML estimate for the position of each node i at the target localization time, T_{loc}, given all distance measurements and only an upper bound on the speed of nodes. We will refer to this problem as *concurrent localization* and formally define it as follows:

Estimate: $\mathbf{P}^{*} = \left\{ P_i^{*}(T_{loc}) \right\} \forall i = 1,..N$

Where,

$$P_i^*(T_{loc}) = \arg\max \iiint_{\Theta_i} p(\mathbf{P}, \mathbf{P}_{aug} \mid \mathbf{Q}, z_D, s_{\max}) \cdot d\Theta_i \qquad \forall i = 1,..N \tag{5.1}$$

$$\Theta_i = \{\mathbf{P} \setminus P_i(T_{loc})\}$$
$$\mathbf{P} = \{P_i(T_{loc})\} \forall i = 1,...N$$
$$\mathbf{P} = \{P_i(t_k), P_j(t_k)\} \forall i, j, t_k \text{ s.t. } d_{ij}^{t_k} \in z_D$$

s_{max} is the maximum speed of nodes.

z_D is the set of distance measurements obtained between nodes in the interval $(0,T)$ and has been formally defined in Chapter 4, equation (4.1).

Q is the positions of all reference nodes in the interval $(0,T)$ which was introduced in Chapter 4, Section 4.1.

Each distance measurement between a node pair introduces a correlation between their positions at the time the measurement was obtained. As a result, the likelihood function for the position of any node i at the target time T_{loc} can only be determined from the joint distribution of the position of nodes at the times ranging was performed. This is reflected in equation (5.1) where we have augmented the set of states that we intend to estimate, namely **P** to include $\mathbf{P}_{aug}$. $\mathbf{P}_{aug}$ is the position of all node pairs at the time instances distance-measurements were obtained. Now, the form of this problem is very similar to the generic tracking problem defined in Chapter 4. So, we can solve it using the same framework. In fact, the concurrent-localization problem is a special case of the more general tracking problem, with some simplifications.

As explained in Chapter 4, we would first need the factor-graph representation of the problem. This is shown in Figure 5.2. This is a simplified version of the generic factor-graph presented in Figure 4.5, however at a shorter temporal scale and with some states eliminated. Since measurements of nodes' motion parameters are not available, we do not need to estimate the velocity of nodes. In Figure 5.2, the position of any node i at time-step k is denoted by a circle, which we refer to as the *state-variable* $P_{i,k}$. All state-variables that belong to $\mathbf{P_{aug}}$ are shown as hidden variables. As before, the progression of node positions in time is shown in the horizontal dimension. The vertical dimension indicates this time progression for each unknown node. The graph is constructed such that the unknown state of interest P_{i,K^*} , where $t_{K^*} = T_{loc}$ is one of the K unknown states being estimated for each node i. We have shown the graph representation for only 3 nodes not to overload the figure.

The likelihood of a distance measurement between a node pair is given by functions of type $\boldsymbol{w_1}$. If a measurement is obtained with a beacon the corresponding function-node is single-ended i.e. it has a link only to the unknown state-variable as shown in Figure 5.2. A zero mean Gaussian model is well suited for ranging error, equation (5.2). However, our solution can accommodate any other model.

$$w_1(P_{i,k}, P_{j,k}) = p(d_{ij}^{t_k} \mid P_{i,k}, P_{j,k}) \sim N\left(\left\| P_{i,k} - P_{j,k} \right\| , \sigma_R \right) \qquad (5.2)$$

Where, σ_R is the standard deviation of ranging error.

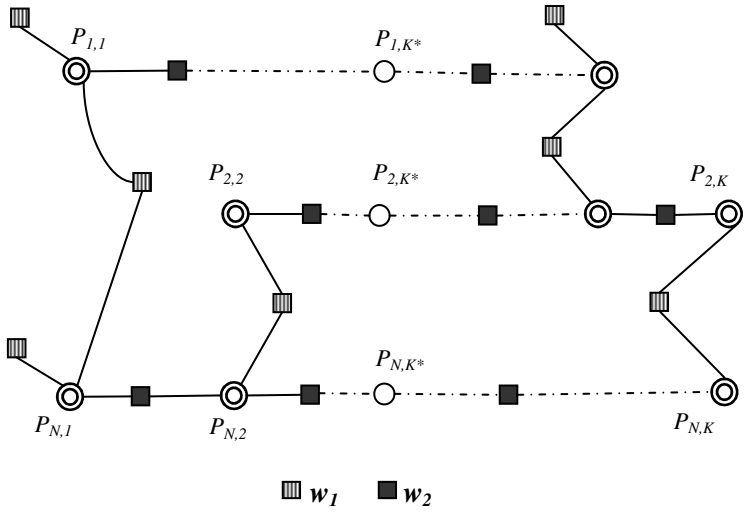


Figure 5.2: Factor-graph representation of the concurrent localization problem

The probabilistic model that describes the evolution of a node's position in time is given by function-nodes of type w_2. We model node speeds to be uniformly distributed between (0, s_{max}), equation (5.3).

$$w_2(P_{i,n}, P_{i,m}) = \left\{ \begin{array}{c} \dfrac{1}{\pi \cdot \left(s_{\max} \cdot |t_n - t_m|\right)^2}, \ if \ \ \left\| P_{i,n} - P_{i,m} \right\| \prec s_{\max} \cdot |t_n - t_m| \\ 0, \qquad otherwise \end{array} \right\} \qquad (5.3)$$

Now that the factor-graph model for the problem has been obtained, the sum-product algorithm can be applied to compute the distributions of state-variables via iterative message passing. The actual messages generated by nodes and the operation of the algorithm have been described in detail in Chapter 4, Sections 4.4 and 4.5.1.

5.3.1 Simulation Results

To evaluate the performance of our localization scheme, we performed simulations in Parsec [PAR]. All simulation parameters are summarized in Table 5.1. We deployed nodes over a 3D region where they move with current streams of equal

thickness. The velocity of these streams varies with depth which is one of the commonly used models [Pom06]. The speed of each layer is independently chosen between (0, s_{max}).

Distance estimates are obtained from broadcast transmissions. However prior to transmitting, nodes choose a random back off between 0 and $T_{backoff}$ to avoid collisions. Four surface beacons are used to localize a network of 15 nodes. The transmission range is R with some variation. We estimate the position of nodes in 2D since nodes know their depth from pressure measurements.

Figure 5.3 shows a view of the network when looking from above and elucidates the problem we are trying to address. In Figure 5.3, the movement of nodes during the time all distance measurements are obtained is shown by blue dots. The actual positions of unknown nodes at a target localization time are represented by stars. Beacon positions at that time are shown as triangles. We observe that nodes are displaced between 10m to 180m during localization which is considerably large compared to their transmission range. For the above scenario we used our proposed method to localize nodes at a target localization time. However, to compare the performance of our scheme we also estimate node positions using a robust self-localization algorithm, Multi-dimensional Scaling (MDS) [Shan03]. To enhance the performance of MDS we chose for each node pair distance measurements that were closest to the target time.

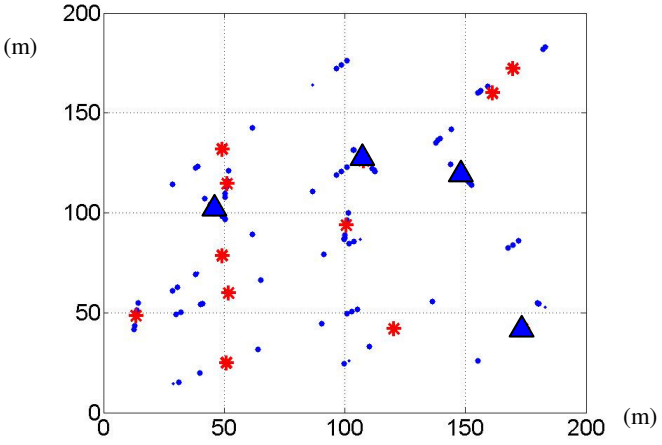


Figure 5.3: Movement in node positions during localization

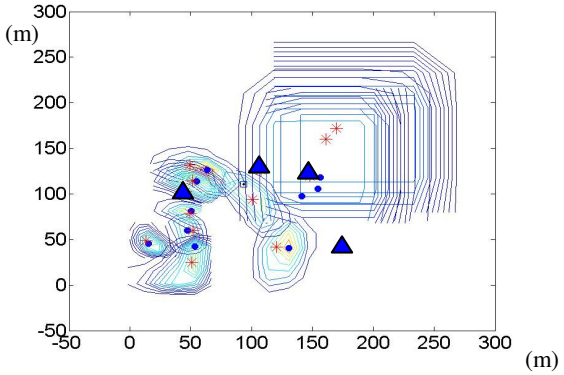


Figure 5.4: Probability contours of position estimates

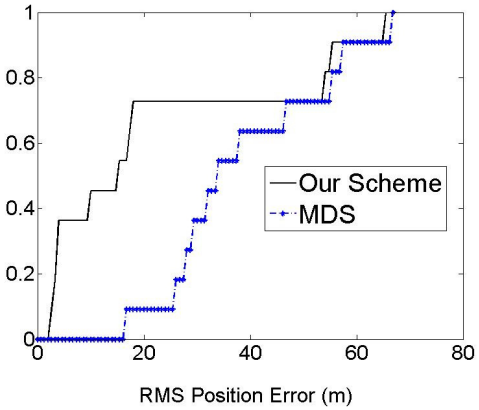


Figure 5.5: Cumulative error distribution of position estimates

Table 5.1: Simulation Parameters

Max Current speed, v_{max}	60 cm/s
Thickness of current layers	10 m
Max Depth , D	100 m
Transmission range, R	100 m
Number of nodes, N	15
Number of beacons, N_B	4
Area of deployment	300m x 300 m
MAC Back-off, $T_{backoff}$	300 s
Number of grids per node	36

Figure 5.5 shows the cumulative error distribution for position estimates using our scheme and MDS. Our proposed scheme localized 70 % of nodes with error lower than the minimum error obtained from MDS. The actual distributions of node positions are shown in Figure 5.4, with the estimate for each node indicated by circle. In Figure 5.5 we observed that around 30% of the nodes had a much larger error compared to the rest of the network when our scheme was used. This is because three of the eleven unknown nodes have multimodal distributions due to insufficient measurements. The distribution of these nodes is spread out over a large region as shown in Figure 5.4.

