UNIVERSITY OF CALIFORNIA
RIVERSIDE

Accurate and Secure Time-Based Localization With 802.11-Compatible Entities

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

Doctor of Philosophy

in

Computer Science

by

Smruti Parichha

September 2012

Dissertation Committee:

Dr. Mart Molle, Chairperson
Dr. Michalis Faloutsos
Dr. Frank Vahid

The Dissertation of Smruti Parichha is approved:

Committee Chairperson

University of California, Riverside

Acknowledgments

When I arrived in USA for graduate studies, I hardly knew how to conduct research, experiment with an idea in a scientific manner and to convey my results with clarity to an audience. My advisor Dr. Mart Molle, took me under his wings and gave me enough opportunities to explore all of these things, to grow as a researcher and to reach out to the world with my ideas. This dissertation would not have been possible without him. His wife Mrs Mary Molle, provided guidance in other matters whenever I needed it: from health to family matters, which was very comforting, especially because I was half way across the world from my own family. I will always be grateful to both of them for all that they have done for me.

My committee members Dr. Michalis Faloutsos and Dr. Frank Vahid have taken time to help and guide me whenever I approached them for anything. From Dr. Faloutsos, I not only learnt about research, but important skills like making good presentations, approaching people and working with teams, and leading a balanced life in general. He has been a mentor in many ways. From Dr. Vahid, I learnt the importance of timeliness and of thinking about all possible applications of a research idea. He has also provided constructive feedback about writing and conveying an idea in a way that can be understood not only by people in my research area, but everyone who is interested in listening. I extend my sincere thanks to both Dr. Faloutsos and Dr. Vahid for their guidance.

The networking group at the Department of Computer Science and Engineering, UC Riverside, is energetic and the group's enthusiasm is contagious. My thanks to Dr. Srikanth V. Krishnamurthy, Dr. Harsha Madhyastha and Dr Chinya. V. Ravishankar, from our group, who have helped me in one way or the other to reach my goals. My

labmates have not only been great colleagues, but also great friends and well-wishers in this journey. I would like to thank Arun Saha for his collaboration. I would like to thank Dhiman, Anirban, Marios and Ioannis for their guidance when I first joined the lab. Not only did I have fun working with my peers Nicholas, Ting-Kai, Sazzad, Huy and Masoud, but also learnt immensely from them. My lunch breaks have always been informative and filled with laughter because of Shailendra and Indrajeet. Nevertheless, our discussions were also research related many-a-times, and I did learn a lot of technical matter from them too. The interaction with the girls – Archana, Yordanos, Jianxia, Zi and Moloud will always be special to me. We were like a mini PhD girls cohort, supporting each other in times of need. Talking of cohort, I can only begin to thank Pamela Bhattacharya for being what she was to me during the most challenging times in graduate school: friend and confidante and as we progressed simultaneously towards finishing our PhDs. Her camaraderie was motivational.

If our department chair Dr. Laxmi Narayan Bhuyan would not have encouraged me to apply to graduate schools in USA, I could not have started studying for a PhD here. For his guidance through the application process and for recommending me for a fellowship, I am grateful. Because of his and his family's hospitality and inclusion, I could continue to enjoy my Oriya identity, culture and heritage in a far away land. My special thanks to our department staff, specially Terry, Amy, Madie and Jackie for helping me in every way they could, for making my time in the department enjoyable. Sometimes, they have gone out of their way to help me solve some problems and difficulties.

As I submit this dissertation, I remember with gratitude, my teachers from Carmel School, Ispat E.M. School and the National Institute of Technology Rourkela (NITR). But for their efforts, I would not be where I am today. They have built my academic foundation

towards tackling difficult research problems in graduate school. Much more than that, they have taught me invaluable life lessons about integrity, hard work and perseverance. Through all these years since kindergarten my friends Arpita, Saswati, Shibani and Supriya have always been there for me. I am fortunate that I have been able to meet with them regularly even throughout my stay here in USA. My personality and decision to go for graduate education was greatly influenced by girls whom I boarded with at NITR, especially Shubhra Rautroy and Samikhya Choudhury. I feel grateful for their influence, company and lifelong friendships. Close friends at Riverside: Shailendra, Payal, Subhas, Karthi, Shruti, Suma, Sandeep, Yusuf, Garima and Venky made me feel at home, far away from home. They have helped me immensely in dealing with the pressures of graduate school. Our get-togethers were the best breaks from work in the lab, and I always went back recharged for more work.

Lastly, I extend my thanks to my family with deep gratitude and love. They have stood by me in the toughest of times and cheered for me at every occasion of achievement. My greatest role models and supporters belong to my family. I feel extremely lucky to belong with them.

This dissertation is dedicated to

My father, Santanu Kumar Parichha, an epitome of mental strength, who taught me not only to set high goals, but also to be tenacious enough to accomplish them;

My mother, Sangeeta Hota, the core strength of our family and a wonderful educator, who taught me that the single most important thing in life is to value people;

My sister, Swasti Parichha, who has facilitated my higher education in USA and shows me how to keep a family together through thick and thin;

My husband, Dennis Jeffrey, for his love and understanding through the toughest of times. His endearing smile constantly inspires me to live life to the fullest.

And, last but not least, to everyone in my extended family.

ABSTRACT OF THE DISSERTATION

Accurate and Secure Time-Based Localization With 802.11-Compatible Entities

by

Smruti Parichha

Doctor of Philosophy, Graduate Program in Computer Science
University of California, Riverside, September 2012
Dr. Mart Molle, Chairperson

Mobile computing is growing at an incredible pace in the world around us. With the ubiquity of personal mobile devices, new application areas continue to emerge in the wireless networking domain. One emerging area that has recently been the focus of extensive research is location-based applications. For such applications, the idea of authentication includes verification of the physical location of the node, in addition to verifying its cryptographic identity. Although mobile entities are equipped with mechanisms like GPS to find their own locations, a location-based application cannot trust a node to report its true location. Due to the privileges associated with the physical location, there is incentive for a node to claim a false location. Therefore, there must be a mechanism in place to determine the location of a possibly malicious node without trusting it.

Secure localization protocols enable a group of mutually trusted nodes (called verifiers) to collectively determine the location of a possibly malicious node (called prover). In this dissertation, we consider time-based secure localization protocols. Two important criteria must be satisfied in the design of such protocols: correctness and proper timing resolution. The correctness criteria is satisfied when we ensure that the protocol is secure

against location cheating, and that the localization algorithm is executed as designed. The timing resolution criteria is satisfied when we ensure that the protocol can be implemented in the target system, and the accuracy of the computed location meets the accuracy requirement of the location-based application. The target system considered in this dissertation is an 802.11-based network and the target accuracy is on the order of a few meters.

Prior works on this topic either focus on the issue of correctness, or on the issue of proper timing resolution. None of the existing protocols addresses both the criteria simultaneously. Furthermore, none of the existing protocols have been designed for, or implemented with 802.11-compatible entities. In this dissertation, we propose a new time-based localization protocol called "Elliptical Multilateration", which simultaneously satisfies both the criteria: correctness and timing resolution. Our protocol also conforms to the 802.11 standard, and can be implemented with off-the-shelf 802.11-compatible hardware.

In the first part of the dissertation, we identify the challenges faced in designing secure time-based localization protocols for 802.11-based networks. We introduce a new protocol that addresses these challenges. Through formal analysis, we prove that our protocol addresses the correctness criterion. The second part of the dissertation focuses on the issue of proper timing resolution. We identify the factors which have so far prevented implementation of time-based localization protocols in 802.11-based networks. We explain why the 802.11 standard does not support time-based localization with accuracy on the order of a few meters. We address this issue by proposing the addition of required architectural support. Next, we quantify the effect of clock synchronization on accuracy of time-based localization. We show how to use statistical averaging to improve accuracy beyond the limits imposed by the physical layer hardware. In secure localization it is desirable to complete

the localization process fast, over minimum number of message exchanges. We propose a new algorithm which leverages the maximum likelihood method for speedy localization. Our method reduces the number of message exchanges required in hyperbolic multilateration by *at least* 50%, and often more, in comparison to the conventional method, without compromising accuracy.

Overall, we show that it is possible to design time-based secure localization protocols that can be implemented with 802.11-compatible entities, such that the positioning accuracy is on the order of a few meters.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Wireless networks have become ubiquitous. Mobile devices like cell phones, tablets, and personal digital assistants (PDAs) are commonplace in todays world. With the growth of wireless networking, many new application areas continue to emerge in this domain. One emerging area that has recently been the subject of extensive research is *location-based applications and services*. These applications and services include location-based access control, applications where privileges are extended to nodes based on their physical location, nodes taking on position-based roles, tracking of asset and personnel, location-based intelligent systems and rescue operations. It is expected that by 2014, location-based applications and services will grow to a $14 billion industry [25].

Many wireless devices are now equipped with Global Positioning System (GPS) technology. This technology can be used by a device to *self-localize*, that is, to obtain coordinates describing its own location. The device can then communicate this location to a location-based application or service that requires this information. The limitation of this approach is that it only works under the assumption that the wireless device is not malicious

and is honest in conveying its true location. However, due to the privileges associated with the physical location, in applications and services considered here, there is incentive for a device to "cheat" and claim a false location: consider expensive equipment in hospitals or industries, which have tracking enabled to prevent theft. If the system relies on the device alone to report its own location, under the control of an adversary, the device can be tampered to report a false location, and can easily be stolen. This is a major problem because GPS technology is unreliable for indoor positioning, and very easy to spoof [33]. In general, self-localization (such as that done through GPS) is not secure.

Secure location verification and localization protocols have been developed to address the security issue associated with positioning devices in a wireless network. In a *secure location verification protocol*, one node (called the *prover*) must establish its physical presence within some designated region, or at a specific location, to a set of mutually-trusted nodes (called *verifiers*). The verifiers must be able to validate or disprove the claim by the prover, even if the prover is dishonest, or the operation of the protocol is being disrupted by malicious activity. *Secure localization* is a harder version of the problem where the verifiers must determine the prover's location without an initial location claim from the prover. [1] Many secure localization protocols already exist, but it is widely believed that *time-based secure localization protocols* provide the best defense against known threats [6]. Therefore, this dissertation considers only time-based secure localization, where the verifiers estimate distances by measuring the time taken by messages to propagate between entities in the network.

In the design of secure time-based localization protocols, two important criteria

[1] The discussions and contributions in this dissertation are in the context of the *secure localization* problem, but they also apply to the problem of *secure location verification*.

need to be considered: *correctness* and *timing resolution*. In the simplest terms, "correctness" of a protocol can be defined as the ability of the protocol to meet the goals for which it was designed. For a secure localization protocol, this includes (1) ensuring that the protocol provides effective defense against the threats for which it was designed; and (2) ensuring proper sequence of message exchanges. The second criteria, "timing resolution", can be defined as ensuring the quality of a time measurement. In particular, the verifiers must be able to record timestamps for message arrivals and departures with the required precision, such that the protocol can meet the accuracy required for a target application.

Existing literature on localization does not address the issues of correctness and timing resolution simultaneously. The body of work that addresses correctness ignores issues that arise in implementation, and in achieving proper timing resolution. On the other hand, the body of work that focuses on achieving good timing resolution, does not consider correctness. Also, none of the existing work considers the use of commercial, off-the-shelf 802.11-compatible (WiFi) hardware. In this dissertation, we show that it is possible to simultaneously address both – correctness and timing resolution, while designing time-based localization protocols. We also show that such protocols can be implemented in real systems and and it is possible to achieve high accuracy in the localization result. First, we introduce a new protocol that ensures correctness. Our protocol, dubbed "Elliptical Multilateration", is secure and requires fewer message exchanges in comparison to existing protocols. Next, we prove that it is possible to *implement* Elliptical Multiltaeration (and other protocols with similar message exchange structure), with off-the-shelf hardware that conforms with the WiFi (802.11) standard. Finally we show that use of statistical techniques (on measurements made by the verifiers), not only allows the verifiers to compute the

position of the prover with high accuracy, but also to complete the localization process after making only a small number of measurements.

1.1 Motivating Examples

In the beginning of this chapter, we briefly mentioned the kinds of applications and services that require secure localization. In this section, we give specific examples of applications and/or services which require accurate (and secure, depending on the application) localization with 802.11-compatible hardware.

In systems where *location-based access control is enforced*, an entity in the network is allowed access to the network, if and only if, it is present within a defined region. Consider these scenarios: (1) Businesses that offer free WiFi to their customers do so to encourage loyalty and customer satisfaction. These businesses pay considerable amounts to internet service providers (ISPs) for extending such services. However, currently there is no way for them to stop someone who is not a customer, from using up bandwidth for free. This is because having a system in place for assigning passwords to every customer, turns out to be inconvenient and expensive. Wastage of bandwidth due to illegitimate access not only adds costs for the business establishment, but also causes paying customers to be dissatisfied due to low network throughput. In this scenario, it is highly desirable to limit network access to entities based on their physical presence within a building, store etc. (2) Consider attendees at a conference or meeting, who want to exchange information between their personal wireless devices without allowing others outside the conference hall or meeting room to intercept it. If the attendee population changes frequently and is large is size, it is difficult to establish secure channels of communication for each pair of devices. In a

ball park, game statistics and event information should be accessible to personal devices of only those who paid for the tickets to enter the park. This is yet another example where distributing and managing large number of cryptographic keys is undesirable, and allowing access depending on physical location solves the problem.

Some applications like Foursquare, Shopkick, Facebook Places, and Gowalla are used to *extend privileges and/or rewards to people who visit an establishment frequently.* However, these applications use GPS location which can be easily spoofed [33]. Often users falsely "checkin" at these establishments even through they are at a remote location, and successfully receive cash rewards or gifts. Better security and fairness in rewarding loyal customers can be achieved by verifying the physical location the users device with trusted access points at the establishment in question [56].

Sometimes, an entity in a network may be required to take on a *position-based role.* For example, in sensor networks, a sensor is required to tag the collected (environmental) data with its physical location. When the collected data is processed, data with missing or incorrect location information is not useful. If a geographic routing protocol is used to compute routes, a node may be the designated "nth hop" depending on its location. Asset and personnel tracking [1, 52] are other application areas, where determining the location of entities securely and accurately is a necessity.

Besides commercial applications, the ability to locate devices is of critical importance in *search and rescue* work. For example, while searching for victims buried under debris after an earthquake, speed is critical for preventing permanent injury or loss of life. A victim's call for help may not be audible through debris, or the victim could be unconscious. Fortunately, many people often carry a WiFi-capable personal device with them. As

long as this device is powered on, rescue workers can use the signal to help locate the victim quickly [43]. Although attenuation of wireless signals in the presence of metallic or concrete obstructions could be a problem, it has been shown [49] that in commonly encountered disaster situations, the signals from people's personal devices are detectable across debris.

These examples show that enabling time-based (and secure, when necessary) localization in 802.11-based wireless networks is very important. They also show that implementing localization protocols with off-the-shelf 802.11-compatible hardware, in a secure manner, is an interesting problem that needs a good solution. This motivates the work presented in this dissertation.

1.2 Challenges

In the previous section, we described various applications where there is a need for accurate and/or secure localization protocols. These protocols must simultaneously address issues related to both (1) correctness and (2) timing resolution. Since 802.11 is the most widely used wireless networking standard, it is also desirable that these protocols be designed according to 802.11 specifications, and be implemented with 802.11-compatible hardware. Despite the growing demand for such protocols, none of the currently known localization protocols meets all the requirements mentioned above. The problem that we address in this dissertation is how to design and implement time-based protocols for 802.11-based networks that are both secure and accurate.

In this section, we throw light on the challenges that make it a hard problem to solve. For a complete understanding of the challenges, we must first understand the problem requirements in greater detail. Therefore, we first explain what it means to fulfill

the "correctness" and "timing resolution" criteria in the context of our problem.

1.2.1 The Correctness Criteria

A protocol is said to be "correct" if it can be proven through formal verification that it meets the goals for which it was designed [3]. To meet the "correctness" criteria, a secure localization protocol that solves our problem, must (1) be secure against the distance fraud attack [2] and (2) execute message exchanges in a manner that guarantees that events at participating entities, occur in a prescribed sequence, and result in localizing the prover. From existing literature [8, 5] we know that it is possible to formally prove if the protocol satisfies (1). The work by Bella [3] shows how it is also possible to write a correctness proof for (2). The standard method for writing such a correctness proof uses a finite state machine representation of the protocol's message exchange structure. Each state in this finite state machine represents whether or not a specific event has already occurred. The proof validates the sequence of events to be appropriate for the goal of localization. If we can validate both (1) and (2) by formal analysis for a localization protocol, it is determined to be "correct" in the context of our problem.

1.2.2 Ensuring Proper Timing Resolution

Besides proving correctness, it is also important to prove that the results obtained from a "correct" protocol meet the *accuracy requirement* of the target application. The finite state machine representation of a protocol, as used for proving correctness, does not include information about the quality of time measurements. By quality of time measurements, we

[2] A big security threat for time-based localization is distance fraud. This attack will be defined and discussed further in chapter 2.

refer to the precision of timestamps recorded for message arrival and departure events, and the timing error introduced during protocol execution. Ensuring proper "timing resolution" in a time-based localization protocol means ensuring that (1) the timestamps for individual message arrival and departure events can be captured with the required precision, and (2) the cumulative timing error from the beginning of the protocol until the current state in the state machine representation, is below a predefined limit [24].

1.2.3 Challenges faced in Design and Implementation

Having defined the "correctness" and "timing resolution" criteria, we can now highlight the challenges that researchers face in addressing both these requirements.

First, let us consider the challenges that arise in ensuring correctness. Many secure localization protocols that defend against distance fraud have been proposed in literature. While researchers have paid attention to the cryptographic issues in order to ensure security, issues related to implementation of these secure localization protocols have been largely ignored. For example, some protocols [4, 8] assume that the messages exchanged between participants may be in the form of single bits. However, zero length messages are neither accommodated in the 802.11 standard, not can be implemented on commercial off-the-shelf hardware. One of the known [8] protocols requires the prover to possess multiple radios, since it must receive multiple messages simultaneously over different channels. Others [50] assume that a participating node can perform some operation (XOR) on a stream of incoming bits and start transmitting the modified stream, even as it is still receiving the incoming stream. This would require the capability to receive and transmit simultaneously. None of these assumptions hold for commercially available 802.11-compatible networking cards.

Therefore, a major challenge in ensuring correctness is using the concepts from pervious work to ensure security, yet, modifying the message structure and format to conform with the 802.11 standard.

Next, we consider the challenges faced in ensuring proper timing resolution when localization protocols are implemented in 802.11-based networks. In time-based localization, the verifiers must measure the time taken by messages to propagate between participating entities. They do so by recording timestamps for message arrival and departure events. A measurement for the propagation time of a message is then converted to the equivalent distance by multiplying the signal velocity. Since this is how distances are estimated in time-based localization protocols, the accuracy of the timestamps captured for arrival and departure of messages directly effects the positioning accuracy. In 802.11-based networks, radio frequency (RF) is the de facto medium. Because of its high speed ($3 \times 10^8 m/s$), it is extremely difficult to make accurate distance measurements over RF. An error of just $3.3ns$ in timing an event converts to an error of $10m$ in the distance estimate! Therefore to locate the prover with a allowable error of a few meters, the verifiers must capture timestamps with nanosecond-level precision. The 802.11 standard currently supports timestamping through the TSF function, which allows for microsecond level precision at best. Therefore, the major challenge in ensuring proper timing resolution, is to find a way to enable 802.11-compatible entities to timestamp events with nanosecond-level precision.

Ensuring correctness and ensuring proper timing resolution, are individually difficult enough to do. To solve our problem however, we must overcome a much greater challenge– addressing both these issues simultaneously to design and implement a secure time-based localization protocol that can locate the prover with high accuracy. We must

also ensure that our solution conforms with the 802.11 standard specifications and hardware. To the best of our knowledge, a solution that satisfies all these criteria has not been attempted before.