So far we have shown how non-concurrent measurements can be optimally combined to combat the error due to large delays in medium access. We next look at an energy-optimization strategy for the same scenario, where nodes have to be periodically localized from range estimates. Since the delay in medium access increases with the number of contenders, we will look at how to reduce the number of contenders by only allowing a subset of nodes to transmit. The key idea is that few

accurate measurements may have the same or better effect on the overall localization performance than many inaccurate ones. By selecting only a subset of nodes for transmission the overall energy-consumption is reduced. However, this subset selection has to be done strategically because every node in the network requires other nodes to self-localize.

5.4 Lifetime Maximization via Selective Transmissions

In this section we will introduce Sufficient Distance Map Estimation (SDME-D). As such, it is neither a ranging scheme nor a localization algorithm. Instead, it is a way of deciding which nodes must transmit every time concurrent self-localization has to be performed so that the overall energy consumption is reduced, without much compromise on performance.

5.4.1 Selection Scheme: SDME-D

During distance estimation, one message needs to be sent on each link. As mentioned in Chapter 2, this can be done efficiently by sending out a broadcast. However, we will show that not all links are required for localization. We therefore, propose a two step distance estimation process, SDME-D. First, a selection algorithm finds the minimum set of nodes that should broadcast to enable sufficiently accurate position estimates post-mission. Second, during the actual distance estimation, only that subset of nodes actually initiates broadcast messages. Since time-synchronization also has to be repeated periodically, we will design the selection scheme so that it can be combined with any time-synchronization protocol that is based on broadcast signaling. The overall system set up is shown in Figure 5.6 where the selection algorithm is run every T_{SYNC} and distance-estimation is performed every T_{LOC}.

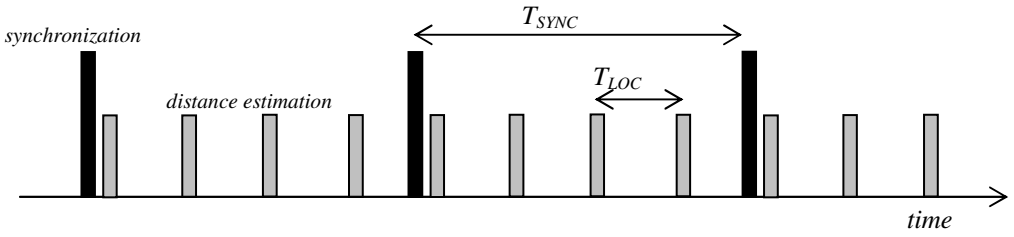


Figure 5.6: System Setup

Without applying SDME and using broadcast-based distance estimation every T_{LOC}, the average energy consumed per node per localization step is given by:

$$E_{BCAST} = \left(E_{Tx} + \lambda \cdot E_{Rx}\right) + \alpha \cdot E_{SYNC} \qquad \alpha = \frac{T_{LOC}}{T_{SYNC}} \qquad (5.4)$$

Where, E_{Tx} and E_{Rx} are the energy consumed to send and receive a ping message, respectively. E_{SYNC} is the energy consumed per node for synchronization. The parameter α captures the fact that synchronization has to be executed only occasionally. Node density λ is defined as the average number of neighbors of a node, and it is assumed that nodes overhear unicast packets due to the broadcast nature of the medium.

The goal of SDME is to reduce the energy-consumption given by equation (5.4) by selecting only as subset of nodes for transmission. We will re-examine this equation at the end of this section after applying the SDME-selection scheme.

By reducing the number of contending nodes (in the selection phase), we would not only reduce the energy consumption but also the delays in medium access in the distance estimation phase. E.g. nodes can reduce the duration of their back off when a random medium access scheme is used. This would reduce the error in

distance estimates with respect to a common time as discussed in the previous sections. Because of powerful post-mission localization techniques, it turns out that the final accuracy in positions does not degrade significantly when measurements are reduced as long as the network is uniquely localizable. Unique localizability is defined as the ability to precisely determine the position of each node if inter-node distance measurements were exact [Ere04] [Hen92] [Gol05]. It essentially means that there are sufficient constraints imposed on the network to disallow position ambiguity. In principle, centralized tests can verify these conditions for a network topology [Hen92][Hop73][Gol05]. However, since our network is mobile, these tests would have to be executed repeatedly during the mission, which would be prohibitive in terms of processing and communication overhead. Instead, we propose to use another sufficient, although not necessary, condition for unique localizability, expressed by Lemma 1, which is more amendable to a distributed implementation.

Lemma1: *A graph has a tri-lateration ordering with seeds* v_1, v_2, v_3 *if its vertices can be ordered as* $v_1, v_2, v_3 \ldots\ldots v_n$ *so that* v_1, v_2, v_3 *induce a complete sub graph and each* v_i, $i > 3$ *is adjacent to at least three vertices in the sub graph –[Gol06].For proof, see[Ere04].*

The goal of the selection algorithm is to reduce the number of nodes that broadcast while ensuring that the resultant graph has a tri-lateration ordering and is therefore, uniquely localizable. A link between two nodes is activated (i.e. added to the graph) if either of them transmits. When nodes broadcast, they automatically activate all their links. To select which nodes need to broadcast, we introduce a level assignment scheme that allows nodes to determine if it they are localizable by

examining the number of active links with neighbors of same or lower level as we will show in Lemma 2. We name this level assignment as *3-parent ordering*: An elected beacon along with two of its neighbors form a local coordinate system and are assigned level 0; any node is assigned level l if it is within the transmission range of at least 3 nodes at level less than l. By arranging nodes in ascending order of their levels, they can be tri-lateration ordered. However, when links are reduced, we can prove that a sufficient condition for tri-lateration ordering for broadcast becomes:

Lemma 2: *If a 3-parent ordering can be imposed on a network and links can only be activated by broadcasting, a tri-lateration ordering exists if every node has at least 3 active links with nodes at a lower or* ***same*** *level.*

Proof: We assume that for any $l > 0$, all nodes at a level lower than l can be tri-lateration ordered, and show by induction that an arbitrary node i, at level l with at least 3 active links to lower or same level nodes can be ordered. Let node i have M active links with nodes of level less than l and K active links with nodes of level l, with M+K>=3. For a link to be active, at least one of the nodes has to broadcast. If node i broadcasts, then it activates its entire set of links to lower level neighbors. Given that nodes at a lower level have tri-lateration ordering, i is also ordered. If node i does not broadcast, M lower level neighbors and K level-l neighbors did broadcast (with $M + K \geq 3$) since node i has active links with all these nodes. The M neighbors are ordered by induction, the K neighbors are ordered independent of i because they broadcast (as reasoned for i above). As such, i is ordered again, if it does not broadcast. □

The selection step in SDME-D incorporates Lemma 2 to determine which nodes should broadcast. Further, it requires a single broadcast per node. Thus, we can integrate it into any synchronization scheme that is based on broadcast signaling, thereby using a broadcast for both purposes. The final heuristic that runs on each node every time synchronization is done is presented in Figure 5.8. The notation used is given below.

The 'level' of a node *k: L(k).* Nodes that form a local coordinate system: *set C.* Time a broadcast from neighbor node k was received by node *i* as per i's local clock: $t_{i,k}^{rx}$ .Time a broadcast was sent by node *i* as per its local clock: t_i^{tx} . Decision for node i to remain active*:* D_i .Set of active neighbors: N_{active}. Set of lower level neighbors: N_{parent}. MAC Back off timer: $t_{backoff}$ Maximum Back off: $T_{backoff}$ .Flag indicating whether node has sent a broadcast B_{send} .

As before, an elected beacon starts the selection process by forming a local-coordinate system with two neighbors. Every T_{SYNC}, all nodes send a broadcast. Consequently, only the nodes that SDME-D designated as active, will send out a broadcast, thereby gathering distance estimates on a sufficient sub-network. However, as topology could change during T_{SYNC}, the reduced set of links may not remain sufficient to localize the network. Therefore, the value of T_{SYNC} should be determined by both the clock drift and the expected time the network topology remains more or less stable. Furthermore, each inactive node listens for broadcasts from all neighbors that it expects to be active. If one of these is missing, the node decides to broadcast anyway. This approach compensates for changes in topology, as well as for other

causes of packet loss. In the worst case, all nodes broadcast.

For any node i:

1. *Initialization:*

   D_i =ACTIVE, B_{send} = BROADCAST_PENDING

   If (*node* $i \in C$) { $L(i) = 0$, *set back off timer,* $t_{backoff} = \text{rand}(0, T_{backoff})$ }
   Else { $L(i) = -1$, $N_{active} = \phi$, $N_{parent} = \phi$ }

2. *If* (*EVENT=Back off timer expired*)
   {
   Broadcast message $m_i = [i, L(i), D_i, t_i^{tx}]$

   B_{send} = BROADCAST_SENT}

3. *If* (*EVENT=Received broadcast*)
   {
   Received message $m_k = [k, L(k), D_k, t_k^{tx}]$ *from node k.*

   Store send and receive times of message: $t_{k,i} = [t_k^{tx}, t_{i,k}^{rx}]$ // Data used post-mission

   If $(L(k) \leq L(i)$ & D_k = ACTIVE) { $N_{active} = N_{active} \cup k$ }

   If ($|N_{parent}| < 3$) { $N_{parent} = N_{parent} \cup k$.

   If ($|N_{parent}| = 3$) { $L(i) = max\ L(N_{parent}) + 1$.

   Set back off timer,
   $t_{backoff} = \text{rand}(0, T_{backoff})$ }

   If (D_k=ACTIVE) { $N_{active} = N_{active} \cup k$ } }

   If (B_{send} = BROADCAST_PENDING & $|N_{active}| \geq 3$) { D_i = INACTIVE } }

Figure 5.7: Pseudo-code for SDME-D selection algorithm

The post-processing steps on data that was collected during the mission are given in Figure 5.8.

1. *Corresponding to each T_{SYNC},*

   Compute clock offsets and/or clock skew using a time sync protocol (refer Chapter 2)

2. *Corresponding to each T_{SYNC} and T_{LOC},*

   a. *For every node that sent a broadcast, compute distance estimates with all neighbors that received it using send and receive time stamps stored on nodes and clock offsets calculated in Step 1.*

   b. *Run localization algorithm (Multi-dimensional Scaling) on distance estimates.*

   c. *Refine position estimates with Maximum Likelihood estimator.*

Figure 5.8: Post-Facto Localization using data collected during the mission

5.4.2 Performance Characterization

To evaluate the effectiveness of the SDME-D selection heuristic, we compare it to a centralized scheme that selects nodes based on Lemma 2. This centralized scheme can be found as the solution to the following mixed integer LP, which we solve using MOSEK [MOS]:

$$X_{opt} = \arg\min_{x_i} \sum_i x_i \qquad (5.5)$$

$$s.t \quad \sum_{j \in N'(k)} (x_k + x_j) \geq 3 \quad k = 1,...N, \quad x_i = 0,\ 1$$

$x_i \in \{0,1\}$ is the decision for node i to broadcast. $N'(k) = \{j : j \in N(k),\ L(j) \leq L(k)\}$.

$N(i)$is the neighbors of i, $L(i)$ is the Level of node i.

We also created an analytical model of the performance of SDME-D, derived below. It predicts the probability p_s that a node broadcasts, assuming 3-D Poisson distributed network with density λ. p_s can be computed numerically as follows:

$$p_s = e^{-\overline{\eta}(\lambda).p_s} \cdot \left(1 + \overline{\eta}(\lambda) \cdot p_s + \frac{\left(\overline{\eta}(\lambda) \cdot p_s\right)^2}{2}\right) \qquad (5.6)$$

Where,

$$\overline{\eta}(\lambda) = \lambda \cdot \left(\overline{p}_l(\lambda) + \frac{1}{16} \cdot \left(1 - \overline{p}_l(\lambda)\right)\right)$$

$$\overline{p}_l(\lambda) = \mathbf{1\text{-}} \int_{\alpha=0}^{1} e^{-\rho(\alpha,\lambda)} \cdot \left(1 + \rho(\alpha,\lambda) + \frac{\rho(\alpha,\lambda)^2}{2}\right) \cdot d\alpha,$$

$$\rho(\alpha,\lambda) = \frac{\lambda}{4} \cdot (1-\alpha)^2 \cdot (2+\alpha)$$

Proof: Let *R* be the transmission range of nodes. Since level assignment begins with a few nodes at close proximity, the propagation of levels through the network is similar to that of a wave front. An instance of the wave front is defined by a plane *B* in 3D that demarcates regions, ζ_{l-1} and ζ_l such that $\forall i \in \zeta_{l-1}, L(i) < l$, $\forall j \in \zeta_l, L(j) \geq l$. Consider a node $i \in \zeta_l$ which is at a distance of *h* from the boundary, *h*<*R*. Since the distribution of nodes is Poisson with parameter λ, the distribution of neighbors of *i* in ζ_{l-1}, denoted as $N_{l-1}(i)$, is also Poisson. The average density of $N_{l-1}(i)$ is given by the fraction of the average neighbors of *i* in the spherical cap extended by it in ζ_{l-1}, given by (5.7).

$$\rho(\alpha,\lambda) = \frac{\lambda}{4}(1-\alpha)^2 \cdot (2+\alpha) \qquad ,\alpha = h/R \qquad (5.7)$$

By construction, for any node $j \in \zeta_l$ with $\alpha > 1$, *P(L(j) > l)=1*. Therefore, the average probability for a node *i* to be at level *l* is defined for $\alpha \sim U(0,1)$ as:

$$\begin{aligned}\overline{p}_l(\lambda) &= \int_{\alpha=0}^{1} P(N_{l-1}(i) \geq 3 \mid \alpha) \cdot d\alpha \\ &= 1 - \int_{\alpha=0}^{1} e^{-\rho(\alpha,\lambda)} \left(1 + \rho(\alpha,\lambda) + \frac{\rho(\alpha,\lambda)^2}{2}\right) \cdot d\alpha \end{aligned} \tag{5.8}$$

Using a similar argument as before, for a node $i \in \zeta_l$ with $\alpha < 1$, the average number of neighbors of i in ζ_l that are no more than R units away from the boundary B, is given by:

$$\beta(\alpha,\lambda) = \frac{\lambda}{4} \cdot \left(2 - 3 \cdot \alpha^2 + 3 \cdot \alpha\right) \tag{5.9}$$

For a node $i \in \zeta_l$ with $\alpha < 1$, the number of neighbors of i which are at level l, $N_l(i)$ is Poisson with parameter, $(\beta(\alpha,\lambda) \cdot \overline{p}_l)$ as shown below.

$$\begin{aligned} P(N_l(i) = n_l \mid \alpha) &= \sum_{n=0}^{\infty} e^{-\beta(\alpha,\lambda)} \frac{\beta(\alpha,\lambda)^n}{n!} \binom{n}{n_l} \cdot \overline{p}_l^{\,n_l} \cdot (1 - \overline{p}_l)^{n-n_l} \\ &= e^{-\beta(\alpha,\lambda).\overline{p}_l} \cdot \frac{(\beta(\alpha,\lambda) \cdot \overline{p}_l)^{n_l}}{n_l!} \end{aligned} \tag{5.10}$$

Therefore, the number of neighbors of a node at the same or lower level, $N_l(i) + N_{l-1}(i)$ is Poisson with parameter, $\overline{\eta}(\lambda)$:

$$\begin{aligned}\overline{\eta}(\lambda) &= \int_{\alpha=0}^{1} \left(\beta(\alpha,\lambda) \cdot \overline{p}_l(\lambda) + \rho(\alpha,\lambda)\right) \cdot d\alpha \\ &= \lambda \cdot \left(\overline{p}_l(\lambda) + \frac{1}{16} \cdot (1 - \overline{p}_l(\lambda))\right) \end{aligned} \tag{5.11}$$

Define S_n as the number of nodes that broadcast out of n nodes, when the probability to broadcast is p_s. The probability that a node transmits as per Lemma 2 is then given by:

$$p_s = \sum_{n=0}^{\infty} P(N' = n) \cdot P(S_n \prec 3) \quad = \sum_{n=0}^{\infty} \left(e^{-\overline{\eta}(\lambda)} \cdot \frac{\overline{\eta}(\lambda)^n}{n!} \sum_{k=0}^{2} \binom{n}{k} {p_s}^k \cdot (1-p_s)^{n-k} \right)$$
$$= e^{-\overline{\eta}(\lambda) \cdot p_s} \cdot \left(1 + \overline{\eta}(\lambda) \cdot p_s + \frac{\left(\overline{\eta}(\lambda) \cdot p_s\right)^2}{2} \right) \tag{5.12}$$

The goal of SDME was to reduce the number of transmissions every time concurrent localization is performed. With the same notation as (5.4), we express the average energy consumed per node per localization as:

$$E_{SDME} = \underbrace{p_s \cdot \left(E_{Tx} + \lambda \cdot E_{Rx}\right)}_{\textit{distance estimation}} + \underbrace{\alpha \cdot E_{SYNC}}_{\textit{synchronization}} \tag{5.13}$$

$E_{SYNC} = E_{Tx} + \lambda \cdot E_{Rx}$ if SDME-S is used as the synchronization scheme, refer Chapter 3.

5.4.3 Simulation Results

To evaluate the performance of SDME, we set up simulations in Parsec, a discrete event simulator [PAR]. The models we implemented capture all elements that affect performance: propagation times, collisions, clock drift, and mobility. *N* nodes are randomly deployed over an area of observation, the size of which is chosen such that the density λ is equal to a desired value. The current-mobility is modeled as different strata with a fixed thickness, each with a randomly chosen speed and direction. To avoid excessive packet loss, nodes select a random back-off before initiating a broadcast. Due to the low data rate of acoustic modems (e.g. 80 bps for the micro-modem), the back-off window has to be relatively large. We use only 3 beacons for localization. All simulation settings are given in Tables 5.2 and 5.3. We had to set localization and synchronization periods T_{LOC} and T_{SYNC} to values different from what

we expect due to a fundamental limit on the total time that could be simulated, given the time granularity needed for offset estimation. To mimic a system that is running for several weeks, we introduced an extra clock offset of 0.1 ms. We used SDME-S as the synchronization scheme (introduced in Chapter 3).

Table 5.2: System Parameters

Data rate $\boldsymbol{R_{data}}$	80 bps
Transmit power $\boldsymbol{P_{Tx}}$	35 W
Receive power $\boldsymbol{P_{Rx}}$	0.3 W
Transmission range $\boldsymbol{R}$	≈1000 m
Clock skew $\boldsymbol{\gamma}$	0.02 ppm
Current speed $\boldsymbol{v}$	$\lvert\cdot\rvert < 0.1$ m/s
Speed of sound $\boldsymbol{c}$	≈ 1500 m/s

Table 5.3: Parsec Simulation Parameters

Maximum depth	100 m		Number of nodes N	100
Strata thickness	10 m		Number of anchors	3
Std. dev. of $\varepsilon_{intrinsic}$	2 m		Packet size	40 bits
Error in speed of sound	0.07 m/s		Random back-off	100 s
Localization period T_{LOC}	10 min		Synchronization period T_{SYNC}	30 min
Density λ	15		Error in beacon position	6m

Figure 5.9 presents a time progression of our algorithm, specifically showing the post-mission distance estimation error for all the different links at different localization times. The black cloud of crosses shows the inherent accuracy at each time, before link measurements are translated to a common time base. The triangles depict the actual accuracy after this translation, which is the metric that affects self-localization performance. The difference between the two was captured by $\varepsilon_{common\_time}$ in Chapter 3, equation (3.8). From this equation, we expect $|\varepsilon_{common\_time}| < 20$ ms, which indeed corresponds to what we observe. Figure 5.9 also includes the histograms of the distance estimation errors and the corresponding (cumulative) error distribution

in position estimates for two simulation times, t=90 min and t=110 min: the first one is during synchronization where all nodes broadcast, the other when only a subset of nodes broadcast. Since distance estimates over all links are obtained during the synchronization phase, we interpret these results as a comparison between the position accuracy achieved using all links, Figure 5.9(c) vs. that when only a subset of links are used, Figure 5.9 (d), thereby showing that localization performance does not degrade significantly when the number of active nodes are reduced. We observe that ranging information is less accurate when it is collected during synchronization. The reason is that at this time, the nodes also establish their levels as explained in Section 5.5. The broadcasts are ordered in time, which leads to larger effective values of T_{MAC}, causing reduced accuracy (see also Chapter 3 equations (3.6) to (3.8)). With a back-off window of 100 seconds, about 9% of the packets were lost due to collisions. However, these packet losses were effectively countered by our algorithm, and did not impact the overall performance. This can be seen in Figure 5.9, where the cumulative final localization error distributions for all nodes are shown. We observe that one node was not localizable in the network even when all links were used. The localization performance with the reduced set of links (in this case only 40% of the nodes transmitted) is as good as with all links, thereby showing that unique localizability is sufficient to set as a constraint. The reason is two factors that counteract each other. On one hand, estimation quality degrades by reducing the number of constraints. On the other hand, it improves when fewer nodes contend for the channel.