1.3 Outline and Contributions

In this dissertation we show that it is possible to support secure time-based localization in 802.11-based networks and obtain highly accurate localization results. We also show that our solution meets all the requirements mentioned in the beginning of section 1.2. To find a solution to our problem, we (1) design a new time-based secure localization protocol, (2) propose easy architectural additions to commercial-off-the-shelf (COTS) 802.11 hardware, (3) show how to reduce measurement error, and (4) apply statistical techniques to increase the accuracy and/or efficiency of localization. In this section we provide an outline of the dissertation. As we walk the reader through the outline, we will also highlight our contributions.

In chapter 2, we survey the existing literature that (a) addresses correctness and (b) addresses timing resolution. We find that literature related to (a) is disjoint from literature related to (b). We discuss the cause of this disjoint, and stress that a solution for our problem must *simultaneously* address correctness as well as proper timing resolution.

In chapter 3, we introduce a new protocol called *"Elliptical Multilateration (EM)"*. Our protocol ensures both – correctness and proper timing resolution. The focus of this chapter is to elaborate on how the "correctness" criteria is met. Discussions on how EM (and other protocols with similar message exchange structure) ensure proper timing resolution are presented in later chapters of the dissertation. Our EM protocol preserves the security

properties of Verifiable Multilateration [6] – a protocol that has been studied extensively in the past, and believed to be extremely secure against distance fraud attacks. In addition, it employs the efficient message exchange structure of Hyperbolic Multilateration [67]– a localization technique that is extensively in use in real-world systems. Overall, EM is a secure time-based localization protocol that conforms to the 802.11 specifications, and can be implemented on 802.11-compatible hardware. Following are our major contributions in chapter 3:

- We designed a new secure time-based localization protocol that ensures both correctness and proper timing resolution. Unlike many existing protocols, the message format in our protocol conforms with 802.11 standard specifications.
- Through analysis, we prove that in comparison to existing secure time-based localization protocols, (1) EM can complete localization in fewer number of messages exchanges, and (2) EM localizes the prover with better accuracy.
- We also prove that EM defends against distance fraud, even when the prover is sophisticated, and possess multiple radios and directional antennas.

The analysis (that we present in chapter 3) to show that EM is secure against distance fraud and that its message exchange structure can efficiently localize a prover, proves that all aspects of "correctness" can be ensured while designing a time-based localization protocol.

In the remainder of the dissertation, we focus on the issue of timing resolution. In chapter 4, we discuss why it is extremely difficult for the verifiers with 802.11-compatible hardware, to capture timestamps for message arrival and departure events with nanosecond-

level precision. We introduce the reader to related work done in the domain of precision clock synchronization. By studying work from this other domain, we identify the reason why time-based localization protocols implemented over RF, currently do not meet the required timing resolution. We propose a solution that allows the verifiers to capture timestamps with nanosecond-level precision. This makes it possible to *implement* secure time-based localization in 802.11-based real systems, while satisfying the requirement for proper timing resolution. Our contributions in this chapter can be listed as follows:

- We identify the dominant factor that prevents time-based localization protocols from meeting the timing resolution requirement in 802.11-based networks.
- We propose easy architectural modifications to 802.11-compatible off-the-shelf hardware that allows the verifiers to time message exchanges with the nanosecond-level precision over RF.
- Until now, it was not possible to implement a time-based localization protocol in a manner that conforms with the 802.11 standard. Ours solution now makes practical implementation possible.

With the proposed architectural changes to commercial-off-the-shelf 802.11 hardware in place, it is possible for a time-based localization protocol to meet the proper timing requirement criteria. By proving the we can ensure correctness, and that practical implementation with proper timing resolution is possible, we showed that the problem addressed in this dissertation does have a valid solution.

Next, we were interested in investigating how we can enhance the accuracy of localization. In chapter 5, we study the effect of factors (other than the dominant factor

identified in chapter 4) like presence of synchronization between the verifier clocks, quantization of the measurements, and the signal processing delays in the transceiver hardware. In the absence of error-correcting techniques, the accuracy of localization is upper bounded by the resolution of the verifiers' clocks. However, using a statistical error-correcting technique allows us to push the upper bound on achievable accuracy. We show that simple averaging, over a large number of measurements, can help us to measure message propagation times, with a granularity greater than that allowed by the verifier clocks. Following are our contributions in chapter 5:

- We present an anatomy of the error introduced into the measurements made by the verifiers. In doing so, we identify all the factors that degrade the accuracy of the localization.

- We propose techniques to compensate for delays in the transceiver's signal processing stages and for error introduced due to multipath.

- We compare the error in localization when (1) time-of-arrival Elliptical Multilateration and (2) time-difference-of-arrival Hyperbolic Multilateration protocols are executed. For each case we also quantify the error when the protocol is executed with, and without synchronization amongst participating verifiers. We find that in both cases, synchronizing verifier clocks prior to protocol execution, increases localization accuracy.

- We show that simple averaging over multiple observations allows the verifiers to measure message propagation times with sub-clock precision. This allows for localization accuracy greater than that allowed by the resolution of the verifier clocks.

Thorbjornsen et al. [63] had first proposed the technique of simple averaging to obtain-sub clock timing accuracy over RF. However, their work addresses the issue of timing resolution in the context of sensor networks. Moreover, their work does not consider the message exchange structure used in secure localization protocols, for example, Elliptical or Hyperbolic Multilateration. Although our work uses simple averaging similar to theirs, we are the first to apply this technique to localization protocols meant for 802.11-based networks. Our work is also the first to include a study of the convergence characteristics, and quantification, of measurement error in time-based localization, with and without clock synchronization.

Although simple averaging allows us to obtain sub-clock accuracy in measuring the propagation times of messages, it requires at least 100, and often, a greater number of message exchanges to apply this technique. Also, we observed the the error convergence to be non-linear: the error magnitude rapidly falls to a steady-state within the first hundred samples. After this, number of samples must be increased by an order of magnitude to observe further enhancement in accuracy.

In secure localization however, the fewer the message exchanges to complete localization, the better it is. This is because of reasons like non-cooperation on the part of the prover over a large number of message exchanges, undesirable traffic overhead in the network, etc. Speedy, yet accurate localization is the motivation behind the work presented in the next chapter 6. While performing analysis on the measurement data in chapter 5, we noticed that the data in the case of Hyperbolic Multilateration, exhibits a unique geometrical property. This allowed us to use the data in a clever way to compute message propagation times with sub-clock accuracy comparable to the accuracy obtained by apply-

ing simple averaging. Only this time, the number of measurements required were far lesser than the number required in simple averaging. Our findings and contributions in the final chapter are summarized as follows:

- We formulated the problem of measuring the propagation time (equivalently distances) as a maximum likelihood estimation problem.
- We analytically formulated the mathematical model that defines the unique geometric property of the measurement data observed in Hyperbolic Multilateration.
- We found a way to bound the solution to a narrow search space, which allowed us to solve the estimation problem with only $15 - 40$ measurements.
- We verified our technique with extensive simulations.
- We also validated our findings by using real experimental data from Cooklev et al's [9] work.

We found that both – the simple averaging technique, and the maximum likelihood estimation technique allow for a localization accuracy in the order of 10 meters in 802.11-based networks. However the latter technique, where applicable, is much faster and efficient because it uses requires fewer measurements.

1.4 Summary

Through our work in this dissertation, we show that *(1) It is possible to design time-based localization protocols that are both secure and accurate. (2) Such protocols can*

also be successfully implemented in real-world 802.11-based networks with off-the-shelf hardware. (3) If the message exchange structure and architectural modifications proposed in this dissertation are employed, it is possible to locate the prover with an accuracy in the order of 10 *meters.*

Chapter 2

Background and Related Work

In chapter 1, we motivated the need for time-based secure localization protocols that can be seamlessly integrated into environments where the entities and infrastructure conform to the current WiFi (802.11) standard. We also introduced the two essential criteria that must be satisfied for designing such protocols: (1) correctness, and (2) proper timing resolution. In this chapter, we survey the existing work related to both (1) and (2).

Considerable work has been done in secure and efficient time-based localization, as well as implementation of (non-secure) time-based localization in real systems. Existing literature concerning correctness comes from researchers in computer security and cryptography, who have focused their efforts on the the pattern of message exchanges and/or cryptographic issues related to their contents. Protocols proposed by these researchers ignore practical implementation. Literature that addresses timing resolution comes from researchers who study clock synchronization and/or positioning in wireless networks. Although the systems that they have proposed are meant to implement time-based localization over RF, they do not address security in time-based localization. As a result, literature that

addresses the correctness of time-based localization protocols is essentially disjoint from literature that addresses implementation and timing resolution of these protocols. None of the existing literature provides a solution for the problem addressed in this dissertation – how to design a time-based localization protocol that addresses correctness as well as timing resolution issues, and can be implemented in 802.11-based environments?

To find a solution to our problem, we started by studying existing work from both bodies of literature. However, we found that ideas from previous works are incomplete in one way or the other towards solving our problem. A major challenge was to identify the best findings from prior work covered in this chapter, and to leverage those ideas to develop a solution geared towards 802.11-based networks. Using the existing literature as a starting point, we proposed a new protocol, added features, proposed architectural changes, and came up with new algorithms to solve our problem.

In the first part of this chapter, we focus on work done on correctness of time-based localization protocols. We discuss known secure time-based localization protocols that address the threat of distance fraud, and execute message exchanges in a sequence designed to complete localization. Therefore all the protocols discussed in this section meet the "correctness" criteria. In the second part of this chapter, we focus on work related to practical implementation and achieving "proper timing resolution". We survey the existing work related to achieving proper timing resolution in a *realistic* implementation.

2.1 Correctness of Time-Based Secure Localization Protocols

2.1.1 System Model and Notation

Existing secure time-based localization protocols have been designed under the assumption of a common system model. However, different works use different notation for representing entities, events, distances and time intervals. For ease of understanding, we introduce a our own notation to describe the protocols from existing literature. Another advantage of coming up with a common notation is that it allows us to keep the notation consistent throughout the dissertation. In subsequent chapters, it also helped us to compare the different protocols by using mathematical analysis. The glossary at the end of the dissertation serves as a reference to our notation.

Under the usual assumptions for time-based secure localization protocols in the literature, the system model consists of a wireless network with mobile nodes, which are free to join and leave the network dynamically over time. However, during one execution of the protocol to verify a particular location claim, we assume that all participating nodes are at rest with respect to each other.

The network consists of two kinds of nodes. Verifiers $\{u, v, w, z, ...\} \in V$ are mutually trusted nodes who can securely exchange information amongst themselves. Handling defective and/or malicious verifiers is beyond the scope of this dissertation. We assume that all verifiers know each other's exact locations relative to some physical coordinate system. Moreover, all verifiers can timestamp message arrival/departure events with high precision, and may possess mutually synchronized clocks. The verifiers might establish shared keys

with the prover for exchange of information not pertaining to location-based applications. The cryptographic aspects of secure time-based localization, however, will not be discussed in this dissertation.

Prover p is an untrusted node, whose physical location is unknown to the verifiers. The goal of the verifiers is to determine p's location, by measuring its response time to a set of skill testing questions. We assume that in general, the prover possesses a single radio with an omnidirectional antenna, therefore it is *"resource-constrained"* in terms of hardware. A sophisticated prover with multiple radios and directional antennas, will be termed as a *"resourceful prover"*, and denoted as p^*. Even for p^*, all of the radio equipment is at the same physical location. Otherwise, we will classify the threat as a group of colluding provers. The minimum response time of the prover $\hat{\Delta}$ is known to the verifiers.

Let x, y, etc be the locations of each node relative to our physical coordinate system, and $D(x, y)$ represent the Euclidean distance function. We assume that nodes communicate by transmitting messages through an isotropic broadcast medium with no obstacles, so a message from x to y follows the line-of-sight path of length $D(x, y) \equiv D(y, x)$. In addition, the environment is anechoic so there are no multipath effects and each node receives a single copy of each message. Without loss of generality, we normalize time and space so that signals propagate at unit speed, i.e., one unit of distance per unit time. Thus, for any two points x and y, we can use $D(x, y)$ to represent both the distance and propagation delay between them.

Although various authors make different assumptions about the message structure – ranging from "a single bit" to "a data stream of arbitrary length", there is a specific reference point within each message that is used for timing message arrival and departure

events. We use the notation e_x^y to represent the discrete event that the reference point from a message sent by node x is now at the location of node y, and C_x^y as y's timestamp for that event.

2.1.2 Timed Challenge-Response Message Exchanges

A single round of any secure time-based localization protocol consists of a challenge-response message exchange. A challenge-response message exchange can either be a challenge-response echo as shown in Fig. 2.1(a), or a challenge-response relay as shown in Fig.2.1(b).

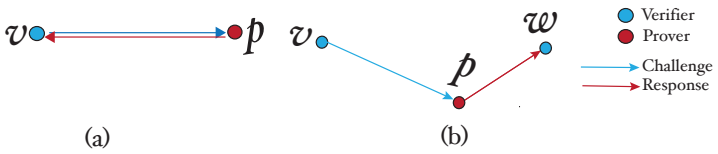


Figure 2.1: (a) a bidirectional challenge-response echo executed between verifier v and prover p (b) a challenge-response relay consisting of a challenge sent from verifier v to prover p, followed by a response sent from p to verifier w.

A two-way challenge-response dialogue between a verifier v and the prover p will be denoted as a "challenge-response echo". Execution of a challenge-response echo consists of the following discrete events: (i) the verifier sends a challenge; (ii) the prover receives the challenge; (iii) the prover sends the response after computing it; and (iv) the verifier receives the response. In Fig.2.2, the challenge-response message pair consisting of the solid blue arrow $v \rightarrow p$, and the solid red arrow $p \rightarrow v$ represents a challenge-response echo. The running time of a challenge-response echo is the interval between the instant when v sends the challenge and the instant when it receives the response. If Δ_v is the time taken by the prover to compute the response and send it, then the running time $T_p(v, v)$, for a

challenge-response echo along the path $v \rightarrow p \rightarrow v$ is

$$\begin{aligned} T_p(v,v) &= D(v,p) + \Delta_v + D(p,v) \\ &= 2 \cdot D(v,p) + \Delta_v \end{aligned} \tag{2.1}$$

A challenge-response relay between a verifier v and verifier w via prover p, consists of the following discrete events: (i) the verifier sends a challenge; (ii) the prover receives the challenge; (iii) the prover sends the response after computing it; and (iv) a verifier other than the one that sent the challenge, receives the response. In Fig.2.2, the challenge-response message pair consisting of the solid blue arrow $v \rightarrow p$, and the solid red arrow $p \rightarrow w$ represents a challenge-response relay.

The running time of a challenge-response relay $T_p(v,w)$, is the interval between the instant when v sends the challenge, and the instant when w receives the response. Therefore, the running time of a challenge-reponse relay along the path $v \rightarrow p \rightarrow w$ is

$$T_p(v,w) = D(v,p) + \Delta_{\{v,w\}} + D(p,w) \tag{2.2}$$

where $\Delta_{\{v,w\}}$ is the time taken by prover p to respond to the challenge.

Due to the broadcast nature of the wireless medium, when a single verifier, say v, executes a challenge-response echo with prover p, multiple passive verifiers can simultaneously observe separate challenge-response relays. We will use the term *"witness"*, coined by the authors in [54], to refer to a verifier who does not send a challenge, but passively observes message exchanges between other participants. It silently observes the challenges sent by other verifiers, and responses sent by the prover. For example, verifier w is a witness

in the challenge-response message exchange shown in Fig.2.2.

2.1.3 Classification

Brands and Chaum first introduced secure time-based ranging, where the physical distance of an entity is verified in addition to its cryptographic identity. Using their distance bounding protocol [4], a verifier can upper bound its distance to the prover p. The verifier achieves this by measuring the running times of multiple challenge-response echoes executed in rapid succession. The verifier also completes a cryptographic authentication process over these challenge-response echoes to verify the identity of the prover. Although we do not discuss cryptographic aspects of the secure time-based localization protocols in this dissertation, we will always assume that this form of authentication is a part of a secure localization protocol.

Building on the basic distance bounding protocol, Sastry et.al [58] then formalized the problem of *in-region* verification. Their *keyed-echo protocol* takes distance bounding a step further by using the distance bounds generated by multiple verifiers to constrain the prover to a small region. However, the authors did identify a security weakness in their protocol. Since each verifier executes its own challenge-response echo with the prover individually, such that the prover responds to only a single verifier at a time, a malicious prover can change its response time Δ, by different amounts for different verifiers. By doing so, it can convince one or more verifiers that its physical distance from it (them) is different from its true distance. When the verifiers subsequently combine their individual measurements, some or all of which are incorrect, they compute an incorrect location for the prover. Therefore, when the prover is required to respond to only a single verifier at

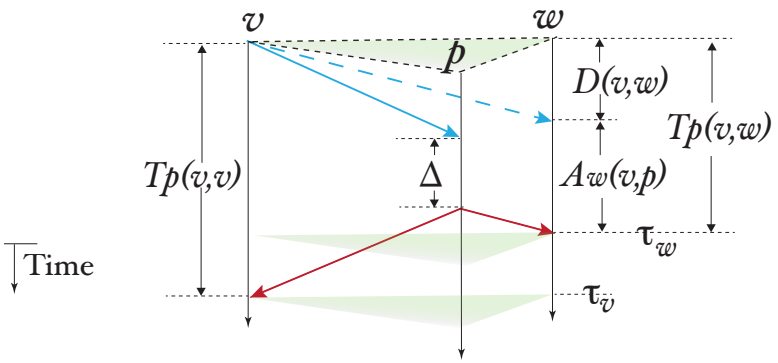


Figure 2.2: Space-time diagrams for a challenge-response echo executed between verifier v and prover p, and a challenge-response relay consisting of a challenge sent from verifier v to prover p, followed by a response sent from p to passive verifier w. Δ will be denoted as Δ_v for the challenge-response echo along path $v \rightarrow p \rightarrow v$, and as $\Delta_{\{v,w\}}$ for the challenge-response relay along path $v \rightarrow p \rightarrow w$.

a time, it can trick the verifiers into accepting a false location. This is called the *distance fraud* attack [11] – one of the greatest security threats for time-based localization.

Following the work by Sastry et al., many time-based localization protocols have been proposed, which are designed to thwart the distance fraud attack. All time-based localization protocols use inputs from multiple verifiers, and perform *multilateration* [1] to determine the location of the prover. We classify these secure time-based localization protocols based on differences in their execution and message exchange structure. We chose this criteria as the basis of classification because the message exchange structure directly effects two properties of a time-based localization protocol – (i) defense mechanism against distance fraud, and (ii) accuracy of localization. Next, we describe our classification of existing protocols. We highlight the differences in message exchange structure, and the

[1]The origin of the term "multilateration" is from the two component terms: "multi", meaning many, and "lateration", which refers to range-based distance measurements made by an entity. Therefore, multilateration refers to positioning by combining range-based distance measurements from multiple entities.

features that provide defense against distance fraud, across the different classes of secure time-based localization protocols.

A. Single Input Single Output (SISO) Time-based Localization

In each round of a SISO localization protocol, the prover p responds to a single challenge (input) generated by a single verifier v. This verifier is also the sole receiver that monitors p's response (output), therefore we term it as *Single Input Single Output (SISO)*. In each round a different verifier measures the running time of a challenge-response echo that it initiates. Fig. 2.3(a) shows the space-time diagram for this. Using its measurement, each verifier can constrain the prover to a circular region. In Fig. 2.3(b), we show how *multilateration* uses a sequence of challenge-response rounds conducted by different verifiers to restrict prover p's location to the intersection of their respective circular constraints. Since individual verifiers take turns to execute a challenge-response message exchange with the prover in SISO protocols, this class of protocols is susceptible to distance fraud.