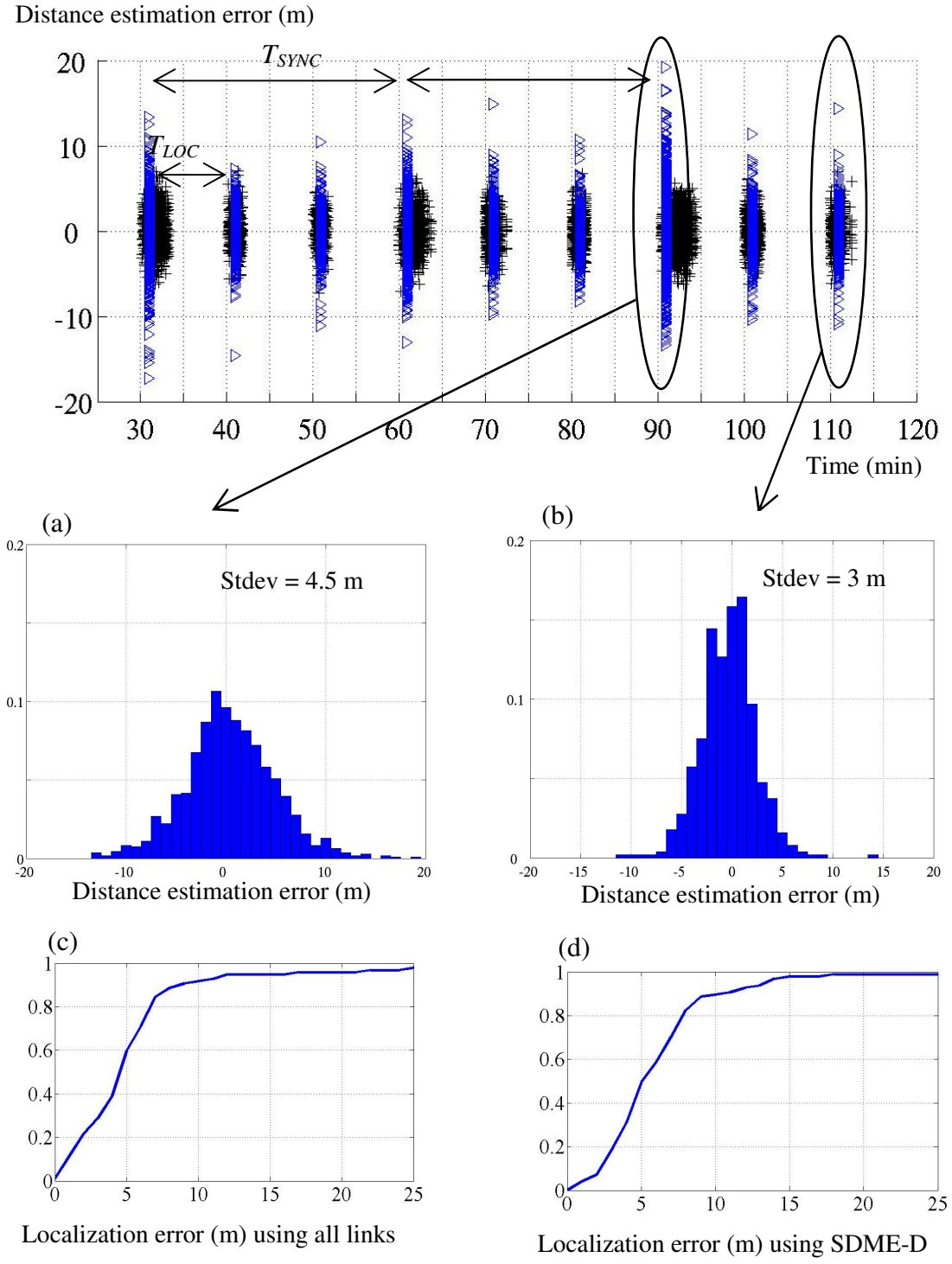


Figure 5.9: Performance of SDME-D vs. localization using all links

To get some more insight into the elements affecting accuracy, Figure 5.10 shows the distribution of total clock offset error, for the same synchronization time as Figure 5.9 (around 90 min). We also observe a good correspondence with our analysis of SDME-S protocol in Chapter 3, equation (3.6), when plugging in the values from Tables 5.2-5.3. Specifically, we find that $|\varepsilon_{motion\_flight}| \leq 40$ μs, $|\varepsilon_{motion\_MAC}| \leq 7$ ms, and $|\varepsilon_{drift}| \leq 2$ μs (these are all upper bounds). The motion of the drifters during the MAC back-off time is the dominant factor impacting error in the offset estimate.

All simulation results presented here are for the parameters we measured or believe are typical in our system. Nevertheless, it is possible that other effects also impact the actual ranging performance. For example, the presence of curved paths could further introduce errors in our distance estimates [Pre06]. However, these are large scale affects. For short range systems, we expect that node mobility and time-synchronization accuracy to be the major factors affecting distance estimation error.

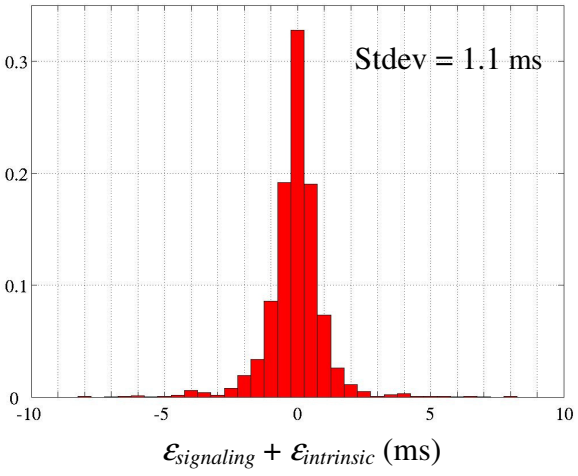


Figure 5.10: Accuracy of offset estimation with SDME-S, obtained from Parsec simulations

5.4.4 Scaling Behavior

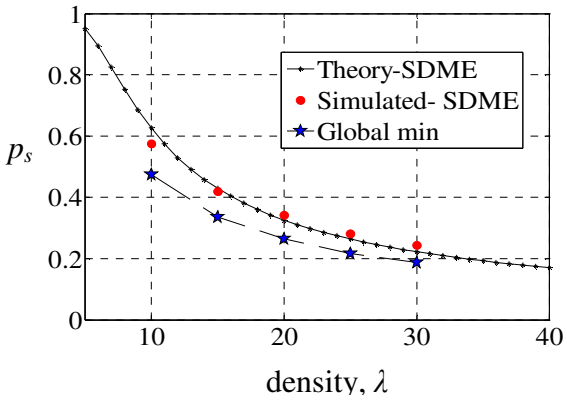


Figure 5.11: Statistical performance of SDME.

Since simulation results capture the performance of SDME-D on single realizations of the network, we now investigate the energy gains as the network density is varied. Figure 5.11 shows the probability that a node is active, p_s as a function of network density when computed analytically from equation (5.6) and then from simulations of SDME. The optimal solution from equation (5.5) is also presented. We observe that the gains using SDME can be well predicted from theory and the performance is close to the centrally-computed optimal solution. We now compute energy consumption of SDME for typical system parameters. As before, we assume T_{LOC} = 10 min. On the other hand, T_{SYNC} is now determined by both the clock drift and changes in topology. We found from simulations that with SDME-D repeated every T_{SYNC} = 1h, the node selection remains mostly valid (and even if it does not, the system automatically corrects itself locally as explained in Section 5.5). This results in $\alpha = 0.17$, refer equation (5.13). From Figure 5.11, we find the values of p_s for different densities. By plugging these into equation (5.13), we observe that the energy consumed per node for ranging can be as low as 0.7 to 0.4 times the energy of a single

broadcast for node densities between 10 and 30.

5.5 Related Work

Traditional collaborative self-localization techniques are primarily designed for static networks and do not explicitly account for displacement of nodes during the ranging epoch [Sha03] [Bis04] [Doh01] [Gol06]. Conversely, most tracking solutions for mobile robots only track devices individually with respect to anchors [Leo91] [Cor07] and rely primarily on measurements of node motion. While inter-robot distance measurements have been used to improve tracking performance, they do not operate in a setting where many nodes only have other to-be-localized nodes as neighbors with distance measurements being the only information available. There are ongoing efforts towards more dense systems, built on cheaper short-range modems [Hei06]. A survey of existing distributed schemes identified some of the challenges for underwater networks, such as degradation in performance due to non-uniform beacon distribution [Cha06]. However, in our approach, we can use centralized schemes due to post-facto processing. These centralized schemes do not exhibit this degradation [Sha03] and are in general known to have better error performance [Mao07].

A number of approaches reduce localization cost for mobile networks by adapting the rate at which distance estimates are obtained to the mobility of nodes [Til05] [You06]. They track node positions to adapt the sampling rate. Our approach is to reduce the number of measurements without using feedback on the actual position of nodes and base it on connectivity information alone. Unlike this related work, we

consider mobile networks with sparse beacons where most nodes are not within the communication range of beacons and cannot be independently localized.