An example of a SISO time-based localization protocol that also addresses distance fraud, is Capkun and Hubaux's [6] *Verifiable Multilateration (VM)*. VM is a time-of-Arrival (ToA) [40] protocol that allows multiple verifiers to locate the prover at the mutual intersection of circular constraints generated by each. To prevent distance fraud, VM adds a the *point-in-triangle* test. Let p denote both the prover and its true location, and let $\hat{p}$ denote the prover's location claim, possibly untrue. If $\hat{p}$ is within the triangle formed by the locations of the three verifiers (more generally, the convex hull generated by the N participating verifiers), then $\hat{p}$ is said to satisfy the point-in-triangle test with these verifiers. Capkun and Hubaux proved that when $\hat{p}$ satisfies the point-in-triangle requirement, a single

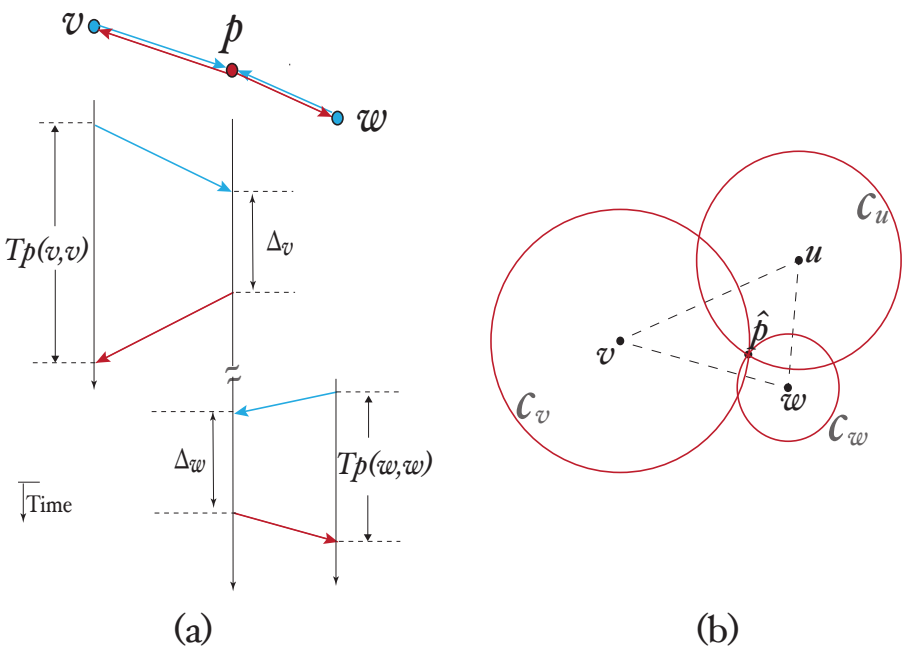


Figure 2.3: (a) Space-time diagram for timed challenge-response echoes between the verifiers and the prover (b) Circular constraints generated by three verifiers in SISO localization.

dishonest prover's response must reach at least one of the verifiers late, unless it can lower the time taken to generate the response to v's challenge below $\hat{\Delta}$. This is impossible because $\hat{\Delta}$ is, by definition, the minimum possible response time. (This is discussed in greater detail in chapter 3). Let the expected running time $\hat{T}(v, v)$ for a $v \rightarrow p \rightarrow v$ timed echo, when $p = \hat{p}$ be

$$\hat{T}(v, v) = 2 \cdot D(v, \hat{p}) + \Delta_v \tag{2.3}$$

and $T_p(v, v)$ be its observed value from the actual measurements. Following the point-in-triangle test, VM administers the test, where δ is the expected measurement error. This test is used to decide whether or not a verifier v should be satisfied by the time taken by p to respond to its challenge. Under the δ test, the verifiers accept the claimed location $\hat{p}$ if $|T_p(v, v) - \hat{T}(v, v)| \leq \delta$ for all $v \in V$. If the prover occupies a location other than the

claimed location $\hat{p}$, then $|T_p(v,v) - \hat{T}(v,v)| > \delta$ for at least one verifier, and p's location claim will not be accepted.

VM has been used for secure localization in sensor networks, and in radio frequency identification (RFIDs) in the work by Tippenhauer et al. [65].

B. Multiple Input Multiple Output (MIMO) Time-based Localization

In each round of a MIMO protocol, the prover p must respond to multiple challenges (inputs) generated by multiple verifiers $\{v, w, u, ...z\} \in V$. Each of the participating verifiers also monitors the prover's response (output), therefore, we term protocols with such a message exchange structure as *Multiple Input Multiple Output (MIMO)* localization protocols. Therefore in each round of MIMO localization, multiple verifiers simultaneously time their own challenge-response echoes. Similar to SISO localization protocols, the prover's location in MIMO protocols is also computed as the intersection of the circular constraints generated with individual verifiers. Fig. 2.4 shows the space-time diagram, and the constraints generated in a MIMO localization protocol.

Chiang et al. [8] introduced a MIMO protocol for secure time-based localization. They extended the *VM* protocol of Capkun and Hubaux to create a Time-of-Arrival (ToA) MIMO protocol for secure location verification by a group of N synchronized verifiers. However Chiang et al.'s protocol differs from VM, in that uses *simultaneous multilateration*. Instead of taking turns to execute a challenge-response dialog with the prover, all the verifiers simultaneously execute challenge-response echoes. Their protocol also incorporates extra features. For example, it requires the prover to have multiple radios for receiving multiple challenges simultaneously. Also, each verifier self-jams the frequency over which it

communicates with the prover, before and after sending its challenge. These extra features are used because the motivation behind their protocol because it is designed to address distance fraud by multiple colluding provers, in addition to distance fraud committed by a single prover.

During the execution of this protocol, the verifiers begin by communicating amongst themselves over a secure channel to generate an N-part challenge and agree on a common arrival time, τ, when all parts of the challenge must reach $\hat{p}$. Clearly the requirement $T(v,p) = T(w,p) = \cdots \equiv \tau$ can be satisfied by choosing the start time as $\tau - D(v,\hat{p})$ for all $v \in V$. By design, p cannot generate its response until it has received all N parts from the challenge. For this protocol, the response is formed by the bit-wise XOR across all N parts of the challenge and a pre-arranged section of a shared secret key. If prover p is honest and claims its true location, it will broadcast its response to all verifiers at time $\tau + \Delta$. Each verifier $v \in V$ hears the response at time $\tau + \Delta + D(p,v)$, therefore, its measurement of the running time for the protocol satisfies $T_p(v,v) = \hat{T}(v,v)$.

From the space-time diagram for MIMO multilateration shown in Fig. 2.4(a), we can observe how in each round, multiple challenges are sent to the prover simultaneously. This accounts for *many more messages in each round of challenge-response as compared to the SISO method.*

To protect against distance fraud, in addition to simultaneous multilateration, Chiang et al.'s MIMO protocol uses the point-in-triangle and the δ tests, similar to the SISO VM. First, p's location must satisfy the point-in-triangle test. If p is honest, then its response also passes the δ test with each verifier since $|T_p(v,v) - \hat{T}(v,v)| \approx 0 \leq \delta$ for all $v \in V$. Conversely, if the prover is dishonest and occupies a location other than the claimed

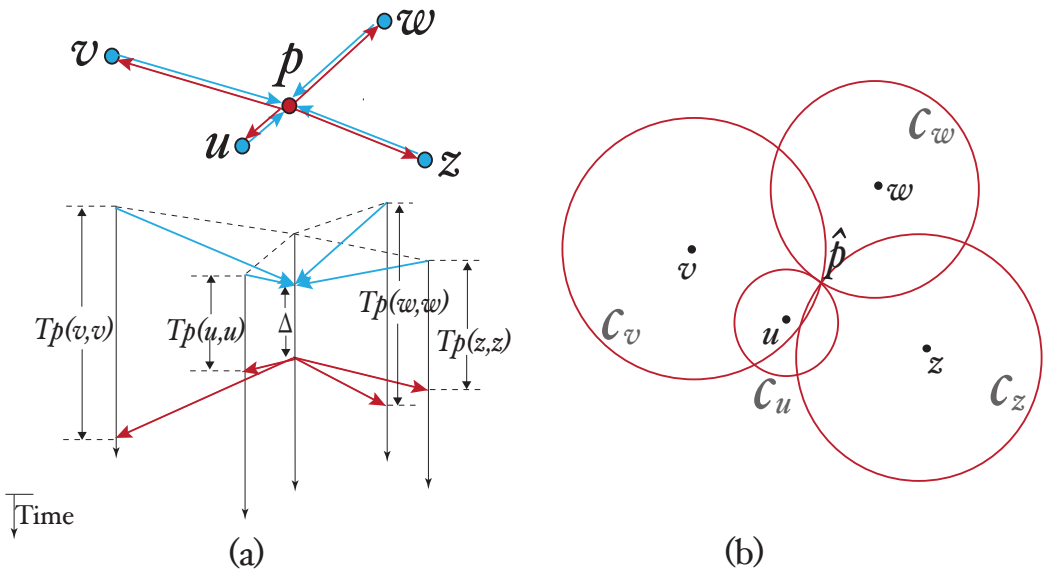


Figure 2.4: (a) Space-time diagram for MIMO multilateration - each verifier sends its own challenge at a scheduled start time such that the prover receives all the challenges simultaneously (b) Circular constraints generated by all the participating verifiers in MIMO multilateration

location, the challenges (inputs) from different verifiers will reach p at different times, not simultaneously, since the start time is still $\tau - D(v, \hat{p})$ for all $v \in V$. Because of the point-in-triangle test, there must be at least one verifier, say z, for which $D(z, p) > D(z, \hat{p})$, thus the response arrives late. Therefore, p will fail the δ test administered by z, and hence its location claim is not accepted.

To the best of our knowledge, there are no other works based on the MIMO localization protocol of Chiang et al. The use of simultaneous challenges in MIMO localization is believed to enhance its security against distance fraud. However, we will show in chapter 4, that this feature causes MIMO localization protocols to be less accurate than the other classes of time-based localization protocols, when they are implemented in real systems.

C. Single Input Multiple Output (SIMO) Time-based Localization

In each round of a SIMO protocol, the prover p receives a challenge from only a single verifier, which we will denote as the "lead" verifier. p's response, however, is observed by multiple witnesses, say $\{w, u, ...z\} \in V$, in addition to the lead verifier, say v, that sent the challenge. Since a single verifier sends the challenge (input), but multiple verifiers observe the response (output), we term protocols with such a message exchange structure as *Single Input Multiple Output (SIMO)* localization protocols. Fig. 2.5 shows the space-time diagram, and the constraints generated in known SIMO localization protocols.

Notice that in each round of a SIMO localization protocol, the lead verifier executes a challenge-response echo with the prover, while multiple witnesses simultaneously observe challenge-response relays. Therefore SIMO localization uses *simultaneous multilateration* to defend against distance fraud.

Existing SIMO protocols are Time-Difference-of-Arrival (TDoA) [40] localization protocols, and use the principle of hyperbolic multilateration. Hyperbolic multiltaeration has a long history of application to surveillance and navigation systems[67]. Here three or more synchronized receivers at known locations can jointly determine the location of another node based on the TDoA of its transmission(s). The same method also allows a single receiver to determine its own location from the time-difference-of-arrival of synchronized transmissions originating from multiple locations, such as in the Global Positioning System. Many such SIMO localization protocols [36, 34, 69] have been proposed for wireless environments also, but they do not address security.

In *secure* SIMO hyperbolic multilateration, a single verifier, say v sends the challenge, but N verifiers independently record the arrival time for the prover's response. In

general, it is assumed that the clocks of the verifiers are synchronized [5, 61]. SIMO hyperbolic multilateration eliminates *distance fraud* attacks by removing Δ from the equations for determining p's location. For example, suppose verifiers v and w both use synchronized clocks to timestamp their respective arrival times for the same response from p, τ_v and τ_w, as shown to the right edge of Fig. 2.5(a). Even though neither verifier knows p's true location, both verifiers are timing the *same message* – which left p at they same time, and thereafter took $D(p, v)$ to reach v and $D(p, w)$ to reach w. Thus

$$D(p, v) - D(p, w) = \tau_v - \tau_w \tag{2.4}$$

whose solution is lobe $\mathcal{H}_{vw}$ of the *hyperbola* with foci v and w. [2]

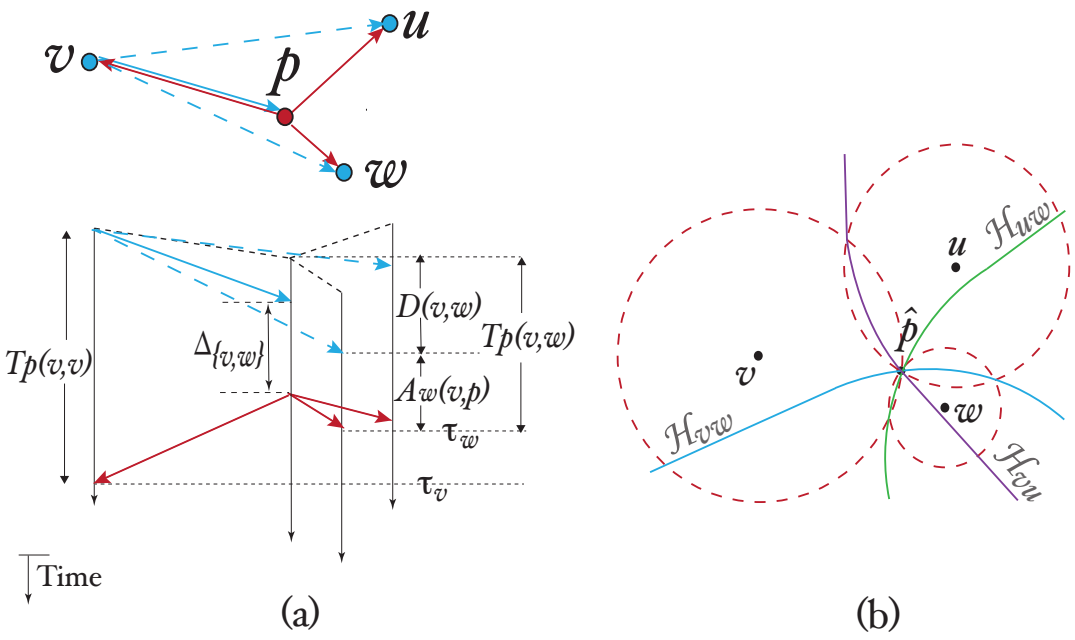


Figure 2.5: (a) Space-time diagram for SIMO multilateration - a single verifier sends its challenge. The prover and all the witnesses receive the challenge. (b) Hyperbolic constraints generated by three verifiers participating in SIMO hyperbolic multilateration.

[2]Note: a hyperbola with foci at a and b will be denoted as H_{vw} in this dissertation.

Saha and Molle [55, 54] later showed that a set of verifiers can obtain the same results without synchronizing their clocks in advance. Their protocol *Localization with Witnesses* is also a secure SIMO time-based localization protocol that uses hyperbolic multilateration. In their protocol, a single verifier, say v, is chosen to engage in a packet-based SISO challenge-response echo with prover p, over an RF broadcast channel. At the same time, each of multiple *witnesses* (receive-only verifiers), observes a challenge-response relay, and timestamps both v's challenge and p's response to determine their *interarrival time*. For example, the inter arrival time measured by witness w is $A_w(v,p)$, as shown along the right edge of Fig. 2.5(a). Since all verifiers know each other's exact positions, w can easily find the running time for a SISO challenge-response *relay* along the path $v \to p \to w$, by adding the known distance $D(v,w)$:

$$T_p(v,w) = A_w(v,p) + D(v,w) \equiv D(v,p) + \Delta_{\{v,w\}} + D(p,w) \tag{2.5}$$

Although none of the verifiers know the exact value of $D(v,p) + \Delta_{\{v,w\}}$, it is common to the running times measured by each verifier, hence vanishes when we form pair-wise differences such as:

$$\begin{aligned} D(v,p) - D(w,p) &= T_p(v,v) - T_p(v,w) \\ &= A_v(v,p) + D(v,v) - (A_w(v,p) + D(v,w)) \end{aligned} \tag{2.6}$$

Notice that Eqs. (3.6) and (2.4) are just different ways to express $(\tau_v - \tau_w)$ – the difference in the arrival times of the response at v and w. Therefore, either equation constrains the prover to the same lobe of hyperbola VW in Fig. 2.5(b).

Although other secure time-based SIMO localization protocols based on hyperbolic multilateration exist, to the best of our knowledge, Localization with Witnesses is the only protocol in this category that demonstrates synchronization-free localization.

Table 2.1: Comparing different classes of secure time-based localization protocols

Class	**SISO**	**MIMO**	**SIMO**
Method	Time-of-Arrival (ToA)	Time-of-Arrival (ToA)	Time-Difference-of-Arrival (TDoA)
Protection from Distance Fraud	Point-in-Triangle test and δ test	Simultaneous Multilateration, Point-in-Triangle test and δ test	Simultaneous Multilateration
Examples	Capkun and Hubaux [6], Tippenhauer et al. [65]	Chiang et al. [8]	Saha and Molle [54], Shmatikov et al. [61], Capkun et al. [5]

2.2 Timing Resolution of Time-Based Localization Protocols

In section 2.1, we discussed existing work that addresses the "correctness" criteria for secure time-based localization protocols. In this section, we focus on work that addresses the issue of proper timing resolution. To ensure proper timing resolution of a time-based localization protocol, we must ensure that the ranging[3] mechanism it uses can be realistically implemented with available hardware. Furthermore, we must ensure that the message propagation times can be measured with the required accuracy during ranging. For time-based ranging over narrowband RF, the desired accuracy is in the order of nanoseconds. Recall that our goal is to ensure proper timing resolution for localization in 802.11-based networks, where narrowband RF[4] is the de facto medium of propagation. Therefore, we

[3]Ranging is defined as the act of measuring the physical distance between two entities.

[4]We use the term "narrowband RF" here to distinguish it from Ultra-Wideband RF. In the rest of this dissertation, we will simply use "RF" to refer to narrowband RF.

will discuss only those systems where time-based ranging has been *implemented* using narrowband RF. This allows us to study how the existing prototype implementations address the specific challenges that arise due to RF being the medium of propagation. Although systems that use other mediums like Ultra-Wideband (UWB) [14, 17, 18], infrared and ultrasound [48, 59] are known to achieve high precision in time-based ranging, we do not include them here, since they are not used widely in 802.11-based networks.

In chapter 1, we explained that the speed of the RF medium is greater than that of any other propagation medium used in wireless networking. It is because of this that the entities that participate in ranging over RF, must be able to timestamp message arrivals and departures (equivalently, measure the message propagation times) with nanosecond-level precision. This would require the entities to detect the reference symbol with nanosecond precision *and* record a timestamp of nanosecond-level resolution. Neither of these capabilities is currently supported by 802.11-standard specifications and hardware. With the current support for timestamping message arrival events, the magnitude of error introduced into the measurements is high enough to impede 802.11-compatible systems from achieving the required timing resolution.

In chapter 5, we will present an anatomy of the error introduced into the timestamps captured for message arrival and departure events. In doing so, we identify and discuss in detail each factor that contributes to error in the measurements. We also present ways to correct for these errors so that 802.11-compatible entities can achieve the timing resolution required for a solution to our problem.

In the survey that follows, we discuss existing implementations of time-based ranging/localization over RF. Most of the existing prototype implementations were designed for

sensor networks. Since RF is also the widely used medium in sensor networks, these prototypes address many challenges that we will need to address for implementing time-based ranging in 802.11-based networks. The survey is organized into three main parts. In the first part, we discuss the existing methods for detecting message arrival times accurately. In the second part, we show how the clocks of entities are synchronized in wireless networks. This discussion is important because lack of synchronization introduces significant amount of error into the measurements. The third part focuses on techniques used to minimize errors due to other factors like signal processing delay in the receiver and channel effects.

2.2.1 Detecting Message Arrival Times Accurately

In order to measure the propagation time of a message between a sender and a receiver accurately, the receiver must be able to determine the exact time of arrival of the message. In this section, we discuss the most widely used techniques for accurate detection of message arrival events.

Through Autocorrelation of the Received Signal

In GPS systems, time of arrival of a signal is detected accurately by correlating the received signal with a locally available copy of the same signal at a receiver. Autocorrelation measures the similarity between a digital sequence and its shifted version. If the digital representations of two identical signal sequences are multiplied point by point, and the result is time averaged, a non-zero value is obtained only when the sequences are aligned. The received signal in GPS systems is a known pseudo random noise sequence (PRN). When an autocorrelation is performed between the received signal and a local copy of the same

PRN sequence, the location of the non-zero peak on the time axis indicates the time of arrival of the signal accurately.

Autocorrelation has also been employed for accurate time-of-arrival detection in sensor networks. In Lanzisera et al.'s ranging system [30], a sensor mote, say A, sends k copies of a finite length ranging signal modulated on an RF carrier. The receiving mote collects these copies, demodulates them, computes the time-average, and stores a circularly shifted copy of the received sequence. This shifted copy is sent back to mote A after a predetermined interval. Range extraction occurs *offline*, when the motes are no longer exchanging ranging messages. Mote A performs an autocorrelation between the copy that it sent and the copy that it received. The peak indicated the exact time at which it receives the shifted copy of the message that it sent. Knowing the sending time, the receiving time, and the time taken by the other mote to respond, it can accurately compute the propagation time of the message between the two motes. To generate a message consisting of k copies of a baseband PRN sequence, Lanzisera et al. used custom hardware. In particular, they used an FPGA module connected to an off-the-shelf transceiver (made for sensor networks) to generate the baseband sequences, as well as to compute the time of arrival by autocorrelation.

In Lanzisera et al.'s implementation, each entity has the ability to perform autocorrelation. However, in some systems designed for 802.11-based networks, autocorrelation is performed in a centralized manner to extract the timing information. This alternative is an option where the time-of-arrival need not be determined in real-time, and where the infrastructure supports offloading of received signal contents to a central server. The individual entities in the network only store a raw trace of the incoming signal. Later, they

send the raw A/D traces to the central authority, which performs the autocorrelation and returns the time of arrival of the signal. Examples of such systems are the Aeroscout system [1, 29], and the time-based localization system proposed by Yamasaki at al. [69]. Both of these systems conform to the 802.11 b/g standard, which shows that this technique can indeed be employed in 802.11-based networks. The limitation is that this technique has not yet been applied in self-organizing, ad-hoc 802.11-based networks. This is because currently available transceivers do not support recording of raw A/D traces and computing the autocorrelation function, although this should not be a difficult change for future designs.

Increasing the Clocking and/or Sampling Frequency

In standard 802.11 transceivers, the standard way to detect the time-of-arrival of a message is to process the received signal and detect the reference symbol in the incoming message. The received signal has to go through a defined set of signal processing stages after being detected at the antenna, before the reference symbol is detected and a timestamp capture triggered. The signal sampling frequency and the resolution of the timestamping clock have significant impact on the accuracy of the timestamp captured for the arrival event. Karalar et al. [27] showed that to achieve timing accuracy in the order of a nanosecond, an OFDM signal must be sampled at $150MHz$ or greater.

The concept of sampling at a higher frequency to increase the accuracy of the timestamp captured for the arrival event was used in work by Li et al. [31]. In their implementation, they could limit the error in timestamping to $20ns$ by upsampling the received signal. Instead of the conventional $11MHz$ sampling for DSSS, they used an upsampling factor of 9 to sample the received signal at $99MHz$. Obtaining many more samples during

A/D conversion lowers the error during interpolation and symbol reconstruction in the timing recovery loop[5]. Yamasaki et al [69] also used a higher sampling rate to further increase the accuracy of the time-of-arrival detection done by correlation. In IEEE 802.11b, the chip rate is $11MHz$. This allows for a resolution of $27m$ in their system, which is not sufficient for meaningful localization. They up-sampled the received signal to increase the resolution to $21cm$, thus enabling accurate time-based localization with the same system.

The PinPoint system [70] is a time-based location determination system where a higher clocking frequency is used in addition to other non-standard hardware features to facilitate high accuracy in ranging. Instead of the $20-80MHz$ crystal oscillator commonly used in off-the-shelf hardware [57], the PinPoint prototype implementation uses a phase locked loop to clock the timestamping unit at a much higher frequency. This phase locked loop runs on the Cyclone 1C20 FPGA development kit that they use to prototype the system, and clocks the timestamping unit at $300MHz$. This high clocking frequency allows their system to capture timestamps with a resolution of $3ns$.

Leveraging the Verier Effect in Conjunction with Statistical Averaging

In the previous section, we discussed systems where high accuracy in measuring the propagation times (equivalently, distances) has been achieved by increasing the clocking and/or sampling frequency. In some situations however, using high clock frequency is not desirable. Since increased clock frequency leads to increased power consumption, use of higher frequencies is not a good method when conserving power is important. This is especially true in sensor networks and 802.11-based networks with handheld mobile entities.

[5]Details in chapter 5.

To achieve resolution in the order of a nanosecond with low frequency clocks (the kind used in off-the-shelf sensor network hardware), Thorbjornsen et al. [63] proposed a method that leverages the vernier effect [28]. Simply stated, two heterogenous clocks, having a small offset, can be used to generate a virtual time resolution, which is greater than that of the individual clocks. In particular, the virtual resolution is equal to the instantaneous offset between the two clocks. Consider two entities x and y whose clocks have similar time periods but a small offset between them. Suppose the true propagation time between x and y is $N + n$ clock periods, where N is an integer and n is a fraction. Since the entities can detect the arrival of a symbol only at a leading edge of the clock, depending on the value of the offset, y detects the arrival of the message either after $N+1$ or after $N + 2$ clock periods. When y sends a response after a known delay, x can also detect its arrival only at the leading edges of its clock. After subtracting the known delay, x's measurement for the two-way round trip time equals either $2N+1$ or $2N+2$ clock periods. For a detailed explanation of this, see [63] and chapter 5 of this dissertation. Over multiple two-way message exchanges, let the number of measurements of x that equal $2N + 1$ be m_{low}, and the number of measurements that equal $2N + 2$ be m_{high}. The true message propagation time between x and y is then determined with sub-clock accuracy by statistical averaging:

$$T(x, y) = \frac{m_{low}(2N + 1) + m_{high}(2N + 2)}{m_{low} + m_{high}} \tag{2.7}$$

Since the accuracy obtained through averaging increases with the number of samples, this technique requires a large number of measurements to ensure high accuracy. By using this technique, Thorbjornsen et al.'s implementation could measure message propagation

times with sub-clock accuracy, using 802.15.4- compliant hardware, and the Zigbee message format. In their experiments, they were able to estimate distances with an accuracy in the order of 10 meters. While the advantage of this technique is that it works with the standard message format and hardware for sensor networks, the major drawback is that the required number of measurements is large.

Other methods for detecting the Time-of-Arrival

Other than these widely known techniques for accurate detection of the time-of-arrival of a message over RF, there exist a few other lesser known techniques. In the ranging system proposed by Karalar and Rabey [27], a transformation of the received signal to the frequency domain is used to easily detect the time of arrival. The OFDM signal is first down-converted at the receiver, then digitized and transformed to the frequency domain. The channel frequency response is then computed using the frequency domain representation. The computed channel frequency response is transformed back to the time domain. The strongest channel tap in the time-domain representation indicates the time-of-arrival of the signal.

In work by Geiger [21], matched filtering is used to detect the arrival time of a standard 802.11 DSSS signal. The author used code provided by the BBN ADROIT project [2] to construct a software defined radio, but added some extra signal processing blocks to employ his technique. The added blocks consist of a block to compute the time-averaged power, a differentiator, threshold detector and a peak finder. Computing the slope of the time-averaged power enables a gross measurement of the start of frame, while the peak finder provides finer grained measurement to determine the time of arrival with high

accuracy. Geiger's method allows for a resolution of $40ns$ in the timestamps captured for message arrival events.

2.2.2 Synchronizing Clocks of Participating Entities

Accurately detecting the time-of-arrival with respect to the receiver clock alone does not suffice for obtaining high accuracy timestamps. In addition, the receiver must accurately know the timing relationship between its clock and the sender's clock. To synchronize the clocks of entities over the wireless medium, two approaches have been used in existing literature [27].

If the message transfer is only one-way between a sender x and receiver y, then *signals with different speeds* may be used for synchronization and ranging purposes. x simultaneously sends two copies of the same message using a slower and a faster medium, for example, ultrasound and RF. The receiver y must measure the interval between the arrival of the two copies. Since the signal velocities are known, and the receiver knows the difference in the arrival times of both signals, it can compute the distance between the sender x and itself. It can also compute the offset of its own clock with respect to the sender's clock. This method of ranging/synchronization has been used in sensor networks [48, 60, 39], but is not the preferred method. The reason is that the entities are required to possess highly directional, expensive and high power transducers to use this technique. Equipping each entity with such hardware is undesirable because it leads to a significant increase in cost and power consumption.

Two way time transfer is the more widely used method to achieve synchronization over a wireless network. If the clocks of two entities are not synchronized, the clock offset

between the entity clocks appears as an additive term in one direction, and a subtractive term in the reverse direction in two-way message exchanges. Since the overall effect of the offset is cancelled out when two-way measurements are made, the error due to the offset can be removed by averaging across multiple two-way measurements. Two-way time transfer works under the assumption that the clock offset is constant during the forward and reverse transmissions. Therefore. measurements should be made in rapid succession. It is worth noting that secure localization protocols also require rapid two-way challenge-response message exchanges; therefore, this method of synchronization can be easily integrated with a time-based localization protocol.

2.2.3 Error Correction Techniques for Improving Accuracy

The factors that add to the error in the timestamps captured for the arrival of a message are (1) positioning of the timestamping unit in the network protocol stack; (2) synchronization between the clocks of the participating entities; (3) hardware and software processing delays; (4) channel effects like multipath and non-line-of-sight (NLOS) conditions.

The positioning of the timestamping unit in the network protocol stack is the most important factor for accurate timestamp capture. Ideally, the timestamping unit must be as close as possible to the point where the reference symbol is detected. For example, in 802.11-compatible receivers, the symbol detection module is within the physical media dependent (PMD) layer of the PHY hardware. However, 802.11-compatible entities currently do not support timestamping within the PHY hardware. Instead, timestamping is supported only through the TSF timer function in the MAC layer. Many non-deterministic delays are

incurred within the PHY hardware, and at the PHY-MAC interface, before the timestamp for an arrival event can be captured by the TSF timer. This limits the resolution of the timestamps to approximately $1\mu s$ in 802.11-compatible entities.

This problem also exists for time-based ranging in other kinds of wireless networks, for example, sensor networks. To minimize the delay between detecting the arrival of a message and capturing a timestamp for the event, many researchers have proposed timestamping in hardware rather than software. This eliminates the delays incurred in capturing a timestamp in a higher layer. For example, the timestamping module may be built on an FPGA, which is interfaced to the signal processing blocks of a regular receiver [26, 34, 38, 53] or placed at the PHY-MAC interface [53]. Many implementations for time-based ranging in sensor networks [30, 70] use custom hardware instead of commercially available hardware so that they have the flexibility to place the timestamping unit close to the point of symbol detection. Chapter 4 of this dissertation is dedicated to addressing this issue in 802.11-compatible entities.

Correcting for Processing Delays

Signal processing delays also add error to the measurements for the time-of-arrival of the signal. If the ranging system is designed for a specific wireless standard like 802.11or 802.15.4, then the messages must be framed according to the standard specifications. The raw symbols undergo signal processing operations like spreading, modulation, up-conversion, etc., before transmission. On the receiver end, the reverse processes must be applied to the received signal, i.e., down-conversion to baseband frequency, demodulation, despreading, etc. Therefore, there is a delay between the time that the signal actually

arrives at the receiver antenna, and the time at which its arrival is detected. This delay is the cause for error in the timestamps captured for the arrival event.

Some implementations use a non-standard messaging format, where these signal processing steps are not required. For example, the PinPoint system [70] uses a repetitive pattern of baseband pulses, where after every 20 baseband cycles, one cycle of "dense" pulses are sent. The cycle of dense pulses acts as the reference symbol in this system. This message format does not require complex signal processing before the detection of the reference symbol, therefore error due to processing delays can be minimized.

Pre-calibration is a better known technique to correct for processing delays, without having to use a non-standard message format. In this technique, the delay due to signal processing is *estimated before* the ranging process. During the pre-calibration phase, the sender and receiver are placed adjacent to each other. Since the signal practically travels zero distance, the error in the measurements for this setup can be attributed to the signal processing delays alone (assuming that the error due to other factors has been remedied in some way). Based on the value of the estimated delay, a correction is applied to every measurement made in the ranging phase. Such a pre-calibration stage has been used in many known implementations for time-based ranging over RF [37, 63, 27].

In existing 802.11b receivers, there is no compensation for the delay incurred in the analog components (analog front end), the timing recovery loop, and the other signal processing units that the signal encounters, before its arrival at the receiver is detected. To compensate for these signal processing delays, Exel at al. [16, 15] proposed modifications to COTS 802.11b receivers. The most important architectural change in their implementation is addition of a module that can record the delay experienced within the timing recovery

loop, and propagate its value to the timestamping module. The timestamping module then uses this value to apply a suitable correction to the timestamp recorded for the arrival event. Exel et al.'s prototype also ensured that the delay incurred in signal processing in units other than the timing recovery loop is deterministic. This known value is also used to correct the timestamp recorded. They also showed that the delay in the analog front end is dependent upon the current gain settings of the receiver. This delay can be estimated using the manufacturer's specifications, therefore a correction for the delay in the analog front end can also be applied.

Minimizing Channel Effects

Channel effects like multipath and non-line-of-sight (NLOS) propagation also affect the quality of measurements in time-based ranging. When a sender sends a signal over the wireless network, the receiver receives multiple copies of it – one directly from the sender and others from the signal bouncing off of surrounding objects. The non-line-of-sight copies arrive with a shift and interfere with the line-of-sight signal, causing it to distort. This makes it difficult for the receiver to detect the true time of arrival of the message.

The PinPoint system [70] bases its calculations on the first and longest chain of baseband signals to reduce the effect of multipath. Note that in this system, it is easy to distinguish between the direct and reflected signal because of the non-standard message format. When a system uses a message and signaling format that is consistent with a wireless networking standard, it is not easy to make this distinction. Therefore, distinguishing and using only the first arriving signal might not be a feasible solution for systems that use standard messaging formats.

In some systems [27], the error due to channel effects like multipath is estimated by pre-calibration, similar to estimating error due to processing delays. To calibrate the system for channel effects, measurements are first made by placing the sender and receiver at known distances. For each measurement made for calibration, the distance between the entities is increased by a known amount and the corresponding increase in error is measured. Assuming that the error due to signal processing delays, synchronization and other factors remain constant, the increase in the error is attributed to the channel effects alone. The error due to the channel effect for an unknown distance between the entities is then computed through interpolation of the data. This information is used to make suitable corrections to the measurements collected during the actual ranging phase.

It has been shown that the error introduced into measurements due to the multipath effect is frequency dependent [66]. This frequency dependence can be leveraged to reduce the error due to multipath. In the system proposed by Lanzisera et al.[30], error due to multipath is mitigated by carrier frequency hopping.

Minimizing Overall Error Due to Various Factors

In the previous subsections, we mentioned techniques used to correct for error introduced by each factor that affects the accuracy of a timestamp recorded for the arrival event. Some systems approach error reduction by considering the total error, instead of correcting for error introduced by each factor. MacCrady et al. [35] showed that the total time delay experienced by the signal is a gaussian random variable formed by summing each of the independent components. If multiple two-way message exchanges are performed, then the variance in the timestamp for the arrival events is expected to reduce by the square root

of the number of transactions. Therefore, the overall error can be reduced either by reducing the variance in each component, or by averaging over a large number of message exchanges. Some of the known implementations for time-based ranging [37, 63] adopt the latter method to minimize error in the measurements. Averaging over hundreds of measurements, these systems can estimate message propagation times over RF, with an accuracy in the order of a nanosecond.

Summary

The following table summarizes features of the existing prototype implementations for time-based ranging:

2.3 Other Related Implementations

In the section 2.1, we mentioned numerous known secure time-based localization protocols. Although some of these protocols were first developed more than a decade ago, there exists almost no work on practical implementation of these protocols, especially with 802.11-compatible hardware and software, over radio frequency(RF). As mentioned earlier, the reason for this is that almost all of the literature on secure localization comes from the cryptographic research community. Authors from this community have paid little or no attention to the issues that arise in practical implementation of these secure protocols.

Known secure time-based localization protocols make many impractical assumptions. For example, the distance bounding protocol [4] and the MIMO localization protocol proposed by Chiang et al. [8] assume that the challenge and response messages are single

Table 2.2: Prototypes for Time-Based Ranging Over RF

Work By	**Lanzisera et al. [30]**	**Youssef et al. [70]**	**Karalar et al. [27]**	**Thorbjornsen et al. [63]**	**Mazomenos et al. [37]**	**Exel et al. [16]**
Signaling	PRN sequence, 2.4 GHz	Sequence of Baseband Pulses, 2.4 GHz	OFDM, 2.4 GHz	ZigBee (802.15.4) Frames, 2.4 GHz	ZigBee (802.15.4) Frames, 2.4 GHz	802.11b Frames, 2.4 GHz
Hardware	COTS transceiver + FPGA	Altera Cyclone IC20 FPGA + Maxim 280 Radio	FPGA + RF dev Boards	TI CC2430 Dev Kit	TI CC2500 Dev Kit	SMiLE Transceiver + Stratix II FPGA
Clocking Freq	25 MHz	300 MHz	100 MHz	32 MHz	16 MHz	44 MHz
Time of Arrival Computation	Autocorrelation	Marker Detection	Channel Impulse Response through FFT	Vernier Effect + Averaging	Vernier Effect + Averaging	SFD Detection
Synchronization	Code Modulus Synchronization (CMS)	Unsynchronized, Mathematical Compensation for Offset	Two-Way Time Transfer	Two-Way Time Transfer	Two-Way Time Transfer	Syntonization through TDoA
Mean Error	$2.6m$	$6.8m$	$2.0m$	$6.7m$	$2.5m$	$13.5m$

bits. This assumption is impractical because none of the wireless networking standards allow for zero length messages. Many authors also make the assumption that message arrival and departure events can be timed with nanosecond-level of accuracy, which is indeed a requirement for reasonable accuracy in localization over RF. This is extremely difficult to do in real systems, with off-the-shelf-hardware. In chapter 4, we elaborate further on this difficulty.

Realizing that single bit exchanges are impractical, authors later proposed the use of bitstreams, or of standard network packets with markers, as the challenge and response

messages. Although this solves the first problem, it still does not address the problem of recording timestamps for message arrivals and departures over RF with nanosecond-level accuracy. The only known implementation of distance bounding over RF [51], achieves this level of resolution by making many modifications to the distance bounding protocol, and uses non-standard transceiver hardware. Firstly, the authors showed that any operation that requires the prover to demodulate and modulate the challenge cannot meet the timing requirement. This includes the XOR function, which is the defacto operation performed on the challenge bits in existing protocols. The authors instead used *Channel Reflection and Channel Selection (CRCS)*, which is a function that allows the prover to respond without demodulating the challenge from the analog domain. Note however, that to do this, the prover must use hardware that is different from 802.11-compatible hardware. Instead of a standard transceiver, where the received signal follows a defined set of signal-processing stages from demodulation to symbol detection, they used hardware with two radios. The received signal from one radio is not demodulated completely, rather the CRCS function is applied to it and it is immediately transmitted from the other radio. Therefore, CRCS requires the ranging messages to be processed in a manner different from processing of regular data packets. This is undesirable for our purposes, where we are aiming at integration of localization capability with the prevalent 802.11 wireless networking capabilities. Singelee and Preneel [62] later added additional features to this implementation to defend against a different kind of attack, but their work still does not address the issue of implementation/integration with 802.11-based networks.