5.6 Acknowledgements

This chapter is a reprint of two published articles: “Energy-Efficient Ranging for Post-Facto Self-Localization in Mobile Underwater Networks”, in IEEE Journal on Selected Areas in Communication (JSAC) Special Issue on Underwater Wireless Communications and Networks, Vol. 26, Issue 9, pp.1697-1707, December 2008; and “Motion-Aware Self-localization for Underwater Networks" in Proceedings of ACM International Workshop on Underwater Networks in conjunction with ACM MobiCom 2008.

In both cases I was the primary researcher and author and Curt Schurgers supervised the research.

APPENDIX A

A.1 Applications of Distributed Underwater Systems

The dynamics of ocean currents play a crucial role in transporting chemicals, nutrients and organisms within the ocean. As a result, many ocean processes are coupled in nature and need to be observed within this inherently mobile environment. A system of mobile entities that can be tracked at high spatial and temporal resolution over tens of kilometers has applications in addressing a number of open questions in physical oceanography, high resolution ocean modeling, developing bio-physical models, tracking pollutants, environmental monitoring and surveillance and effective design of marine protected areas. In this Section we will describe some of these applications in more detail.

A. Mapping the Sub-surface Flow-field

Mapping the sub-surface flow-field is a key application that involves resolving the trajectories of ocean-currents (also known as transport processes) at high spatial resolution (10 to 100m) over large geographic extents O(km) [Dal06]. Knowledge of transport processes and the development of high resolution ocean models would enable understanding the interaction of numerous processes with currents which in turn impacts various other applications, such as determining the connections between geographically separate ecosystems, building early warning systems that detect the spread of pollutants, understanding larval transport and the design of marine protected areas. We will describe some specific questions that are of interest to scientists, the

data they would like to obtain, existing sources of data and the utility of distributed systems in this area.

One of the challenges in physical oceanography is to better understand cross-shore transport mechanisms and describe them via mathematical models. In this context, some of the specific questions scientists are looking to answer include: distinguishing subsurface pathways along isotherms (equal temperature paths) from those on the surface, obtaining the depth dependence of eddy diffusion and computing their Lagrangian statistics such as de-correlation time and space scales. However, answering these questions requires data about the sub-surface flow field in time-scales of minutes to days and spatial scales of 10 meters to 10 kilometers, which is currently scarce.

Existing technologies for obtaining velocity maps include High-Frequency (HF) radar and Acoustic Current Doppler Profilers (ADCPs), however they have a number of limitations [Gaw07]. HF radar systems can produce maps of currents at spatial resolutions of 100m to 1000m, but a major disadvantage of these systems is that they can only provide information about surface currents [Gaw07]. HF systems have become key off-shore tools for studying surface characteristics of coastal flows. ADCPs are currently the most suitable method for resolving flows at high resolution in 3D. The high cost of the instruments and deployment costs are the main setbacks of this technique. The resolution at which measurements are obtained typically reduces with long range systems and although the micro-structure of flows can be resolved by shorter range ADCPs, this would require deploying arrays of ADCPs in two or three dimensions. An example of such an approach for smaller domains has been

demonstrated by Gaylord et. al. in studying the flows in and around a kelp forest [Gay07].

Another method for observing dispersion processes is by releasing dye along a streak at a target depth and tracking it over several days. Diffusion rates are inferred by the rate of spreading. However, data obtained from dye experiments are not always reliable and alternate methods where Lagrangian trajectories could be observed at higher spatial and temporal resolutions are required [Sun01].

Observation of the Lagrangian trajectories at higher spatial and temporal scales is possible with a system of multiple floats [Jaf06]. Tracking the sub-surface trajectory of such a system is a complementary solution to dye experiments and can overcome some of the drawbacks of HF Radar and ADCPs. By equipping floats with acoustic communication capability, the collaborative tracking techniques that we have proposed in this thesis would be well suited to tracking Lagrangian floats. To obtain true Lagrangian pathways, devices need to be small enough to experience the same effects of small-scale shear and mixing as micro-organisms. Due to the small size of devices, energy-efficient tracking is a key requirement in these systems.

B. Developing Bio-physical Models

Physical transport mechanisms, although complex, are just one of the processes that determine the connections between geographically separate eco-systems and the spatial dynamics of marine species, also known as the problem of connectivity. A mechanistic understanding of connectivity is crucial for effective design of marine protected areas [Gai07] [Jon07] [Fog07]. However, this is a challenging problem because it involves understanding the interaction of marine-

organisms with the flow-field. Bio-physical models are key tools that capture the coupled nature of processes by incorporating behavioral traits into physical models. Such models should ultimately serve to improve resource management in observation systems and in predicting the effect of climate change and human activity on ecosystems.

One of the key challenges in developing powerful bio-physical models is the ability to test model assumptions and validate predictions. In this context a number of techniques used in conjunction are valuable, one of which is the use of smart drifters (as mentioned in the earlier Section) that have the ability to adjust their buoyancy and mimic the behavior of larvae (vertical migration) [Gaw07] [Jaf06] [Jaf07]. It is known that larvae respond to a number of factors such as time of day, light, temperature, turbulence, pressure and the availability of food [Pin07]. Concurrent sampling of these relevant parameters in addition to tracking the movement of smart drifters would be very valuable in developing coupled bio-physical models and testing hypotheses about the role of behavior of larvae in connectivity [Wer07]. Therefore, the utility of such distributed sampling systems, that can measure a number of relevant parameters, goes beyond methods such as dye experiments making it especially suited for observing ocean processes that are inherently coupled.

Once again the development of comparative systems is critical to validate conclusions. For example there is a need to match a process-based understanding of connectivity with other methods of correlating migration of larvae along the coastline, such as tagging or mass marking, the use of geochemical signatures [Tho07] or genetic tools [Wer07] [Tho07] [Heg07].

C. Tracking Contaminants

One of the main challenges is to monitor and control water pollution due to the cumulative effect of runoff from a number of sources, none of which are known to directly discharge waste products into water bodies. These are commonly referred to as non-point sources. Examples include the run off of contaminants such as pesticides and toxins, nutrients, chemicals, suspended sediments and trash from agricultural operations, urban and sub-urban areas, forestry and mining operations and even from atmospheric inputs. Often *continuous monitoring* of water bodies is essential to identify and control pollution from entering larger water bodies and to build early warning systems. For example, intensive data collection is essential to building computer-models that can predict hazardous conditions such as eutrophication and hypoxia that are fatal to marine organisms.

A key requirement is to track pollutants along water pathways that ultimately flow into the oceans. This requires correlating the concentration of chemicals and nutrients such as nitrogen and phosphorous to the dynamics of the flow-field. While a system of drifters can be used to obtain Lagrangian data, other type of mobile devices such as AUVs, robots and gliders can also be used for environmental monitoring of lakes and oceans as well as for coastal surveillance.

D. Tracking Tagged Marine Animals

Another application that involves tracking mobile elements is observing the movement of tagged marine animals. This data is valuable for obtaining a census on marine species to assess and explain the diversity, distribution and abundance of

marine life. An example is the Pacific Ocean Shelf Tracking (POST) project where acoustic transmitters are implanted on a variety of species [Post]. The POST project has been successful in tracking the migration of small Salmon from the headwaters in Rockies through Pacific and Alaska. Series of receivers arranged in grid configurations across the continental shelf are used to track the movement of animals. There is a substantial effort and cost involved in laying down infrastructure for this purpose and alternate low cost methods for tracking tagged marine organisms would be valuable.

A.2 Discussion on Existing Underwater Localization Techniques

Broadly, there are three main techniques used for underwater navigation. These are dead-reckoning, acoustic navigation and environmental based navigation. In dead-reckoning methods, devices begin with a known initial position and use motion information obtained from their on-board sensors to estimate a running fix of their positions. On the other hand acoustic navigation is primarily based on acoustic time of flight measurements to external elements. Acoustic positioning methods include Long Baseline (LBL) and Ultra-short baseline (USBL). These systems can in principle estimate vehicle positions without the need for on board navigational sensors, however for high precision tracking they are used in combination with dead reckoning techniques. In LBL systems a vehicle triangulates its position from acoustically obtained distance estimates to a network of surface or sea-bed deployed transponders. In Ultra Short Baseline (USBL) systems an array of acoustic hydrophones are used to estimate distance and bearing to a vehicle from which positions are derived. The third class is environmental-based navigation. The main idea here is to use either a known

landmark map or to build this map by extracting features that can serve as landmarks. In either case, with the aid of sensors such as optical cameras and sonar, devices are tracked with respect to these landmark features.

Now, at a high level, existing work in underwater navigation shows the tradeoffs between accuracy, device cost and infrastructure costs. For example acoustic navigational methods reduce device cost by alleviating the need for precision on board navigational sensors; however they require planned deployments of position references. With advances in one way time of travel based ranging [Eus06] [Sin01] [Cur05] [Fre05a] [Fre05b] [Sin06], LBL and USBL systems can support higher update rates with infrastructure costs that scale well with the number of devices. This may in fact be a preferred positioning method when multiple devices are deployed in smaller geographic areas such as lakes. However these solutions break down for *large deployments*. Even with nominal current speeds of 10 cm/s, Lagrangian drifters can travel over a km distance in a few hours. Therefore, for applications where Lagrangian data has to be collected over time scales of days to weeks, mobile nodes would easily go out of range of even long range (10km) LBL transponders. Further the paths followed by devices are largely unpredictable. As a result, LBL based localization would require over deployment of long range transponders to ensure coverage which substantially increases the system complexity, cost and maintenance. Although, the need for infrastructure is reduced in case of Short Baseline (SBL), Ultra Short Baseline (USBL) and moving baseline systems [Cur05] systems, they still suffer from similar problems as LBL systems because they require direct communication with reference nodes.