Recently, the idea of synchronization-free TDoA localization, first introduced in [54], has been implemented in *Whistle* [68]. Although this implementation shows that

synchonization-free TDoA hyperbolic multilateration is indeed a possibility, Whistle uses the acoustic medium and cell phone hardware. To the best of our knowledge, implementation of synchronization-free TDoA localization over RF, with the standard 802.11 network protocol stack and hardware, has not yet been attempted.

2.4 Conclusions

In section 2.1, we surveyed the existing secure time-based localization protocols. We learnt about the requirements for designing the message exchange structure for secure localization. In particular, we learnt that a time-based localization protocol must use multilateration. To secure a protocol against distance fraud, we can use techniques like the point-in-triangle and δ-tests introduced by Capkun et al. [6], use simultaneous multilateration, or use a combination of all these techniques. We also learnt that although simultaneous multilateration is a desirable feature for secure localization, simultaneous challenges (MIMO multilateration) are not essential for achieving simultaneity in multilateration. We can achieve the same effect by using multiple receivers in each round of challenge-response (SIMO multilateration). Although the protocols discussed in this section satisfy the "correctness" criteria because of their ability to securely localize the prover, none of them have been implemented on real-word 802.11-based systems. We attribute the primary reason for this to features like zero-length messages, non-standard hardware requirements, etc., which do not comply with the 802.11 specifications.

In section 2.2, we surveyed existing prototype implementations for time-based ranging over RF. We observed that most of these prototypes are customized for sensor network implementations. Since sensor networks also widely use narrowband RF as the

medium of propagation, studying the techniques used in these prototypes to achieve meter-level accuracy (in distance estimates) provides us with a good start to achieve the same in 802.11-based networks. By studying prior work in this area, we learnt that the position of the timestamping unit relative to the network protocol stack is very important for time-based ranging. We also studied the techniques that can be employed for synchronizing clocks of different entities over wireless, so that the error in recording the time-of-arrival (ToA) and time-difference-of-arrival (TDoA) of incoming signals can be minimized. In addition, we also learnt that pre-calibration, statistical averaging and removal of outliers can significantly reduce the error in distance estimates.

In section 2.3, we studied other related work, which demonstrate that secure time-based localization has been implemented in other setups like cellular networks, or with customized hardware.

In conclusion, we find that practical realization of secure time-based localization protocols with 802.11 wireless networking standard hardware and specifications, is still an unsolved problem. In the subsequent chapters we solve this problem by leveraging some of the ideas from prior work, while introducing a new protocol, architectural adaptations, and new algorithms to enable secure time-based localization in 802-11-based networks.

Chapter 3

Elliptical Multilateration – A New SIMO Secure Localization Protocol

In chapter 1, we introduced and defined the two criteria that must be satisfied in the design of secure and accurate time-based localization protocols – correctness and proper timing resolution. After surveying the prior work on time-based localization protocols in chapter 2, we found that none of the existing protocols simultaneously addresses both these criteria. Moreover, features of most protocols do not comply with 802.11 standard hardware and specifications.

In this chapter, we introduce a new secure time-based localization protocol, dubbed "Elliptical Multilateration (EM)". Our protocol simultaneously addresses both correctness and proper timing resolution, however, the discussions in this chapter will be focussed on the issue of "correctness". In subsequent chapters, we will cover the issues related to "timing resolution", and show how our EM protocol performs in this regard.

In section 2.1, we classified time-based localization protocols, which have been

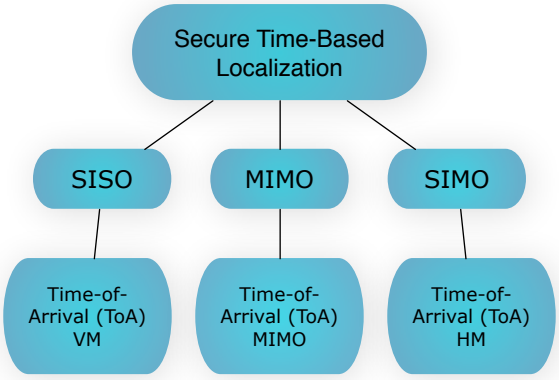


Figure 3.1: Classification of existing secure time-based localization protocols

formally proven to be "correct", into three classes: SISO, MIMO and SIMO. We found that existing SISO and MIMO protocols are time-of-arrival protocols, whereas the existing SIMO protocols are time-difference-of-arrival protocols. This distinction can be easily seen in table 2.1 and in the pictorial depiction shown in Fig. 3.1. Our EM protocol shares the message structure of SIMO hyperbolic multilateration (HM). However, it is a time-of-arrival (ToA) multilateration protocol similar to known SISO and MIMO protocols. Therefore the introduction of EM creates a new subclass of SIMO protocols. Fig. 3.2 illustrates this addition to the existing classification of these protocols.

In the following sections, we consider security features that protect existing protocols from distance fraud launched by a single prover. We also consider the message exchange structures across different protocols. We briefly mentioned these in section 2.1.3. In this chapter, we discuss SISO VM and SIMO HM in greater details. In the step-by-step discussion of Verifiable multilateration (VM), we explain its security feature – the point-in-triangle test administered in conjunction with the δ-test. Next, we describe how simultaneous multilateration with multiple receivers secures time-difference-of-arrival (TDoA) based Hyper-

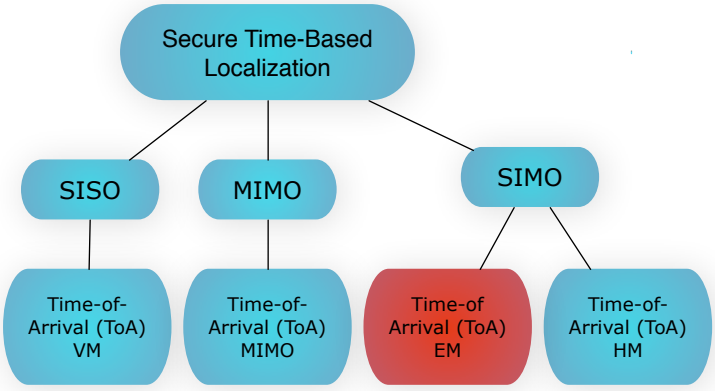


Figure 3.2: Our protocol Elliptical Multilateration (EM) is a Time-of-Arrival (ToA) multilateration protocol similar to existing SISO and MIMO protocols. Yet, it shares the message structure of HM, therefore it is a subclass of SIMO protocols.

bolic Mulilateration (HM) from distance fraud launched by a single resource-constrained prover. Step-by-step discussions of both protocols and their security features help us to better understand which security features work with the different kinds of message exchange structures. It also helps us to understand the advantages and disadvantages of these features under a similar threat model. Given the message structure and the threat model for a time-based localization protocol, we can then incorporate the right security features to ensure defense against distance fraud.

Next, we introduce our Elliptical Multilateration (EM) protocol, which incorporates the best security features from Verifiable Multilateration (VM) and Hyperbolic multilateration (HM). We claim that EM is secure against distance fraud in the presence of a resource-constrained, as well as a resourceful prover. Even in the presence of a resourceful prover, two rounds of challenge-response are sufficient for EM to detect distance fraud. We provide a formal proof to support our claim. We also show that under similar assumptions for the number of participating verifiers and resourcefulness of the prover, EM can securely

localize the prover in equal or fewer message exchanges as compared to existing protocols. Therefore, our EM protocol fully addresses the "correctness" criteria for secure time-based localization.

3.1 Features That Ensure Correctness in Existing Time-Based Localization Protocols

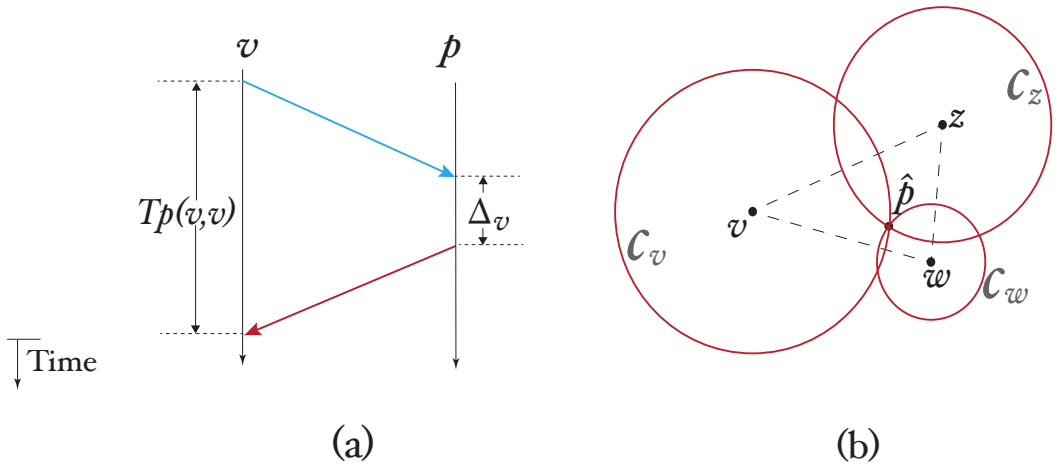


Figure 3.3: (a)Verifier v executes a challenge-response echo while passive verifier w observes a challenge-response relay (b) Circular constraints on the prover's location formed by three verifiers, each individually executing a challenge-response echo with the prover.

In a time-based localization protocol, each verifier must estimate its distance from the prover by executing a challenge-response message exchange with it. Fig. 3.3(a) shows a two-way message exchange comprising of a challenge from verifier v and the response from prover p. The interval

$$T_p(v, v) = 2 \cdot D(v, p) + \Delta_v \tag{3.1}$$

along its left edge shows the time to complete the challenge-response echo over the path

$v \rightarrow p \rightarrow v$. Verifier v can measure $T_p(v, v)$ directly, even though it does not know the prover's exact response time to this challenge, Δ_v. However, since $\hat{\Delta} \leq \Delta_v$ must hold; the verifier substitutes $\hat{\Delta}$ for Δ_v in Eq.(3.1),

$$\mathcal{C}_v = \{p : D(v, p) \leq (T_p(v, v) - \hat{\Delta})/2\} \tag{3.2}$$

to form a circular constraint $\mathcal{C}_v$ that upper bounds its distance from the prover.

For time-based location verification, we recast the problem into evaluating the evidence to support some hypothetical location $\hat{p}$, then we could substitute $\hat{p}$ into Eq.(3.1) and solve for

$$\nabla_v = T_P(v, v) - [D(v, \hat{p}) + D(\hat{p}, v)] \tag{3.3}$$

where ∇_v is the prover's *perceived* response time by verifier v if the location $\hat{p}$ is indeed correct. In Fig. 3.3(b), we show how *multilateration* can use a sequence of challenge-response rounds conducted by different verifiers to restrict p's location to the intersection of their respective circular constraints.

3.1.1 Verifiable Multilateration

Capkun et al [6] added the point-in-triangle requirement to sequential multilateration to block distance fraud in their protocol – *Verifiable Multilateration (VM)*. The steps for the execution of VM are shown in Algorithm 1. Lines 4-10 of VM resemble sequential multilateration. In each round of the protocol, a different verifier executes a challenge-response echo with the prover. By measuring the running time of the challenge-response echo, each verifier then forms a circular constraint on p's location using Eq.(3.2). The only difference between VM and sequential multilateration at this point is that in VM, each verifier also computes the perceived response delay of the prover using Eq. (3.3), as shown

in line 7.

If VM is being executed for secure location verification, then the prover claims a location $\hat{p}$ before the start of the protocol. In secure localization, the verifiers have no knowledge of the prover's location at the beginning of protocol execution. In this case, VM computes the the minimum mean square estimate (MMSE) $\hat{p}$, which minimizes the following formula:

$$\sum_{v \in V} (\nabla_v - \hat{\Delta})^2 \tag{3.4}$$

using a subset N of all the verifiers V in the transmission range of the prover, as shown in lines 12-16.

VM enforces the point-in-triangle requirement added by Capkun et al. [6] to block distance fraud. The point-in-triangle requirement leverages a basic geometrical property of a triangle. In particular, If two points, say $p_1 \neq p_2$, are contained in a triangle $\langle a, b, c \rangle$, then p_1 must be further away than p_2, from at least one vertex of the triangle. This is illustrated in Fig. 3.4 in the context of multilateration. Suppose a malicious prover claims the location $\hat{p}$, but is truly located at p. Then, it must be further away than $\hat{p}$, from at least one verifier in $\{v, w, z\}$, which in this case, is verifier w. Hence, its response time to w will be greater than the expected response time, and it will be late in responding to w. The prover's cheating will be caught by w because $D(w, p) = D(w, \hat{p}) + W'$ and responding in $\Delta_w = \hat{\Delta} - 2 \cdot W'$ is impossible. However, if the point-in-triangle condition is not imposed, p could easily claim a false location outside of triangle $\langle v, w, z \rangle$. For example, p can claim the location $\hat{q}$ even though $D(p, \hat{q}) > D(p, \hat{p})$, because p is *at least* as close as $\hat{q}$ to every verifier.

In lines 17-22, VM must find at least one verification triangle for $\hat{p}$. Since VM

chooses the participating verifiers before finding the MMSE estimate $\hat{p}$, it may now discover that $\hat{p}$ happens to fall outside of all existing verification triangles. In this case, VM must add at least one new verifier to N such that it can find a verification triangle for $\hat{p}$. If VM cannot add any more verifiers, then it aborts, and $\hat{p}$ is not accepted as the prover's true location. Lines 24-30 show how the protocol proceeds if it cannot find a valid verification triangle for $\hat{p}$.

Algorithm 1 Verifiable Multilateration

$\mathcal{R} \Leftarrow Everywhere$ {Solution region, $\mathcal{R}$, is unbounded}
$V \Leftarrow$ Set of all verifiers in prover's transmission range
$\mathcal{T} \Leftarrow \emptyset$ {Set of verification triangles}
{Step 1: Verifiers measure running times and perceived response delays}
for all $v \in V$ **do** {Execute challenge-response echo}
  $T_p(v, v) \Leftarrow \ldots$ {v measures running time of echo}[Use Eq.(3.1)]
  $\nabla_v \Leftarrow \ldots$ [Always use Eq.(3.3)]
  {Step 2: Form circular constraint on prover's location}
  $\mathcal{C}_v \Leftarrow \ldots$ [Use Eq.(3.2)]
end for
$N \subseteq V$ {Choose a subset of N verifiers from set V}
for all $v \in N$ **do**
  {Step 3: Estimate location, if none was claimed}
  if $\nexists\, \hat{p}$ **then**
    $\hat{p} \Leftarrow MMSE(.)$ {Minimizes expression (3.4)}
  end if
  {Step 4: Find at least one verification triangle}
  for all $\langle v, w, z \rangle$ **do**
    if $\hat{p} \in \langle v, w, z \rangle$ **then** $\{\mathcal{T} \Leftarrow \mathcal{T} \cup \langle v, w, z \rangle\}$
      {Include $\langle x, y, z \rangle$ in the set of verification triangles}
    end if
  end for
end for
if $\mathcal{T} = \emptyset$ **then**
  if $N = V$ **then**
    return($\mathcal{R}$){Abort verification, $\mathcal{R}$ is still unbounded}
  else
    Expand N and repeat from Step 3
  end if
end if
{Step 5: Accept constraints that pass δ-test}
for all $\langle v, w, z \rangle \in \mathcal{T}$ **do**
  if $|\nabla_x - \hat{\Delta}| \leq \delta\ \forall x \in \langle v, w, z \rangle$ **then** {Pass δ-test}
    {p is within intersection of these circles}
    $\mathcal{R} \Leftarrow \mathcal{R} \cap \mathcal{C}_v \cap \mathcal{C}_w \cap \mathcal{C}_z$
  end if
end for
if δ-test failed **then** {$\hat{p}$ is rejected}
  return($\mathcal{R}$){$\mathcal{R}$ is still unbounded}
end if
return($\mathcal{R}$)

The verifiers do not accept $\hat{p}$ unless it is within at least one triangle formed by three verifiers. The triangle $\langle v, w, z \rangle$ in Fig. 3.4 is an example of a valid verification triangle for $\hat{p}$.

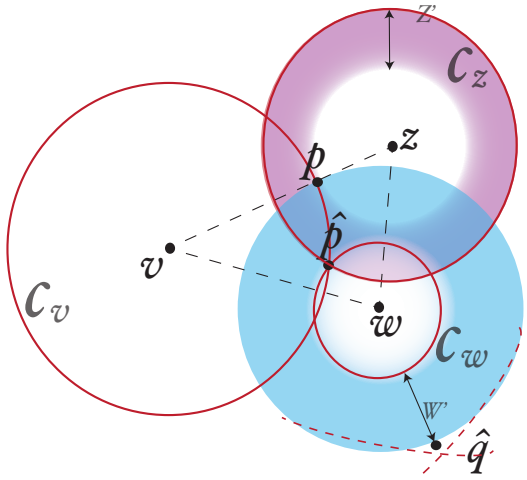


Figure 3.4: Point in Triangle Test

Lines 31-40 show how VM applies the δ-test in conjunction with the point-in-triangle test. To pass the δ-test, the perceived response time for each verifier should not differ from the expected response time by a value greater than δ. If the prover passes the δ-test for any verification triangle, the solution region $\mathcal{R}$ is immediately reduced to a small region around $\hat{p}$ in line 35 of the algorithm. If the prover fails the δ-test, then $\hat{p}$ is rejected.

Allowing for a tolerance of δ in the measured response time, ensures that the system does not flag a false negative or a false positive due to unavoidable measurement errors. δ is a system parameter whose value can be adjusted according to the target application. The point-in-triangle test in conjunction with the δ test provides effective defense from distance fraud launched by a single prover. These tests also impose a geometric constraint on the possible location of the prover – it must be *within* the convex hull formed by three or more

verifiers.

3.1.2 Hyperbolic Multilateration

Hyperbolic Multilateration (HM) introduced the concept of using *multiple* verifiers to monitor a *single* response sent by the prover. In general, it is assumed that the clocks of the verifiers are synchronized. As described in section 2.1.3, three or more verifiers record the time of arrival of the same response transmission from the prover. Although they do not know the exact time when the response was transmitted, they assume that it is same for all the verifiers. The verifiers can then pair-wise combine their measurements to form hyperbolic constraints on the location of the prover, according to Eq.(2.4).

Algorithm 2 Hyperbolic Multilateration: Localization with Witnesses

1: $\mathcal{R} \Leftarrow Everywhere$ {Solution region, $\mathcal{R}$, is unbounded}
2: **while** $\mathcal{R}$ is unbounded **do** {Execute one challenge-response round}
3: $\bar{v}_r \Leftarrow RandomPermutation(V)$ {Select lead verifier}
4: {Step 1: Verifiers measure message propagation times}
5: $T_p(\bar{v}_r, \bar{v}_r) \Leftarrow \ldots$ {$\bar{v}_r$ measures running time of echo}[Use Eq.(3.1)]
6: **for all** $w \in V \backslash \{\bar{v}_r\}$ **do**
7: $A_w(\bar{v}_r, p) \Leftarrow \ldots$ {Inter-arrival time}
8: $T_p(\bar{v}_r, w) \Leftarrow A_w(\bar{v}_r, p) + D(\bar{v}_r, w)$ {Running time of relay}[Use Eq.(3.5)]
9: **end for**
10: {Step 2: Form hyperbolic constraints}
11: **for all** $\{u, w\} \in V$ **do** {Pairwise combination of measurements}
12: $\mathcal{H}_{uw} \Leftarrow \ldots$ {Equation of hyperbola}[Use Eq.(3.6)]
13: **end for**
14: {Step 3: Constrain the prover to intersection of two or more hyperbolas}
15: **for all** $\{u, w, z\} \in V$ **do**
16: {p is at the intersection of two or more hyperbolas}
17: $\mathcal{R} \Leftarrow \mathcal{R} \cap \mathcal{H}_{uw} \cap \mathcal{H}_{wz} \cap \ldots$
18: **end for**
19: **end while**
20: **return**($\mathcal{R}$)

Localization with Witnesses [54] is a hyperbolic multilateration protocol that allows the verifiers to obtain the same hyperbolic constraints *without synchronizing their clocks in*

advance. The steps for the execution of this protocol are shown in Algorithm 2. In line 3, a single verifier (referred to as the lead verifier) is chosen to send the challenge to the prover. If any subsequent rounds are executed, the selection of the lead verifier in those rounds is also random, without replacement. In lines 4-9 the lead verifier executes a challenge-response echo, the other verifiers act as passive observers (witnesses). They silently observe the challenge from the lead verifier as well as the response from the prover. The lead verifier measures the running time of the challenge-response echo using Eq.(3.1). At the same time all witnesses measure running times of the challenge-response relays that they observe, as shown in Fig.2.5(a). For example, witness w computes the running time of the relay as:

$$\begin{aligned} T_p(v,w) &= D(v,p) + \Delta_{\{v,w\}} + D(p,w) \\ &= A_w(v,p) + D(v,w) \end{aligned} \tag{3.5}$$

where $A_w(v,p)$ is the interval between the arrival of the challenge and the arrival of the response at witness w. It is important to note that in hyperbolic multilateration, all the verifiers can collect information about the prover's location in *each* round of the protocol. In comparison, in a SISO protocol like VM, where only a single verifier can collect information about the prover's location in every round of the protocol. Since the verifiers executing hyperbolic multilateration (and in general any SIMO protocol) collect much more information about the prover's location in every round of challenge-response, they "harvest information" much more efficiently than verifiers participating in a SISO protocol. Efficient information harvesting allows SIMO protocols to complete the localization process over fewer challenge-response rounds than SISO protocols [44, 46]. In the best case, a SIMO protocol requires

only a single challenge-response round to localize the prover.

In lines 10-13, the verifiers compute the pairwise difference in their measurements:

$$\mathcal{H}_{vw} = \{p : D(v,p) - D(w,p) = T_p(v,v) - T_p(v,w)\} \tag{3.6}$$

where $\mathcal{H}_{vw}$ is a hyperbola with the foci coincident with v and w. If the prover is honest, and sends its response to all the verifiers over a *single* transmission, the response delay of the prover $\Delta_{\{v,w\}} = \Delta_v$, does not figure in the equation of the hyperbola. Therefore, the hyperbolic constraints formed are not affected whether or not the value of $\Delta_{\{v,w\}}$ matches the expected response delay. This feature secures hyperbolic multilateration from distance fraud launched by a single resource-constrained prover.

This advantage however, turns into a disadvantage if p is replaced by a resourceful prover p^*. p^* possesses multiple radios and directional antennas, and can therefore delay its response transmission to different verifiers by different amounts. Since hyperbolic multilateration does not impose any bounds on the response time (unlike the δ-test in VM), p^* can successfully claim a false location. In this case, a hyperbolic multilateration protocol completely fails to detect distance fraud launched by a resourceful prover.

HM also does not impose any restrictions regarding the proximity of the verifiers to the computed location of the prover. Since two or more hyperbolas may intersect at two different points in space, a malicious prover may be at the intersection that is outside of the convex hull formed by the verifiers, and successfully claim to be at the alternate intersection inside the convex hull. This is illustrated in Fig. 3.5. This is another scenario when HM fails to detect distance fraud.

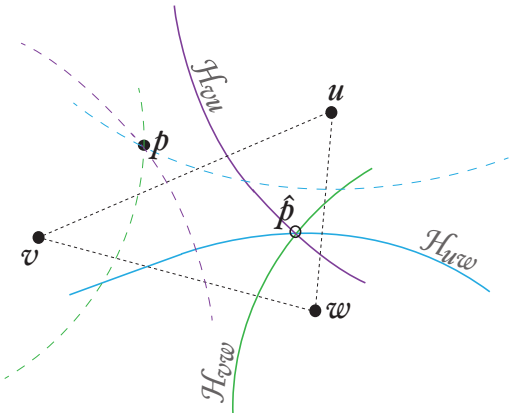


Figure 3.5: A malicious prover can successfully claim to be at a location inside of the convex hull formed by the verifiers, while it is actually at outside of the convex hull. When the two intersection points of all the hyperbolas are far from each other, the verifiers accept a location which is no where near the claimed location.

3.2 Elliptical Multilateration

3.2.1 Motivation

In the previous sections, we highlighted the features that secure SISO VM and SIMO HM from distance fraud attacks. We also discussed the message exchange structure of both secure localization protocols. We observed the following:

VM is a time-of-arrival (ToA) protocol that exhibits strong security properties against distance fraud attacks. By administering the point-in-triangle and the δ-tests, VM ensures that the prover is located within the convex hull formed by the verifiers. In VM, a resourceful prover p^* has no advantage over a resource constrained prover p, because in either case, the response time is matched to the known response time with an allowable error of δ. Deviation from the expected response time by more than δ causes the prover to fail the test, and the verifiers do not accept $\hat{p}$ as its true location.

Despite these advantages, VM has some drawbacks. Firstly, it the value of δ is not

set properly, a false negative may cause the verifiers to reject an honest prover's location claim. Also, only a single verifier can collect information about the prover's location in each round of protocol execution. Due to this SISO VM requires a greater number of message exchanges than SIMO HM to complete the localization process.

HM is more efficient in harvesting information in each challenge-response round because it is a time-difference-of-arrival (TDoA) SIMO protocol. Recall that SISO VM cannot complete the localization process if the prover deviates from the expected response time by more than δ. A deviation from the expected response time can occur even if the prover is not malicious: the hardware may not support the required precision, or the measurements errors may be larger than value that δ is set to. If such is the case, VM either flags a false negative or aborts the localization process. In comparison, HM is agnostic to changes in the response time of the prover. Unlike VM, HM always terminates by localizing the prover, irrespective of exact value of p's response delay.

This advantage turns into a disadvantage in certain situations. Since the exact value of Δ does not affect the result of localization, a resourceful prover p^* may change it arbitrarily for different verifiers and cause them to compute incorrect hyperbolic constraints. Even if the prover does not manipulate the response delay, HM may accept an incorrect location since multiple hyperbolas can intersect at more than one point.

Thus, we find that both VM and HM have some advantages and some disadvantages. This begs the question: *"Is it possible to design a secure time-based localization protocol that incorporates all the advantages of VM and HM?"*

We propose a new time-of-arrival (ToA) SIMO multilateration protocol for secure time-based localization/location verification. In our protocol, the verifiers use their mea-

surements to form elliptical constraints on the prover's location, hence we call it ***"Elliptical Multilateration (EM)"***. Our protocol builds on the security features of VM to provide effective defense against distance fraud. In addition, it uses the SIMO message structure of HM for efficient information harvesting in every challenge-response round. Therefore, our protocol retains the best features of both SISO VM and SIMO HM.

3.2.2 Protocol Description

The steps of execution of EM are shown in Algorithm 3. In each round of the protocol, one verifier is randomly selected as the "lead verifier" to send a *single* challenge to the prover. This is shown in line 5 of the algorithm. For each additional round executed, a different lead verifier is picked at random without replacement. For added security, the lead verifier uses a random MAC address instead its true address while sending the challenge.

In lines 6-13, the verifiers measure the running time of the challenge-response relay as well as the perceived response delay of the prover. The lead verifier executes a challenge-response echo with the prover along the two-hop path $v \rightarrow p \rightarrow p$. The running time of the challenge-response echo as measured by $\bar{v}$ and the response delay it perceives are given by Eq.(3.1) and Eq.(3.3) respectively.

At the same time, each passive verifier, say w, observes the challenge-response relay along the path $v \rightarrow p \rightarrow w$. If the clocks of the verifiers are synchronized, the witness only needs to record the time of arrival of the response. To obtain the exact time at which the verifier sent the challenge, the witnesses can query the verifier. By subtracting the time at which the challenge was sent from the time at which the response arrived, a witness can compute the running time $T_p(v, w)$ of the challenge-response relay. A witness can also

estimate $T_p(v, w)$ when the verifier clocks are not synchronized. In this case, the witness uses the synchronization-free technique from the "Localization with Witnesses" protocol. The witness records the time of arrival of both the challenge and the response, and uses Eq.(3.5) to estimate $T_p(v, w)$. In addition it also records the perceived response delay of the prover given by Eq.(3.7)

$$\nabla_{vw} = T_P(v, w) - [D(v, \hat{p}) + D(\hat{p}, w)] \tag{3.7}$$

In line 14, EM selects a subset N from the set V of all the verifiers who are in the transmission range of the prover (and the lead verifier, if the clocks are not synchronized), such that N contains the lead verifier chosen for that round. In lines 15-17, each verifier $w \in N$ forms an elliptical constraint $\mathcal{E}_{vw}$ on the prover's location using Eq.(3.8), such that the foci of the ellipse coincide with the locations of the lead verifier and the witness itself.

$$\mathcal{E}_{vw} = \{p : D(v, p) + D(p, w) \leq T_p(v, w) - \hat{\Delta}\} \tag{3.8}$$

If EM is being executed to solve a location verification problem, the prover claims a location $\hat{p}$ at the beginning of the protocol. If EM is being executed for localization, then the verifiers have no initial input from the prover about its location. In this case, EM computes the minimum mean square estimate (MMSE) $\hat{p}$ which minimizes

$$\sum_{v \in V} \left((\nabla_v - \hat{\Delta})^2 + \sum_{w \in V, w \neq v} (\nabla_{vw} - \hat{\Delta})^2 \right) \tag{3.9}$$

Algorithm 3 Eliptical Multilateration

1: $\mathcal{R} \Leftarrow Everywhere$, {Solution region, $\mathcal{R}$, is unbounded}
2: $V \Leftarrow$ Set of all verifiers in prover's transmission range
3: $\mathcal{T} \Leftarrow \emptyset, r \Leftarrow 0$ {Set of verification triangles, r increments when prover passes δ-test}
4: **while** $r \leq 1$ **do** {Execute one challenge-response round}
5: $\bar{v}_r \Leftarrow RandomPermutation(V)$
6: {Step 1: Verifiers measure running time and perceived response delay in this round}
7: $T_p(\bar{v}_r, \bar{v}_r) \Leftarrow \ldots$ {$\bar{v}_r$ measures running time of echo}
8: $\nabla_{\bar{v}_r} \Leftarrow \ldots$ {$\bar{v_r}$ uses Eq.(3.3)}
9: **for all** $w \in V \backslash \{\bar{v}_r\}$ **do**
10: $A_w(\bar{v}_r, p) \Leftarrow \ldots$ {Passive inter-arrival time}
11: $T_p(\bar{v}_r, w) \Leftarrow A_w(\bar{v}_r, p) + D(\bar{v}_r, w)$ {running time of relay}
12: $\nabla_{\bar{v}_r w} \Leftarrow \ldots$ {Witnesses use Eq.(3.7)}
13: **end for**
14: $N \subseteq V$ {Choose a subset of N verifiers from set V, such that $\bar{v}_r \in N$}
15: **for all** $v \subseteq N$ **do**
16: {Step 2: Form elliptical constraints}
17: $\mathcal{E}_{\bar{v}_r w} \Leftarrow \ldots$ {Elliptical constraint formed by $\bar{v}_r$ and w}[Use Eq.(3.8)]
18: {Step 3: Estimate location, if none was claimed}
19: **if** $\nexists\, \hat{p}$ **then**
20: $\hat{p} \Leftarrow MMSE(\cdot)$ {Minimizes expression (3.9)}
21: **end if**
22: {Step 4: Find at least one verification triangle}
23: **for all** $\langle u, w, z \rangle \in N$ **do**
24: **if** $\hat{p} \in \langle u, w, z \rangle$ **then** $\{\mathcal{T} \Leftarrow \mathcal{T} \cup \langle u, w, z \rangle\}$
25: {Include $\langle u, y, z \rangle$ in the set of verification triangles}
26: **end if**
27: **end for**
28: **end for**
29: **if** $\mathcal{T} = \emptyset$ **then**
30: **if** $N = V$ **then**
31: Repeat from line 4 {Pick a new lead verifier and execute another round}
32: **else**
33: Expand N and repeat from Step 2
34: **end if**
35: **end if**
36: {Step 5: Accept constraints that pass δ-test}
37: **for all** $\langle u, w, z \rangle \in \mathcal{T}$ **do**
38: **if** $|\nabla_{xy} - \hat{\Delta}| \leq \delta\ \forall (x, y) \in \langle u, w, z \rangle$ **then** {Pass δ-test}
39: {p is within intersection of these ellipses}
40: $\mathcal{R} \Leftarrow \mathcal{R} \cap \mathcal{E}_{\bar{v}_r u} \cap \mathcal{E}_{\bar{v}_r w} \cap \mathcal{E}_{\bar{v}_r z}, r \Leftarrow r + 1$
41: **end if**
42: **end for**
43: **if** δ-test failed **then**
44: **return**$(\mathcal{R})$ {Abort verification, $\mathcal{R}$ is still unbounded}
45: **end if**
46: **end while**
47: **return**$(\mathcal{R})$

Although this feature of EM is similar to VM, there is a difference in the MMSE formulation. In the expression (3.9), there is an extra inner sum, which represents the additional information gathered by passive witnesses in each round of the protocol. Therefore EM incorporates efficient information harvesting similar to HM. While N verifiers contribute N terms to the MMSE formulation in VM, the same number of verifiers contribute N^2 terms to the MMSE formulation in EM. If the value of N and the resourcefulness of the prover in terms of hardware is same for both protocols, EM can complete the localization process in fewer rounds than VM [44]. The accuracy of localization is also better. This will be discussed in greater detail in chapter 4.

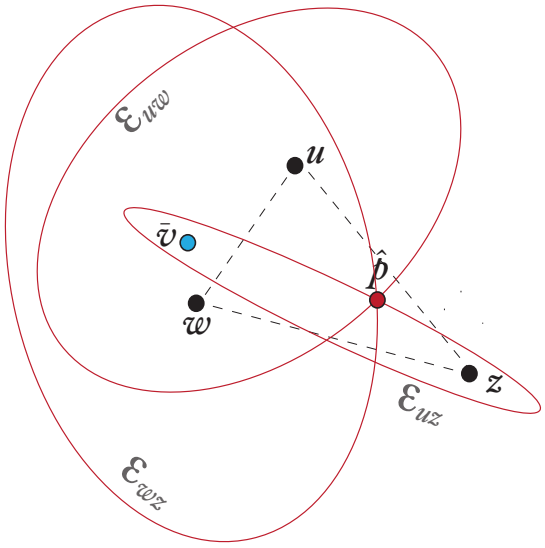


Figure 3.6: Three verifiers belonging to a valid verification triangle can localize the prover to the intersection of the elliptical constraints formed by them.

The rest of the steps of EM are similar to VM. The only difference is In lines 22-27, EM searches for at least one triangle $\angle u, w, z\rangle$ formed from verifiers in N, with which $\hat{p}$ satisfies the point-in-triangle test. If EM is unable to do so, N is expanded until

it finds a valid verification triangle, or $N = V$. If EM is successful in finding a verification triangle, it proceeds to administer the δ-test. If not, and $N = V$, then EM discards the measurements from the current round, picks a new lead verifier, and executes another challenge-response round. Later in section 3.2.3, we will prove that EM requires at most two rounds of challenge-response, given that it could find at least one verification triangle in each round.

In lines 36-45, EM administers the δ-test similar to VM. If the prover satisfies the test with all three verifiers from a valid verification triangle, then the solution region $\mathcal{R}$ immediately reduces to the intersection of the elliptical constraints formed by the three verifiers in triangle $\langle u, w, z \rangle$. Algorithm 3 shows that EM must execute at least two rounds in which the prover passes the δ-test with a valid verification triangle. The second round is needed to secure EM against distance fraud launched by a resourceful prover p^*. We elaborate on this in the next section, where we analyze the security features of EM.

3.2.3 Security Analysis

In the previous section, we showed how the EM uses the point-in-triangle test in conjunction with the δ-test to provide security against distance fraud. Recall that the point-in-triangle test ensures that a single response transmission from the prover will be late for at least one verifier in the verification triangle. When the point-in-triangle requirement is satisfied, a prover not located at $\hat{p}$ fails the δ-test with at least one verifier, and $\hat{p}$ is rejected. Therefore, if EM can find a valid verification triangle $\langle u, w, z \rangle$ in some challenge-response, and the prover passes the δ-test with all three verifiers in the verification triangle, then the prover can be localized in a single challenge-response round. The prover's location is

determined to be the intersection of the three elliptical constraints formed by the verifiers in $\langle u, w, z \rangle$, which is a small region around $\hat{p}$ having an $O(\delta^2)$ area. However, *a single challenge-response round suffices to securely localize the prover only if it sends its response over a single transmission to all the verifiers.*

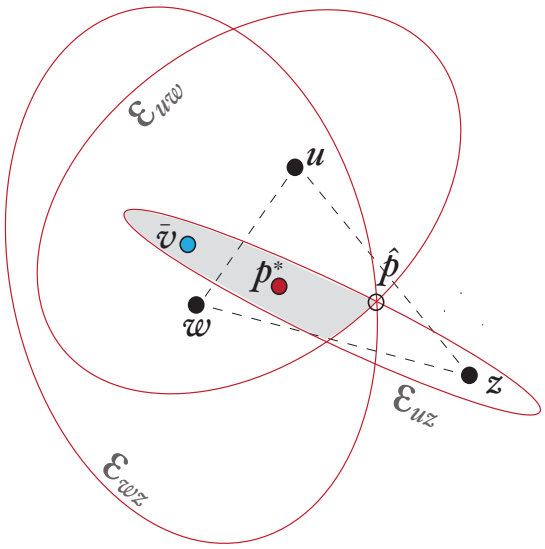


Figure 3.7: A resourceful prover located within the elliptical constraint formed by a verifier can delay its response to that verifier and pass the δ-test with it. Since the prover p^* in this figure is within the elliptical constraints formed by all three verifiers of a verification triangle $\langle u, w, z \rangle$, it can enlarge its response time to pass the δ-test with all of them.

Let us consider how the situation changes if the resource constrained prover p is replaced by a resourceful prover p^*. Since p^* has multiple radios and directional antennas, it may time-shift its responses to different verifiers by different amounts. Although p^* cannot lower its response time below the minimum response time $\hat{D}elta$, it can enlarge it by any amount to match the expected response time. Consider Fig. 3.7. Let p^* be present within or on the boundary of the shaded region shown in the figure. For any verifier $x \in \langle u, w, z \rangle$, $D(\bar{v}, p^*) + D(p^*, x) \leq D(\bar{v}, \hat{p}) + D(\hat{p}, x)$. If p^* sends separate responses to the verifiers

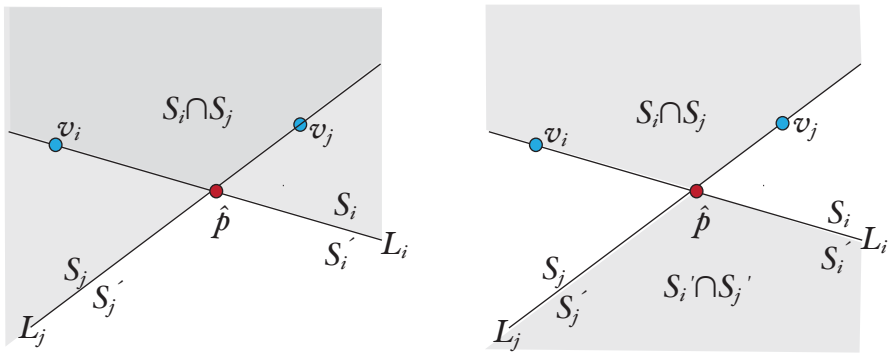
in $\langle u, w, z\rangle$ in a single round, it can time-shift them appropriately to match the expected response time for each verifier, therefore, pass the δ-test with all three verifiers. In this case, a single challenge-response round is not sufficient to securely determine/verify the prover's location. Distance fraud launched by a resourceful prover is a serious threat not only to our EM protocol, but to all time-based localization protocols.