To avoid the need for pre-deployed infrastructure (as in LBL systems) and high end navigational sensors, environmental based (also called terrain-based) navigation techniques were developed. Here scientific sensors such as optical cameras and bathymetric sonars are used along with landmark maps (magnetic, gravitational or topographic) to determine positions [Mas97] [Eus05] [Rom05] [Vaj98] [Wil03] [Wil00] [Fed98] [New98]. Environmental based navigation has also been an area of research in robotics. These methods use perceptual sensors to localize a robot with respect to landmarks in the environment. The probabilistic framework to do this was first laid out by Smith et al. in 1990 and is known as the problem of Simultaneous Localization and Mapping (SLAM) [Smi90]. The original problem in robotics relied on extracting features from the environment using sensor data and performing localization by matching measurements to features. The difficulty in extracting features underwater has resulted in modified solutions to the SLAM problem for underwater [Eus05] [Rom05] [Fle00] [Gar01] [Wil04]. The limitation of environmental based navigation techniques is that sensors for detecting features are often short range, typically O (10-100m) for bathymetric sonars and O (below 10m) for optical cameras. Therefore, terrain-based techniques and SLAM are more suited for near-bottom navigation.

We next consider the *performance* of existing techniques and identify key problems that limit performance. A number of existing techniques apply dead reckoning to precise navigational sensors such as IMUs and ADCPs and can provide positioning accuracy of around a meter, however for short periods (less than 30 minutes) [Bla03] [Kin07]. This period has been extended to a few hours with

accuracies of 0.4% of the traveled distance when a bottom lock could be obtained [Bro94] but this is only under the special case of being close to the sea floor. In general, position errors from dead reckoning methods accumulate over time even with precision sensors. For extended mission periods, devices have to either resurface or use position updates from acoustic references (LBLs) to correct for dead reckoning errors [Whi98] [Spi76].

LBL systems at 300 KHz can provide centimeter accuracy over short ranges (below 100m) when there is no uncertainty in the positions of the baseline transponders as in well calibrated sea floor deployments [Vic98] [Kin03] [Whi98]. However, when transponders are deployed on buoys there would be far more uncertainty (tens of meters) due to the motion of the buoys themselves. The accuracy of long (km) range LBLs that can go up to 10 kms [Hun74] can further vary over several orders of magnitude depending on the acoustic frequency, distance, water properties and acoustic path geometry. This is because in long range systems large scale effects come into play, such as variations in the speed of sound and the presence of refractive paths as a result of water stratification.

Since the acoustic range of transponders is frequency dependent, long range baselines have to operate at low frequencies as mentioned earlier. This has two noteworthy implications both for the transmitters on the baselines and the receivers on the vehicles- transceiver size and cost, both of which increase as the operational frequency is lowered. Our system requirements severely limit the use of large and expensive transducers pushing the operating frequencies into the ultrasonic regime and therefore limiting the communication range of devices to below a km. Additionally,

large scale coverage of the oceans with low frequency transponders is a source of noise pollution to marine organisms such as whales that communicate in these frequencies (0.5 to 40 KHz).

Finally, the energy consumed in tracking plays an important role in system lifetime because position estimates are required either periodically (or continuously) which is a recurring cost. This is further aggravated by the fact that even short to medium range modems (below 1km) have high transmit powers for acoustic communication. One of the major drawbacks of existing techniques is that point to point communication is the primary way of obtaining distance estimates which is inefficient in a network setting. While one-way time of travel has been used to improve the update rates (and positioning accuracy) for multi-vehicle localization [Eus06] [Sin01] [Cur05], we have proposed to use it for obtaining distance estimates in an energy efficient way in underwater networks.

Among the applications discussed in Section A.1, the application that demands the most stringent requirements in terms of position accuracy is obtaining high resolution velocity maps of the flow-field, where velocities have to be resolved at spatial scales of tens of meters. This remains a challenge given other system and application level constraints. Other applications such as tracking pollutants and marine animals, or environmental monitoring can tolerate much larger errors in positioning (of the order of tens to hundreds of meters). For these applications many of the existing schemes including our proposed methods which are primarily based on short range time of flight ranging can provide sufficient positioning accuracy. However, our

solutions are more scalable to larger systems and take other system requirements such as lifetime and cost into account.

REFERENCES

[Ala02] Alameda, Jr. W., "Seadevil - a totally integrated inertial navigation system (INS) solution", in Proceedings of the 2002 Underwater Intervention Symposium, New Orleans, LA, 2002.

[Aud04] Audric, M., "GAPS, a new concept for USBL", Oceans'04, Kobe, Japan. pp. 786–788, 2004.

[Aru02] Arulampalam, S., S. Maskell, N. Gordon, T. Clapp, "A tutorial on particle filters for on-line nonlinear non-Gaussian Bayesian tracking", *IEEE Transactions in Signal Processing*, 50(2), 174-188, 2002.

[Bel91] Bell, B.M., B.M. Howe, J.A. Mercer, R.C. Spindel, "Nonlinear Kalman filtering of longbaseline, short-baseline, GPS, and depth measurements", *in Proceedings of the Twenty-Fifth Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, CA, USA. pp. 131-136, 1991.

[Bet02] Bettstetter, C., "On the minimum node degree and connectivity of a wireless multihop network", Proceedings of ACM MobiHoc, Lausanne, Switzerland, June 2002, pp. 80-91.

[Bin06] Bingham, B., Seering W., "Hypothesis grids: Improving long baseline navigation for autonomous underwater vehicles", Journal of Oceanic Eng., Vol.31, No.1, 2006.

[Bis04] Biswas, P., Ye Y., "Semidefinite programming for ad hoc wireless sensor network localization", IPSN'04, Berkeley, CA, Apr. 2004.

[Bla03] Blain, M., S. Lemieux, R. Houde, "Implementation of a ROV navigation system using acoustic Doppler sensors and kalman filtering", in Proceedings of MTS/IEEE OCEANS'03, San Diego, CA, Vol. 3.,pp. 1255-1260, 2003.

[Bro97] Brokloff, N., "Dead reckoning with an ADCP and current extrapolation", *in Proceedings of IEEE/MTS Oceans'97*, Halifax, NS, Canada,Vol. 2., pp. 994-1000, 1997.

[Bro94] Brokloff, N., "Matrix algorithm for Doppler sonar navigation", in *Proceedings of IEEE/MTS* Oceans'94, Brest, France. Vol. 2. pp. 378-83, 1994.

[Cha06] Chandrasekhar, V., Seah W.K.G, Choo Y.S., Ee H.V., "Localization in Underwater Sensor Networks - Survey and Challenges", WUWNET'06*,* Los Angeles, CA, pp. 33-40, Sept 2006.

[Che06] Chen, C., R.C. Beardsley, G. Cowles, "An Unstructured Grid, Finite-Volume Coastal Ocean Model (FVCOM) System", Special Issue on Advances in Computational Oceanography, Oceanography Magazine,Vol. 19, No. 1, March 2006.

[Chir08] Chirdchoo, N., Soh W. S., and Chua K. C., "Mu-sync: a time synchronization protocol for underwater mobile networks", in *Proceedings of the third ACM international workshop on Underwater Networks WuWNeT '08*, pp. 35-42, New York, NY, USA, 2008.

[Cor07] Corke, P. , Detweiler C. , Dunbabin M. , Hamilton M., Rus D. , Vasilescu I., "Experiments with Underwater Robot Localization and Tracking", International Conference on Robotics and Automation, 2007.

[Cur05] Curcio, J., J. Leonard, J. Vaganay, A. Patrikalakis, A. Bahr, D. Battle, H. Schmidt, M. Grund, "Experiments in moving baseline navigation using autonomous surface craft", *In Proceedings of the IEEE/MTS Oceans'05Conference Exhibit*, Washington, D.C., 2005.

[Dal06] Dalrymple, R.A., S.T. Grilli, J.T. Kirby, "Tsunamis and Challenges for Accurate Modeling", Special Issue on Advances in Computational Oceanography, Oceanography Magazine,Vol. 19, No. 1, March 2006.

[Doh01] Doherty, L., Pister K. S. J., Ghaoui L. E., "Convex position estimation in wireless sensor networks", INFOCOM 2001, Anchorage, AK, April, 2001.

[Dus93] Dushaw, D., Brian D., Worcester P., Cornuelle B., Howe B., "On equations for the speed of sound in seawater", Journal of the Acoustical Society of America, 93 (1), pp: 255-275.

[Els02] Elson, J., Girod L., and Estrin D., "Fine-grained network time synchronization using reference broadcasts", in Proceedings of the Fifth Symposium on Operating Systems Design and Implementation (OSDI 2002*)*, Boston, MA, USA, pp. 147–163 December 2002.

[Ere04] Eren, T., Goldenberg O., Whiteley W., Yang Y., Morse A., Anderson B., Belhumeur P., "Rigidity, computation and randomization in network localization", INFOCOM 2004, March 2004.

[Ero07] Erol, M., Vieira L., Gerla M., "Localization with Dive'N'Rise (DNR) beacons for underwater acoustic sensor networks", WUWNET'07, 2007.

[Eus05] Eustice, R., H. Singh, J. Leonard, "Exactly sparse delayed-state filters", in Proceedings of the IEEE International Conference on Robotics and Automation, Barcelona, Spain, pp. 2428-2435, 2005.

[Eus06] Eustice, R.M., Whitcomb L.L., Singh H., Grund M., "Recent Advances in Synchronous-Clock One-Way-Travel-Time Acoustic Navigation", OCEANS 2006 , pp.1-6, Sept. 2006.

[Fed98] Feder, H., J. Leonard, C. Smith, "Adaptive sensing for terrain aided navigation", *In Proceedings of the IEEE/MTS Oceans'98*, Nice, France,Vol. 1, pp. 336-341, 1998.

[Fle00] Fleischer, S., "Bounded-error vision-based navigation of autonomous underwater vehicles", *PhD thesis, Stanford University*, 2000.

[Fog07] Fogarty, M.J., L.W. Botsford, "Population Connectivity and Conservation of Marine Biodiversity", Special Issue on Marine Population Connectivity, Oceanography Magazine, Vol. 20, No. 3, September 2007.

[Fox00] Fox, D., Burgard W., Kruppa H., Thrun S., "A Probabilistic Approach to Collaborative Multi-Robot Localization", Autonomous Robots, 8(3):325–344, 2000.