We prove that our EM protocol provides effective defense against distance fraud even when the prover is capable of directional transmission, so that different verifiers receive different transmissions of the response, time-shifted by different amounts. The conditions for securing EM against distance fraud are summarized in Theorem 1:

Theorem 1 *If EM is executed with an arbitrary number of verifiers in V, then EM can detect distance fraud if (i) EM can find at least one verification triangle $\langle x_i, y_i, z_i\rangle$ in the ith round and at least one verification triangle $\langle x_j, y_j, z_j\rangle$ in the jth round (ii) the prover satisfied the δ-test with $\langle x_i, y_i, z_i\rangle$ in the ith round and with $\langle x_j, y_j, z_j\rangle$ in the jth round (iii) at least one verifier in $\langle x_i, y_i, z_i\rangle$ forms a verification triangle with v_i and v_j and at least one verifier in $\langle x_j, y_j, z_j\rangle$ forms a verification triangle with v_i and v_j, where v_i and v_j are the lead verifiers in the ith and jth round respectively.*

Proof: Let $\mathcal{R}$ be the plane verifiers $\{v, w, u...\} \in V$, prover p and the claimed location of the prover $\hat{p}$. Consider Fig. 3.8(a). Let v_i be the lead verifier in the ith challenge-response round, and v_j be the lead verifier in the jth challenge-response round. We denote the straight line passing through v_i and $\hat{p}$ as L_i. Similarly we denote the straight line passing through v_j and $\hat{p}$ as L_j. L_i partitions $\mathcal{R}$ into two half planes S_i and S_i' such that S_i contains v_j. L_j partitions $\mathcal{R}$ into two half planes S_j and S_j' such that S_j contains v_i. The intersection of the half planes S_i and S_j is the region $S_i \cap S_j$ within $\angle v_i \hat{p} v_j$, as shown

in Fig. 3.8(b). The intersection of S_i' and S_j' is the region $S_i' \cap S_j'$ within the angle opposite to $\angle v_i \hat{p} v_j$, also shown in Fig. 3.8(b).



(a) (b)

Figure 3.8: (a)Line L_i partitions $\mathcal{R}$ into half planes S_i and S_i'. Line L_j partitions $\mathcal{R}$ into half planes S_j and S_j'. (b) The intersection $S_i \cap S_j$ is within the non-reflex angle $\angle v_i \hat{p} v_j$. The intersection of the complementary half planes $S_i' \cap S_j'$ is within the opposite angle.

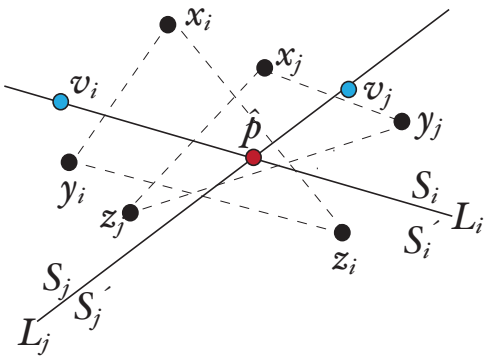


Figure 3.9: The lead verifiers v_i and v_j shown along with the verification triangles $\langle x_i, y_i, z_i \rangle$ and $\langle x_j, y_j, z_j \rangle$ for the respective rounds.

Suppose $\langle x_i, y_i, z_i \rangle$ is a verification triangle with which $\hat{p}$ satisfies the δ-test in the first round. By definition, $\hat{p}$ must with strictly within $\langle x_i, y_i, z_i \rangle$, therefore, at least one verifier in this triangle must be on each side of the line L_i. Let verifier x_i be in S_i and verifier y_i be in S_i'. The third verifier z_i can be either in S_i or in S_i'. However, of the two

verifiers in the half plane containing z_i, v_i's distance from z_i is greater than its distance from the other verifier. This means if the half-plane S_i contains the two verifiers x_i and z_i, then x_i is closer to v_i. Similarly, if half plane S_i' contains the two verifiers in y_i and z_i, then y_i is closer to v_j. This is illustrated in Fig. 3.9. The same figure also shows the lead verifier v_j and verification triangle $\langle x_j, y_j, z_j \rangle$ with which the prover passes the δ-test in the second round. In this case, the verifier that is closest to v_j in the half plane S_j, is denoted as x_j, and the verifier that is closest to v_j in the half plane S_j' is denoted as y_j. z_j may be either in S_j or in S_j'.

Let $\mathcal{R}_i$ be the region to which the verifiers in $\langle x_i, y_i, z_i \rangle$ constrain the location of the prover in the ith round. From Lemma 1,we know that $T_{v_i y_i}$ – the tangent to the elliptical constraint $\mathcal{E}_{v_i y_i}$, is the one of the boundaries for $\mathcal{R}_i \cap S_i$, the other being the line L_i. This is shown in Fig. 3.10(a) Similarly, if the verifiers in $\langle x_j, y_j, z_j \rangle$ constrain the prover to the region R_j after a second round, then $T_{v_j y_j}$ – the tangent to the elliptical constraint $\mathcal{E}_{v_j y_j}$ is one of the boundaries for $\mathcal{R}_j \cap S_j$, the other being line L_j. This is shown in Fig. 3.10(b). The intersection $R_i \cap R_j$ is bounded on either side by $T_{v_i y_i}$ and $T_{v_j y_j}$ is shown in Fig. 3.10(c).

If $R_i \cap R_j \neq \phi$, i.e., an intersection of the constraints formed across two rounds exists, then a resourceful prover located in $R_i \cap R_j$ can successfully cheat in both rounds to claim the location $\hat{p}$. Therefore, to ensure that the prover's cheating is caught, we must ensure that $R_i \cap R_j = \phi$.

Next, we find the condition that ensures $R_i \cap R_j = \phi$. Consider Fig. 3.10(d) Let the non-reflex angle between v_i and v_j be m°, and line B be its bisector. If the angle between line L_i and B is denoted as $\angle v_i \hat{p} b$, and the angle between line L_j and B is denoted

as $\angle v_j\hat{p}b$, then $\angle v_i\hat{p}b = \angle v_j\hat{p}b = (m/2)^\circ$. Let l_i and l_j be two points on lines L_i and L_j respectively as shown in the figure. Since $\angle v_i\hat{p}v_j = m^\circ$, $\angle v_i\hat{p}l_j = \angle v_j\hat{p}l_i = (180 - m)^\circ$.

Next, consider Fig. 3.10(e), which shows y_i and y_j. Let $\angle v_i\hat{p}y_i = n_i^\circ$ and $\angle v_j\hat{p}y_j = n_j^\circ$. $T_{v_iy_i}$ denotes the tangent to the elliptical constraint $\mathcal{E}_{v_iy_i}$. $T_{v_jy_j}$ denotes the tangent to the elliptical constraint $\mathcal{E}_{v_jy_j}$. If we denote the angle made by line L_i with $T_{v_iy_i}$ as $\angle v_i\hat{p}t_i$, and the angle made by line L_j with $T_{v_jy_j}$ as $\angle v_j\hat{p}t_j$, then from the tangent properties of ellipse we have $\angle v_i\hat{p}t_i = \frac{(180-n_i)}{2}^\circ$ and $\angle v_j\hat{p}t_j = \frac{180-n_j}{2}^\circ$.

Given the angles in Fig. 3.10(d) and Fig. 3.10(e), the following two conditions must be true to ensure that $R_i \cap R_j = \phi$:

(i)$\angle v_i\hat{p}t_i \leq \angle v_i\hat{p}b$. Plugging in the values of the angles, we have

$$\begin{aligned} \frac{180 - n_i}{2} &\leq \frac{m}{2} \\ \Rightarrow 180 - m &\leq n_i \end{aligned}$$

which means that $\angle v_i\hat{p}y_i$ must be grater than $\angle v_i\hat{p}l_j$. Thus y_i should be in the region $S_i' \cap S_j'$. Since any point in region $S_i' \cap S_j'$ forms a verification triangle with the two lead verifiers, y_i must also form a verification triangle with v_i and v_j.

(ii) $\angle v_j\hat{p}t_j \leq \angle v_j\hat{p}b$. Plugging in the values of the angles, we have

$$\begin{aligned} \frac{180 - n_j}{2} &\leq \frac{m}{2} \\ \Rightarrow 180 - m &\leq n_j \end{aligned}$$

which means that $\angle v_j \hat{p} y_j$ must be grater than $\angle v_j \hat{p} l_i$, thus y_j also should be in the region $S'_i \cap S'_j$. By reasoning similar to that in (i) above, y_j must also form a verification triangle with the lead verifiers v_i and v_j. QED.

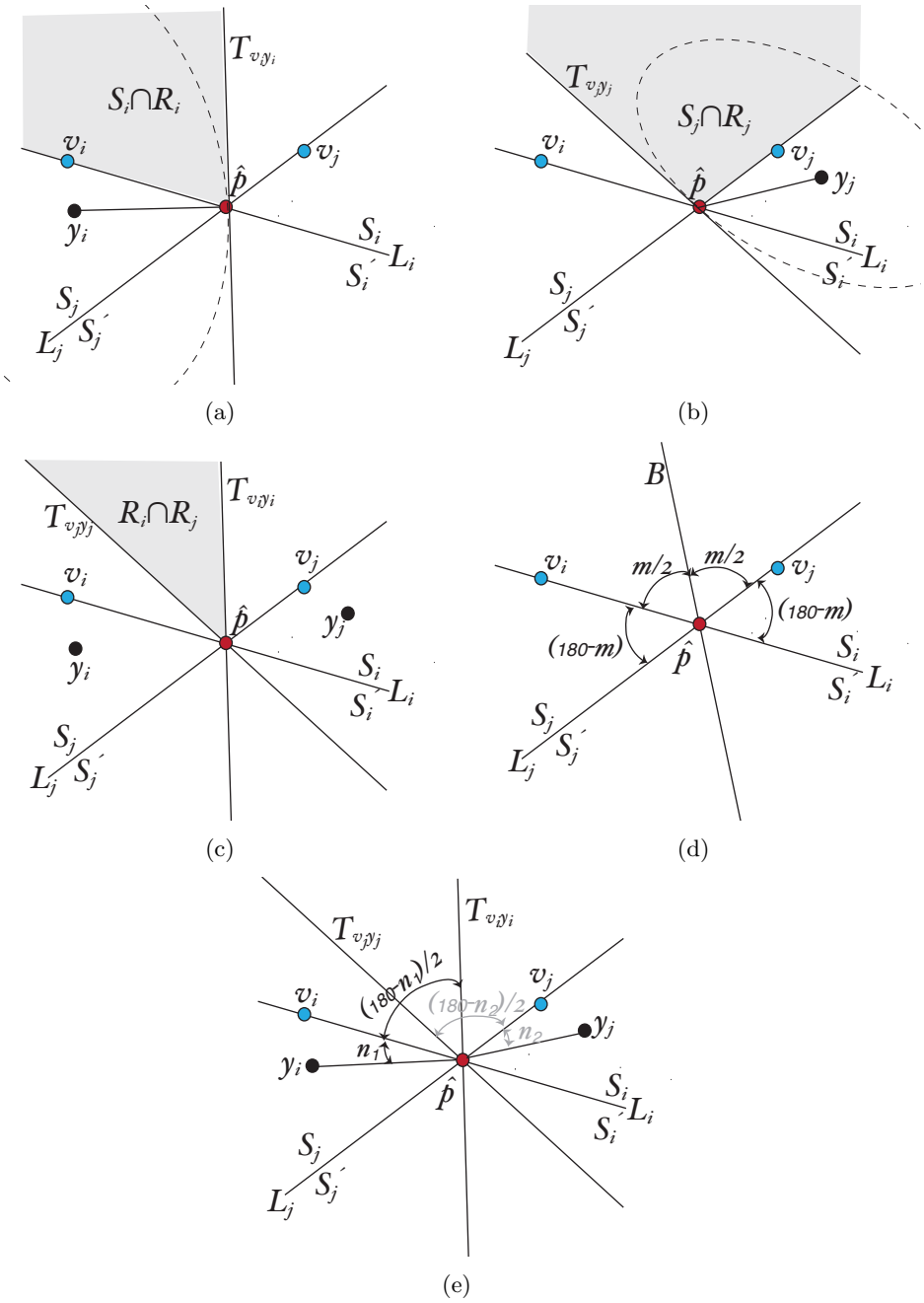


Figure 3.10: (a)$T_{v_i y_i}$ bounds the region $S_i \cap R_i$, where R_i is the region to which the prover is constrained in the ith round (b) $T_{v_j y_j}$ bounds the region $S_j \cap R_j$, where R_j is the region to which the prover is constrained in the jth round (c) Intersection of the regions to which the prover is constrained in the ith and jth rounds (d) Angles formed by the lines B, L_i and L_j (e) Relationship between the angle $\angle v_i \hat{p} y_i$ and $\angle v_i \hat{p} t_i$

3.2.4 Comparison with Verifiable Multilateration and Hyperbolic Multilateration

We showed that the SIMO message structure of EM allows it to complete the localization process efficiently. We also showed that EM is secure against distance fraud attacks when the prover is resource-constrained or resourceful in terms of hardware. Hence our EM protocol satisfies the "correctness" criteria for secure time-based localization.

In the set-by-step explanation of EM, we showed that it combines the best properties of VM and HM. Like VM, EM incorporates the point-in-triangle test in conjunction with the δ-test. In the case of VM a resourceful prover p^* does not have any advantage over a resource-constrained prover p, because only a single verifier interacts with the prover in every challenge-response round. However EM is a SIMO protocol capable of efficient information harvesting by using multiple receivers in every round of the protocol. In SIMO protocols, p^* *does* have an advantage over a resource-constrained prover p. Under the circumstances mentioned earlier in this section, a resourceful prover might be successful in cheating the verifiers in one round. However, if EM continues execution until it completes two rounds such that the conditions in Theorem 1 are satisfied, then EM can prevent even resourceful prover p^* from launching s distance fraud attack. In theorem 1 we do not make any assumptions regarding the number of verifiers that participate in EM. If EM is executed with exactly three verifiers, then EM requires only two (*consecutive*) rounds of challenge-response to accept or disprove that the prover is indeed located at $\hat{p}$.

Due to the point-in-triangle and δ-tests, EM imposes strict geometric constraints on the possible location of the prover, similar to VM. The prover must be located within the convex hull formed by the participating verifiers. Although EM shares the SIMO message

structure of HM, it does not inhere the disadvantage that HM has in this regard. Unlike HM, EM does not accept the location of a prover outside of the convex hull formed by the verifiers. The elliptical constraints formed in each round intersect only at a single point in comparison to the two possible intersection points for hyperbolic constraints. This avoids the possibility of accepting an alternate location instead of the true location of the prover.

EM also uses features that obfuscate the specifics of the lead verifier in every round of challenge-response. Sine the lead verifier uses a random MAC address while sending the challenge, the prover cannot determine its identity. The lead verifier in each round is selected at random without replacement. Therefore, even if the prover has a map of the network, and knows the locations of the verifiers, it must determine which verifier sent the challenge, by determining the direction from which the challenge arrived. This is extremely difficult even for a resourceful prover p^*. To commit distance fraud by using different copies of the response for different verifiers, p^* must possess *directional receivers* in addition to directional transmitters. Therefore, obfuscation of the lead verifier's identity and location are powerful security features of EM. This feature is not only beneficial for our EM protocol, but can be added to existing time-based localization protocols like VM to enhance their defense against distance fraud.

3.3 Conclusions

Our most important contribution in this chapter is to introduce Elliptical Multilateration (EM) – a new time-based secure localization protocol. Unlike earlier SIMO protocols that use multiple receivers for simultaneous multilateration with the time-difference-of-arrival(TDoA) technique, EM has a more conventional time-of-arrival (ToA) formulation.

The SIMO message structure of EM allows all the the verifiers to collect information about the prover's location in each round of the protocol. Due to this feature, EM can localize the prover over fewer message exchanges than a SISO protocol like VM using the same number of verifiers.

EM also employs well-established security properties like the point-in-triangle test and the δ-test to provide effective defense against distance fraud. We formally proved that EM can detect distance fraud committed by a single resource-constrained prover by executing only a single round of challenge-response. Even when the prover is resourceful in terms of hardware and is capable of sending separate copies of the response, time-shifted by different amounts to different verifiers, EM detects cheating and rejects the claimed location $\hat{p}$. We observed that elaborate timing attacks fail when the identity and location of the lead verifier are obfuscated. We consider this feature to be very effective in securing time-based localization against distance fraud, and advocate its inclusion in other time based localization protocols like VM and HM.

Overall, we show that our EM protocol addresses all issues related to "correctness" of secure time-based localization. In subsequent chapters, our discussions focus on the issue of "proper timing resolution" when EM and other secure time-based localization protocols are implemented in 802.11-based networks.

Chapter 4

Architectural Support for Time-Based Localization with 802.11-Compatible Entities

In chapter 3, we introduced a new protocol called Elliptical Multilateration (EM) and proved that it addresses the "correctness" criteria for secure time-based localization in 802.11-based networks. Starting with this chapter, we focus on addressing the criteria of proper "timing resolution". In chapter 1, we described why we need to capture timestamps with nanosecond-level precision for meaningful localization results in 802.11-based networks. The IEEE 802.11 standard currently does not support timestamping with such precision. In this chapter, we propose simple architectural modifications that are necessary for achieving timestamping precision on the order of nanoseconds. Making these modifications enables accurate time-based localization with 802.11-compatible entities.