[Fox98] Fox, D. , "Markov Localization: A Probabilistic Framework for Mobile Robot Localization and Naviagation", PhD thesis, University of Bonn, Germany, December 1998.

[Fre05a] Freitag, L., M. Grund, J. Partan, K. Ball, S. Singh, P. Koski, "Multiband acoustic modem for the communications and navigation aid AUV", *In Proceedings of the IEEE/MTS Oceans'05 Conference Exhibit*, Washington, D.C., 2005.

[Fre05b] Freitag, L., M. Grund, S. Singh, J. Partan, P. Koski and K. Ball, "The WHOI micro-modem: an acoustic communications and navigation system for multiple platforms", *In Proceedings of the IEEE/MTS Oceans'05 Conference Exhibit*, Washington, D.C., 2005

[Fri06] Fringer, O.B. , J.C. McWilliams, B.L. Street, "A New Hybrid Model for Coastal Simulations", Special Issue on Advances in Computational Oceanography, Oceanography Magazine,Vol. 19, No. 1, March 2006.

[Gai07] Gaines, S.D., B. Gaylord, L.R. Gerber, A. Hastings, B. Kinlan, "Connecting Places: The Ecological Consequences of Dispersal in the Sea", Special Issue on Marine Population Connectivity, Oceanography Magazine, Vol. 20, No. 3, September 2007.

[Gan03] Ganeriwal, S., Kumar R., Srivastava M. B., "Timing-sync protocol for sensor networks", in *Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 138-149, New York, NY, USA, 2003.

[Gar01] Garcia, R., J. Batlle, X. Cufi, J. Amat, "Positioning an underwater vehicle through image mosaicking", *In Proceedings of IEEE International Conference on Robotics and Automtion*, Seoul, Vol. 3. .pp. 2779-2784, 2001.

[Gaw07] Gawarkiewicz, G., Monismith S., Largier J., "Observing Larval Transport Processes Affecting Population Connectivity: Progress and Challenges", Special Issue on Marine Population Connectivity, Oceanography Magazine, Vol. 20, No. 3, September 2007.

[Gay07] Gaylord, B., J.H. Rosman, D. Reed, J.R. Koseff, J. Fram, S. MacIntyre, K. Arkema, C. McDonald, M.A. Brzezinski, J.L. Largier, "Spatial patterns of flow and their modification within and around a giant kelp forest", *Limnology and Oceanography*, 52(5): 1,8381,852, 2007.

[Gol05] Goldenberg, D., Krishnamurthy A., Maness W. ,Yang Y., Young A., Morse A., Savvides A., "Network localization in partially localizable networks", INFOCOM 2005, pp.313-326, March 2005.

[Gol06] Goldenberg, D., Bihler P., Cao M., Fang J., Anderson B., Morse A., Yang Y., "Localization in Sparse Networks using Sweeps", ACM MOBICOM, Los Angeles, CA, Sept 2006.

[Gre03] Greunen, J. V. and Rabaey J., "Lightweight time synchronization for sensor networks", International conference on Wireless sensor networks and applications, San Diego,CA, 2003 pp. 11–19.

[Heg07] Hedgecock, D., P.H. Barber, S. Edmands, "Genetic Approaches to Measuring Connectivity", Special Issue on Marine Population

Connectivity, Oceanography Magazine,Vol. 20, No. 3, September 2007.

[Hei06] Heidemann, J., Li Y., Syed A., Wills J. and Ye W., "Research Challenges and Applications for Underwater Sensor Networking", WCNC'06, 2006

[Hen92] Hendrickson, B., "Conditions for unique graph realizations", SIAM J. Comput., vol. 21(1), pp. 65–84, 1992.

[Hop73] Hopcroft, J. E., Tarjan R.E., "Dividing a graph into triconnected components", SIAM J. Comput., vol. 3, pp. 135–158, 1973.

[Hud98] Huddle, J., "Trends in inertial systems technology for high accuracy AUV navigation", in *Proceedings of IEEE Symposium on Autonomous Underwater Vehicle Technology*, Cambridge, MA, USA. pp. 63-73, 1998.

[Hun74] Hunt, M.M., W.M. Marquet, D.A. Moller, K.R. Peal, W.K. Smith, R.C. Spindel, "An acoustic navigation system", *Technical Report WHOI-74-6. Woods Hole Oceanographic Institution*, Woods Hole, Massachusetts 02543 USA, 1974.

[Ihl04] Ihler, A., Fisher J., Moses R., Willsky A.,"Nonparametric belief propagation for self-calibration in sensor networks", IPSN'04, 2004.

[Isa09] Isard, M., MacCormick J., Achan K., "Continuously-adaptive discretization for message-passing algorithms", Neural Information Processing Systems (NIPS), 2009.

[Jaf06] Jaffe, J., Schurgers C., "Sensor networks of freely drifting autonomous underwater explorers", WUWNET'06, Los Angeles, CA, pp. 93-96, Sept 2006.

[Jaf07] Jaffe, J., R. Glatts, C. Schurgers, D. Mirza, P. Franks, P. Roberts, F. Simonet, "AUE: An Autonomous Float for Monitoring the Upper Water Column", in *Proceedings of IEEE/MTS Oceans'07*, Aberdeen, Scotland, June 2007.

[Jak05] Jakuba, M., D. Yoerger, A. Bradley, C. German, C. Langmuir, T. Shank, "Multiscale, multimodal AUV surveys for hydrothermal vent localization", *in Proceedings of the 14th Unmanned Untethered Submersible Technology Conference*, Durham, NH, 2005.

[Joh97] Johnson, M., Freitag L., and Stojanovic M., "Improved doppler tracking and correction for underwater acoustic communications", in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP-97.,* pp. 575 - 578 vol.1, 21-24 1997.

[Jon07] Jones, G.P., M. Srinivasan, G.R. Almany, "Population Connectivity and Conservation of Marine Biodiversity", Special Issue on Marine Population Connectivity, Oceanography Magazine, Vol. 20, No. 3, September 2007.

[Kap96] Kaplan, E. D., Understanding GPS: Principles and Applications, 1996.

[Kin03] Kinsey, J.C., D.A. Smallwood, L.L. Whitcomb,"A new hydrodynamics test facility for UUV dynamics and control research", *In Proceedings of IEEE/MTS Oceans'03*, San Diego, CA. pp. 356-361, 2003.

[Kin06] Kinsey, J.C., Eustice R.M., Whitcomb L.L., "A Survey of Underwater Vehicle Navigation: Recent Advances and New Challenges", *In Proceedings of the IFAC Conference of Manoeuvering and Control of Marine Craft (MCMC'06)*, September 2006.

[Ksc01] Kschischang, F.R., Frey B.J., Loeliger H.A.,"Factor graphs and the sum-product algorithm", *IEEE Trans. on Information Theory*, Vol.47, No. 2, 2001.

[Lar00] Larsen, M.,"Synthetic long baseline navigation of underwater vehicles", in *Proceedings of IEEE/MTS Oceans'2000 Conference Exhibit*, Vol. 3.pp. 2043-2050, 2000.

[Lar02] Larsen, M. B., "High performance underwater navigation- experimental results", in *Proceedings of Hydro International*, 2002.

[Leo91] Leonard, J.J., Durrant-Whyte H.F., "Mobile robot localization by tracking geometric beacons", IEEE Transactions on Robotics and Automation, Vol 7, 1991.

[Lib08] Li, B., Zhou S., Stojanovic M., Freitag L., and Willett P. , "Multicarrier communication over underwater acoustic channels with nonuniform Doppler Shifts", in *IEEE Journal of Oceanic Engineering,* 33(2):198 - 209, April 2008.

[Liu09] Liu, J., Zhou Z., Peng Z., and Cui J.-H., "Mobi-sync: Efficient time synchronization for mobile underwater sensor networks", in *Proceedings of the fourth ACM international workshop on Underwater Networks*, *WUWNeT '09,* New York, NY, USA, 2009.

[Loe04] Loeliger, H.A., "An introduction to factor graphs", Signal Processing Magazine, Vol. 21, No. 1, 2004.

[LuF09] Lu, F., Lee S., Mounzer J., and Schurgers C., "Low-cost medium-range optical underwater modem: short paper", in *Proceedings of the Fourth ACM International Workshop on Underwater Networks*,*WUWNet '09,* pp.1-4, New York, NY, USA, 2009.

[Mao07] Mao, G., Fidan B. and Anderson B. D. O., "Wireless sensor network localization techniques", Computer Networks, Vol. 51, Issue 10, pp. 2529-2553, 11 July 2007.

[Mar04] Maroti, M., Kusy B., Simon G., and Ledeczi A., "The flooding time synchronization protocol", in *Proceedings of the 2nd international conference on Embedded networked sensor systems, SenSys '04*, pp. 39-49, New York, NY, USA, 2004.

[Mas97] Massa, D.D., W. Stewart, "Terrain-relative navigation for autonomous underwater vehicles", in Proceedings of MTS/IEEE OCEANS'97, Halifax, NS, Canada, Vol. 1., pp. 541-546, 1997.

[Mas08] Mason, S. F., Berger C. R., Zhou S., and Willett P., "Detection, synchronization, and Doppler scale estimation with multicarrier waveforms in underwater acoustic communication", in *IEEE Journal on Selected Areas in Communications*, 26(9), pp.1638-1649, 2008.

[Mir08] Mirza, D., C. Schurgers, "Motion-Aware Self-Localization for Underwater Networks", in *Proceedings of Workshop on underwater networks*,WUWNET'08, San Francisco, CA, pp. 51-58, Sept. 2008.

[Mir09] Mirza, D. and C. Schurgers, "Collaborative Tracking in Mobile Underwater Networks", in Proceedings of ACM International Workshop on Underwater Networks (WUWNET) in conjunction with ACM SenSys 2009, Berkeley, California, pp. 1-8, November 2009.

[MOS] The MOSEK Optimization Software, http://www.mosek.com/

[Neg03] Negenborn, R., "Robot Localization and Kalman Filters- On finding your position in a noisy world", PhD Thesis, September 2003.

[New98] Newman, P., H. Durrant-Whyte, "Using sonar in terrain-aided underwater navigation", *In Proceedings of the IEEE International Conference on Robotics and Automation*, Leuven, Belgium, Vol. 1. pp. 440-445, 1998.

[Par09] Parrish, N., Roy S., and Arabshahi P., "Symbol by symbol Doppler rate estimation for highly mobile underwater OFDM", in *Proceedings of the Fourth ACM International Workshop on Underwater Networks WUWNet '09,* pp.1-8, New York, NY, USA, 2009.

[PAR] PARSEC, parallel simulation environment for complex systems, http://pcl.cs.ucla.edu/projects/parsec/

[Pin07] Pineda, J., Hare J.A., Sponaugle S., " Larval Transport and Dispersal in the Coastal Ocean and Consequences for Population Connectivity", Special Issue on Marine Population Connectivity, Oceanography Magazine,Vol. 20, No. 3, September 2007.

[Pom06] Pompili, D., Melodia T., Akyildiz I. F., "Deployment Analysis in Underwater Acoustic Wireless Sensor Networks", in *Proceedings of ACM International Workshop on UnderWater Networks*, WUWNet 06, Los Angeles, CA, pp 48-55, September 2006.

[Post] http://www.postcoml.org/

[Pre06] Preisig, J., "Acoustic propagation considerations for underwater acoustic communications network development", WUWNET'06, Los Angeles, CA, pp. 1-5, Sept 2006.

[Rom05] Roman, C. N., "Self Consistent Bathymetric Mapping from Robotic Vehicles in the Deep Ocean", PhD thesis, MIT - WHOI Joint Program, 2005.

[Sch08] Schmid, T., Friedman J., Charbiwala Z., Cho Y. H., Srivastava M., "XCXO: An Ultra-low Cost Ultra-high Accuracy Clock System for Wireless Sensor Networks in Harsh Remote outdoor Environments", ISSCC/DAC 2008, February 2008.

[Sha00] Sharif, B., Neasham J., Hinton O., and Adams A., "A computationally efficient Doppler compensation system for underwater acoustic communications", in *IEEE Journal of Oceanic Engineering*25(1), pp. 52 -61, Jan 2000.

[Sha03] Shang, Y., Ruml W., Zhang Y., Fromherz M., "Localization from Mere Connectivity", ACM MobiHoc, June 2003.

[She05] Sheng, X., Hu Y. H., Ramanathan P., "Distributed particle filter with GMM approximation for multiple targets localization and tracking in wireless sensor network", IPSN'05, pp. 181-188, 2005.

[Sic04] Sichitiu, M.L., Ramadurai, V, "Localization of Wireless Sensor Networks with a Mobile Beacon", in *Proceedings of IEEE International Conference on Mobile Ad-hoc and Sensor Systems*, 2004, pp. 174-183, Oct. 2004.

[Sin01] Singh, H., J. Bellingham, F. Hover, S. Lerner, B. Moran, K. von der Heydt, D. Yoerger, "Docking for an autonomous ocean sampling network", *IEEE Journal of Oceanic Engineering*, 26(4), pp. 498-514, 2001.

[Sif08] Sifferlen, J.F., H.C. Song, W.S. Hodgkiss, W.A. Kuperman, and M. Stevenson, "An iterative equalization and decoding approach for underwater acoustic communications", *IEEE J. Oceanic Eng.*, 33 (2), 182-197 (2008).

[Sin06] Singh, S., M. Grund, B. Bingham, R. Eustice, H. Singh, L. Freitag, "Underwater acoustic navigation with the WHOI micro-modem", *in Proceedings of the IEEE/MTS Oceans'06 Conference Exhibit*, Boston, MA, 2006.

[Siv04] Sivrikaya, F. and Yener B., "Time synchronization in sensor networks: A survey", *IEEE Network*, pp.45-50, 2004.

[Smi90] Smith, R., M. Self, P. Cheeseman, "Estimating uncertain spatial relationships in robotics", Autonomous Robot Vehicles (I.J. Cox and G.T. Wilfong, Eds.) Springer-Verlag, pp. 167-193, 1990.

[Spi76] Spindel, R.C., R.P. Porer, W.M. Marquet, J. L.Durham, "A highresolution pulse-Doppler underwater acoustic navigation system", *IEEE Journal of Oceanic Engineering*, 1(1), 6-13, 1976.

[Sto02] Stojanovic, M., Freitag L., Leonard J., Newman P., "A network protocol for Multiple AUV Localization", IEEE Oceans 2002, Vol. 1, pp. 604-611, 2002.

[Stu08] Stutters, L., L. Honghai, C. Tiltman, D.J. Brown, "Navigation Technologies for Autonomous Underwater Vehicles", *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, Vol.38, Issue 4. pp. 581 - 589, July 2008.

[Sun01] Sundermeyer, M.A., J.R. Ledwell, "Lateral dispersion over the continental shelf: Analysis of dye-release experiments", Journal of Geophysical Research,Vol. 106, No. C5, pp. 9603 - 9621, 2001.

[Sye06] Syed, A.A., Heidemann J., "Time Synchronization for High Latency Acoustic Networks", in *Proceedings of IEEE Infocom* 2006, pp 1-12.

[Sye08] Syed, A.A., Heidemann J., "T-Lohi: A New Class of MAC Protocols for Underwater Acoustic Sensor Networks", in *Proceedings of IEEE Infocom* 2008, pp 231-235.

[Tho07] Thorrold, S.R., D.C. Zacherl, L.A. Levin, "Population Connectivity and Larval Dispersal Using Geochemical Signatures in Calcified Structures", Special Issue on Marine Population Connectivity, Oceanography Magazine, Vol. 20, No. 3, September 2007.

[Til05] Tilak, S., Kolar V., Abu-Ghazaleh N.B., Kang K.D., "Dynamic localization control for mobile sensor networks", Performance, Computing and Communications Conference, (IPCCC) 2005, 7-9 April 2005 pp. 587 – 592

[Tri98] Trimble, G., "The Doppler inertial acoustic system for littoral navigation (DIAS)", in Proceedings of the Workshop on Autonomous Underwater Vehicles(AUV'98), Cambridge, MA. pp. 27-33, 1998.

[Vas05] Vasilescu, I., Kotay K., Rus D., Corke P., Dunbabin M., Schmidd P., "Data collection, storage and retrieval with an underwater sensor network", In Proc. of IEEE SenSys, pp.154-165, 2005.

[Vaj98] Vajda, S., A. Zorn, "Survey of existing and emerging technologies for strategic submarine navigation", *In Proceedings of the IEEE Symposim for Position Location and Navigation*, Palm Springs, CA, USA, pp. 309- 315, 1998.

[Vic98] Vickery, K., " Acoustic positioning systems. A practical overview of current systems", in *Proceedings of the Workshop on Autonomous Underwater Vehicles*(AUV'98), Cambridge, MA, USA. pp. 5-17, August 1998.

[Web10] Webster, S. E, "Decentralized single-beacon acoustic navigation: Combined communication and navigation for underwater vehicles", Ph.D. Dissertation, Johns Hopkins University, Baltimore, MD, USA, June 2010.

[Wer07] Werner, F.E., Cowen R.K., and Paris C.B., "Coupled Biological and Physical Models: Present Capabilities and Necessary Developments for Future Studies of Population Connectivity", Special Issue on Marine Population Connectivity, Oceanography Magazine, Vol. 20, No. 3, September 2007.

[Whi98] Whitcomb, L.L., D.R. Yoerger, H. Singh, D. A. Mindell, "Towards precision robotic maneuvering, survey, and manipulation in unstructured undersea environments", *In Proceedings of Robotics Research – The Eighth International Symposium (Y. Shirai and S. Hirose, Eds.)*, Chap. 2, pp. 45-54. Springer-Verlag, London. (Invited paper), 1998.

[Whi99] Whitcomb, L.L., D. R. Yoerger, H. Singh, "Advances in Doppler-based navigation of underwater robotic vehicles", in *Proceedings of the IEEE International Conference on Robotics and Automation*, Detroit, Michigan. Vol. 1. pp. 399-406, 1999.

[Wil00] Williams, S., P. Newman, G. Dissanayake, H. Durrant-Whyte, "Autonomous underwater simultaneous localization and map building", *In Proceedings of the IEEE International Conference on Robotics and Automation*, San Francisco, CA., Vol. 2., pp. 1793-1798, 2000.

[Wil03] Williams, S., "A terrain-aided tracking algorithm for marine systems", in *Proceedings of the International Conference on Field Service Robitics*, 2003.

[Wil04] Williams, S., I. Mahon, "Simultaneous localization and mapping on the Great Barrier Reef ", *In Proceedings of IEEE International Conference on Robotics and Automtion*, Vol. 2. pp. 1771-1776, 2004.

[WOO] Woods Hole Oceanographic Institution, Acoustics Communications group, "WHOI Micro-modem", http://acomms.whoi.edu/micromodem/

[Wym08] Wymeersch, H., Ferner U., and Win M. Z., "Cooperative Bayesian sel tracking for wireless networks", IEEE Commun. Lett., vol. 12, pp. 505–507, Jul 2008.

[Wym09] Wymeersch, H., Lien J., and Win M. Z., "Cooperative localization in wireless networks", Proc. IEEE, vol. 97, no. 2, pp. 427–450, Feb. 2009, invited paper.

[You06] You, C.W., Yi-Chao C., Chiang J.R., Huang P., Chu H.H., Lau S.Y., "Sensor-Enhanced Mobility Prediction for Energy-Efficient Localization", Sensor and Ad Hoc Communications and Networks (SECON '06) Volume 2, 28-28 Sept. 2006 pp. 565 – 574.

[Yun01] Yun, X., E. Bachmann, S. Arslan, K. Akyol, R. McGhee, "An inertial navigation system for small autonomous underwater vehicles", Advanced Robotics,15(5), pp. 521-532, 2001.

[Zha08] Zhang, P., Martonosi M., “LOCALE: Collaborative Localization Estimation for Sparse Mobile Sensor Networks”, IPSN’08, April 2008.

[Zho07] Zhou, Z., Cui J. H., Bagtzoglou A., “Scalable Localization with Mobility Prediction for Underwater Sensor Networks”, WUWNET’07, 2007.