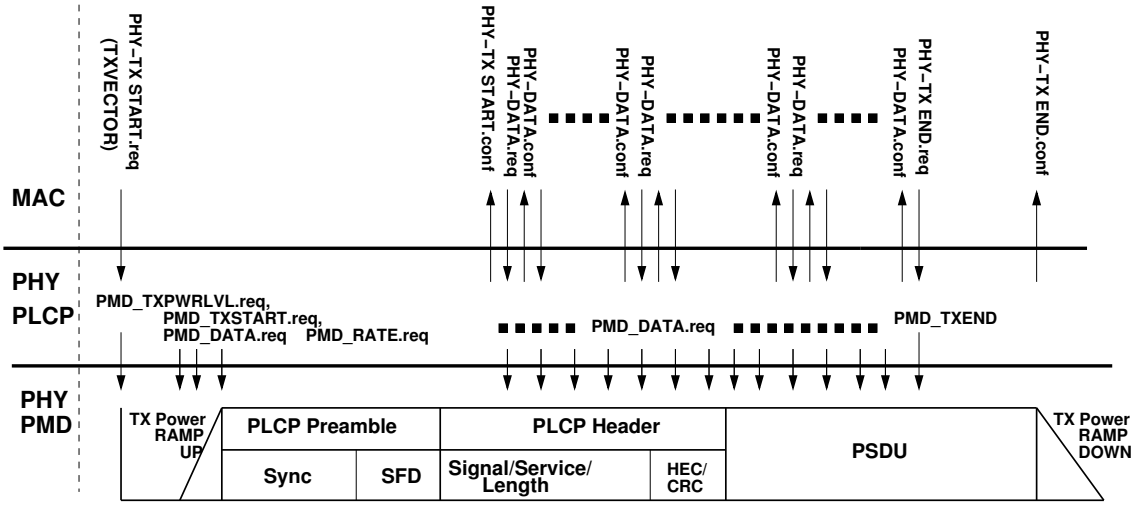
4.1 Motivation

4.1.1 Overview of Packet Transmission and Reception

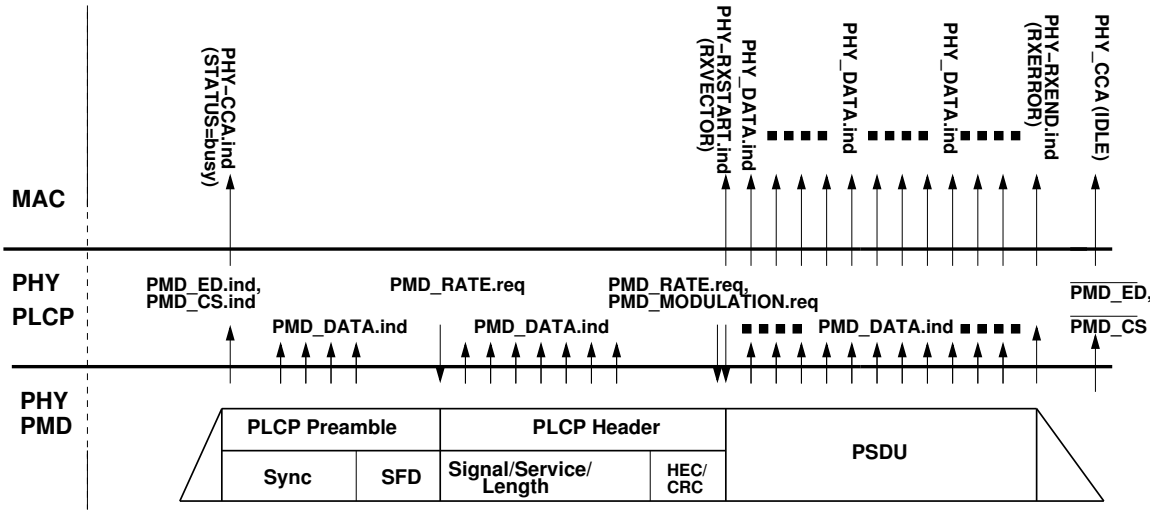
The motivation for the work presented in this chapter is that currently available support for timestamping in the 802.11 standard fails to meet the accuracy required by time-based localization protocols. To understand why this is so, we first need to understand the interactions between the PHY and the MAC layers of an 802.11-compatible entity. The 802.11 PHY consists of two sublayers [22]: the *PMD* (i.e., the actual radio transceiver) and the *PLCP* (i.e., a set of functions for controlling and/or [re-]configuring the PMD). Although these two PHY sublayers are functionally separate, the PLCP-PMD boundary in the actual hardware is somewhat vague because it was never intended to be an exposed interface.

Fig. 4.1 shows the sequence of primitives that cross the MAC-PHY interface to handle a single packet. First, let us consider the packet transmission process in 802.11-compatible entities. This is illustrated in Fig. 4.1(a). By default, the PHY is configured to receive incoming packet headers (CS/CCA state). Therefore, to initiate a packet transmission, the MAC issues the `PHY-TXSTART.request` to the PLCP, together with a parameter list including the data rate, packet length, preamble type, modulation to be used, scrambler initialization vector (if OFDM is used), and the transmit power level. Receipt of this primitive causes the PLCP to ready the PHY for this packet transmission. The PLCP then issues various primitives to the PMD to configure and then power up its transmit function. The packet Preamble and PLCP header generation are handled entirely in the PLCP sublayer: unlike Ethernet autonegotiation (which configures the PHY once at link startup), 802.11

nodes may need to reconfigure the PHY on a packet-by-packet basis to communicate with different nodes in the same Basic Service Area or even – for some modulation schemes – in mid packet. The PLCP then passes the stream of header data to the PMD for transmission.



(a)



(b)

Figure 4.1: Interactions between the two sublayers of the PHY and between the PHY and MAC layers during an 802.11 packet (a) transmission (b) reception.

Once the PMD has been configured and transmission of header data is under way, the PLCP issues the `PHY-TXSTART.confirm` primitive, telling the MAC of its readiness to

accept the outgoing packet, one octet at a time, through an exchange of `PHY-DATA.req` and `PHY-DATA.confirm` primitives. After supplying the final octet of data, the MAC issues the `PHY-TXEND.request` primitive to the PLCP. Receipt of this primitive causes the PLCP to power down the PMD's transmit function and restore it to the CS/CCA state after the entire packet has been sent, then issue a `PHY-TXEND.confirm` primitive to the MAC acknowledging its completion.

Next we consider the packet reception process in 802.11-compatible entities. Packet reception involves similar interactions between the MAC and PLCP, shown in Fig. 4.1(b). As soon as the PMD detects a signal on the medium, the PLCP notifies the MAC by issuing the `PHY-CCA.indication` primitive with `STATUS=busy`, and then waits for the PMD receive function to synchronize with the incoming data stream. Once the PLCP has received enough of the incoming PMD data stream to detect a valid SFD and decode the parameters (including its length) from the PLCP Header, it issues a `PHY-RXSTART.indication` primitive to notify the MAC that a data packet is now arriving, and, possibly, reconfigures the PMD to a new rate and modulation scheme. Subsequently, each correctly-received octet is passed to the MAC with the `PHY-DATA.indicate` primitive. When it finds the end of the packet, the PLCP notifies the MAC by issuing the `PHY-RXEND.indicate` primitive with `RXERROR=no_error`, and reconfigures the PMD back to its default CS/CCA state. Finally, when the PMD has stopped detecting a signal, the PLCP notifies the MAC by issuing the `PHY-CCA.indication` primitive with `STATUS=idle`.

It is important to note that the ***IEEE 802.11 standard does not specify the exact timing of these primitives, relative to events at the air interface.*** At the transmitter, the `PHY-DATA.confirm` primitive must be issued *before the end* of the header er-

ror check (HEC) has been transmitted, but it could be as early as the start of the Preamble. Similarly, the `PHY-DATA.confirm` must be issued *before* any of the associated data is transmitted, and the `PHY-TXEND.confirm` must be issued *after* the transmission of last bit of the packet. Conversely, at the receiver, the `PHY-RXSTART.indication` primitive must be issued *after the end* of the HEC has been received, but it could be much later as long as the PHY has enough buffer space. Similarly, the `PHY-DATA.indicate(DATA)` and `PHY-RXEND.indicate` primitives must be issued *after* the associated data has been received. Since the exact timing of these primitives is unknown during actual implementation, it is not possible to compute the exact delay between the time at which the packet is transmitted/detected at the antenna and the time at which a certain primitive is issued. Therefore ***error due to the delays incurred within the PHY cannot be corrected by subtracting a deterministic value from the timestamps captured for packet arrival and departure events***.

4.1.2 Current Timestamping Support in the 802.11 Standard

The IEEE 802.11 Standard [22] currently provides a Time Synchronization Function (TSF), through which all stations syntonize their local MAC-layer protocol timers to the "timestamps" (actually 64-bit microsecond counter values) broadcast by the Access Point in periodic *Beacon Frames*. Separate from the TSF, the IEEE 802.11 standard also includes an optional capability called `MLME-HL-SYNC` in the MAC-layer management entity (MLME), which is intended to support application-layer time synchronization protocols.

To enable the `MLME-HL-SYNC` capability, the MAC client issues the `MLME-HL-SYNC.request` primitive to the MLME, together with a target multicast MAC address; this triggers the

MLME to immediately issue the MLME-HL-SYNC.confirm primitive, together with a result code of either SUCCESS or NOT_SUPPORTED. If it is supported, the MLME starts searching for the next frame that contains the target multicast MAC address as its destination; when found, the MLME waits until the end of the frame and then issues the MLME-HL-SYNC.indication primitive to the MAC client, together with the source MAC address and sequence number from the triggering frame. Notice that the MLME-HL-SYNC capability handles both transmitted and received frames, in which case the MLME-HL-SYNC.indication primitive will coincide with either the PHY_TXEND.confirm or the PHY_RXEND.indication primitive, respectively.

Recall that the MAC becomes aware of a packet transmission or reception only when it receives the PHY_TXEND.confirm or the PHY_RXEND.indication primitive. However, in the previous subsection, we explained that the exact interval between the departure/arrival of a packet at the air interface(antenna), and the issuance of either of these two primitives cannot be determined. Therefore, any timestamp captured in the MAC always includes the delay incurred in PHY processes and in signaling between the PHY and the MAC, as additive error. We also explained that it is not possible to correct for this error term by subtracting a deterministic value based on the time at which a specific primitive is issued. Therefore, neither the TSF timer nor the MLME-HL-SYNC capability can match the precision timing requirements of time-based localization over narrowband RF. Timestamping in the MAC layer through the TSF timer or the MLME-HL-SYNC capability provides in the best case, an accuracy on the order of a microsecond.

Also, the specified tolerances for the TSF timer are rather loose ($\pm 0.01\%$) and in practice its accuracy will likely be substantially worse because the update mechanism does not account for variability in MAC-layer channel access delays. Moreover, the role of

the MLME is strictly limited to issuing the `MLME-HL-SYNC.indication` primitive at certain end-of-packet events; the MAC client is left with the full responsibility for generating the timestamp to this event by consulting some sort of external clock. Therefore, additional delays may be incurred even after the MAC raises an interrupt to record a timestamp.

From the discussions in this section, we concluded that the current support provided in the 802.11 standard for timestamping packet arrival and departure events cannot support the nanosecond-order accuracy required for meaningful localization over RF. This led us to ask the question – *"How to provide the required timestamping support while complying with the current specifications of the 802.11 standard?"*

4.2 Available Implementations for Precision Timestamping

To solve the problem of enabling nanosecond resolution timestamping with 802.11-compatible entities, we studied another system that needs comparable timestamping accuracy. The IEEE 1588 Precision Time Protocol (PTP) [23] is a time-based protocol with a similar precision timing requirement. This protocol is currently in practical use and has been implemented even in commercial systems. By studying the prototype implementations for the PTP, we gained insight into how the placement of the timestamping unit (relative to the network protocol stack), affects the error in the timestamps recorded for packet events.

4.2.1 Precision Timing Requirement in Time-Based Localization

Let us consider a basic challenge-response round from a time-based localization protocol. Recall that a single round of challenge-response in a multilateration protocol includes the following discrete events (Refer Fig.4.2(a)): (i) e_v^v, the verifier sends the chal-

lenge; (ii) e_v^p, the prover receives the challenge and computes the response by applying a pre-arranged function (such as XOR) to it; (iii) e_p^p, the prover sends the response; and (iv) e_p^v, the verifier receives the response and then stores it. The distance between the verifier and the prover is then computed by v from the timestamps as follows

$$D(v,p) \equiv \tau_{vp} = (C(e_p^v) - C(e_v^v) - \hat{\Delta})/2 \tag{4.1}$$

where $\hat{\Delta}$ is the response delay of the prover whose value is publicly known and τ_{vp} is the one-way propagation delay of a reference point within the message, between v and p.

4.2.2 Similarity Between Time-Based Localization and IEEE 1588 PTP

The 1588 PTP [23] is a protocol that has been designed and standardized for synchronizing the clocks of nodes in a local area network. 1588 PTP allows a "slave" entity, to synchronize its clock to a "master" entity (whose clock serves as the time reference). Fig 4.2(b) shows a space time diagram for the sequence of message exchanges between the "master" m and "slave" s.

In our discussions regarding time-based localization protocols, we assumed that all the verifier clocks are synchronized to a common time reference. Hence, while defining the notation in section 2.1.1, a timestamp for event e_x^y captured by any observer entity was denoted simply as $C(e_x^y)$, x being the entity that sent the message, and y being the entity that received it. In the following discussions related to the 1588 PTP, the clocks of the two entities are initially not synchronized. Each entity captures its timestamps according to its local clock, hence the notation must be changed to indicate the entity that captured the timestamp. In the following discussion about the PTP, timestamps captured by an

obeserver entity o, for the event e_x^y, will be denoted as $C_o(e_x^y)$. The other assumptions about the medium being isotropic and anechoic and the normalization between time and space continue to hold.

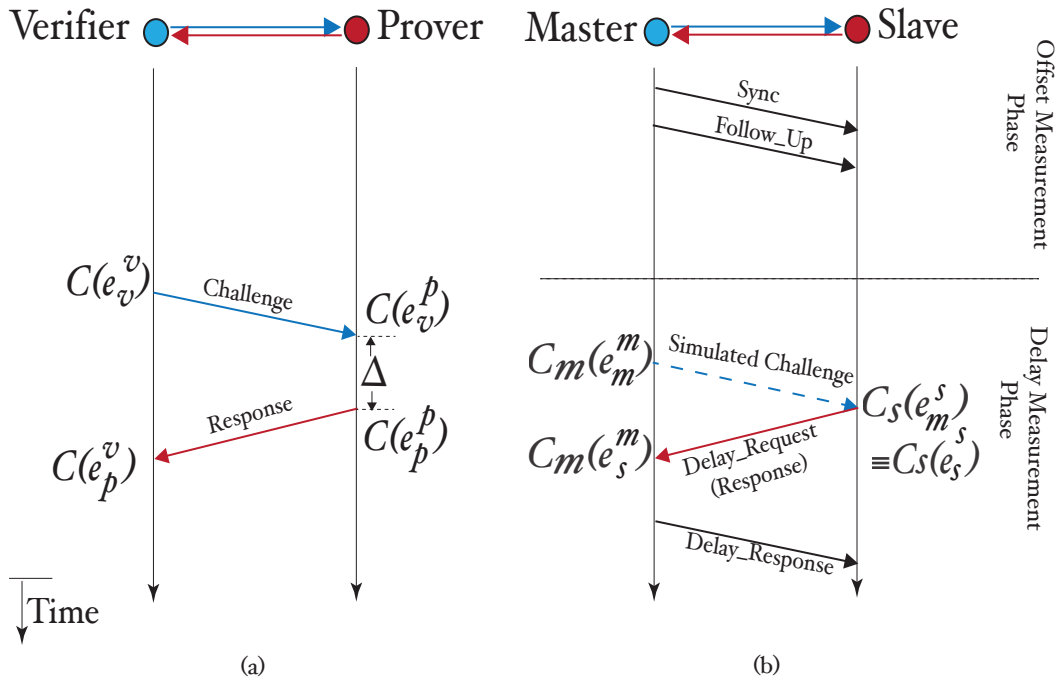


Figure 4.2: (a) a basic challenge-response round in a time-based secure localization protocol (b) message exchanges in the IEEE 1588 Precision Time Protocol

In Fig. 4.2(b), we can see that during the first phase of PTP – the offset measurement phase, the trusted "master" entity v sends periodic "Sync" and "Followup" messages to the slave entity s. The messages sent by the "master" are used by the slave to adjust its clock to satisfy $C_s(\cdot)||C_m(\cdot)$ with an offset lagging behind the master entity's clock by exactly $\tau(m,s) \equiv D(m,s)$, where $C_x(\cdot)||C_y(\cdot)$ denotes the case when the clocks of entities x and y are *syntonized* (i.e., they run at the same rate while maintaining some fixed offset, possibly not zero).

In the second phase of PTP– the delay measurement phase, the slave entity s ef-

fectively *simulates* the basic challenge-response dialog from a time-based secure localization protocol. For this, consider the timed and the untimed packet transmissions: e_s^s, when s sends a message corresponding to the "response" (Delay Request packet), and e_s^m, when the "response" reaches m. The "simulation" inserts an imaginary challenge ahead of the "response", defined by events e_m^m, when it left m, and e_m^s, when it reached s. Because of phase one, s knows that $C_s(e_m^s) = C_m(e_m^m)$ must hold. Moreover, because it is just a simulation, s sets $C_s(e_m^s) = C_s(e_s^s)$, and hence $\Delta \equiv 0$. Therefore, once m sends $C_m(e_s^m)$ as payload of the untimed Delay Response packet, the slave s (but *not* the master m) knows $C_m(e_m^m)$, Δ, $C_m(e_s^m)$, and can find τ_{ms} similar to Eq.(4.1).

We observe that similar to a time-based localization protocol, the IEEE 1588 PTP also has the precision timing requirement and *accuracy* is limited by each participating entity's ability to measure event times. A detailed description of the similarities in message structure as well as timestamping support required by either protocol can be found in [45].

4.2.3 Placement of the Timestamping Unit and its Effect on Error

A few different studies quantify the achievable timestamping accuracy in IEEE 1588 PTP prototype implementations. To understand how the placement of the timestamping unit (relative to the network protocol stack) affects the accuracy of the timestamps, we cite them here:

Timestamping in the Device Driver

Kannisto et al. [26] implemented a Linux PC based prototype for IEEE 1588 PTP. The PTP protocol was implemented as a user module in the application layer. The

message exchanges between the Master and the Slave entity used UDP/IP via Sockets Application Programming Interface (API). Both Ethernet and 802.11 drivers were modified to generate timestamps when PTP packets were transmitted or received. The driver stores the timestamp to a temporary variable when an interrupt is raised by the Network Interface Card (NIC). These interrupts are raised upon issuance of the `PHY_TXEND.confirm` `PHY_RXEND.indication` primitives. When the PTP implementation running in the application layer is informed of a timestamp through the UDP/IP stack, it reads the timestamp from the device driver.

In their experiments, Kannitso et al. gave the Slave clock some initial offset and then started the IEEE 1588 synchronization protocol in both Master and Slave nodes. After discarding the first 5 minutes of "warmup" data, they calculated the average clock offset over the remainder of the measurement period. Using 10 replications of the complete experiment, they calculated the average offset to be $1.8\mu s$ with a variance between replications of $0.7\mu s^2$ over Ethernet. The corresponding values for the 802.11b WLAN implementation were $0.66\mu s$ and $0.2\mu s^2$.

Hardware-Based Timestamping at the PHY-MAC interface

Two experimental studies have investigated hardware-assisted timestamping in commercial IEEE 802.11b hardware to support IEEE 1588 Synchronization. In both studies, the triggering events were derived from interface signals between the transceiver (PHY) and controller (MAC), using the Intersil PRISM 2.4 GHz WLAN Chip Set product family. In these Intersil products, the rising and falling edges, respectively, of the `TX-RDY` interface signal provide the `PHY-TXSTART.confirm` and `PHY-TXEND.confirm` primitives. Similarly, the

rising and falling edges of the `TX-RDY` interface signal provide the `PHY-RXSTART.indicate` and `PHY-RXEND.indicate` primitives. It is interesting to note that the IEEE 1588 standard ([23], section 6.6.5) and the IEEE 802.11 `MLME-HL-SYNC` capability specify their respective timestamp reference points at opposite ends of the packet transmission: whereas 1588 uses "the beginning of the first symbol following the start of frame delimiter", the 802.11 `MLME-HL-SYNC.indication` uses the end-of-packet.

Kannisto et al. [26], who studied timestamping in the device driver, also implemented a prototype for IEEE 1588 synchronization on a pair of Altera Excalibur EPXA1 embedded development boards connected to Intersil HW1151-EVAL transceivers equipped with the (slightly older) Intersil HFA3860B chipset. The ARM9 processors on each board handled the IEEE 1588 protocol, while FPGAs were used to implement the two 32-bit local second and nanosecond clocks for the 1588 protocol and generate packet timestamps triggered by the rising edge of the interface signals. The FPGAs also handled experimental data collection through serial ports connected to an external pulse generator (running at approximately 1 Hz) and a PC analyser connected to both development boards. The simultaneous arrival of a pulse to both development boards triggered their respective FPGAs to send a copy of its clocks (counters) to the PC analyser, which tracked the clock offset between the two boards over a 10 minute measurement period. Using the same experimental setup and procedure described in the previous subsection, they found that the overall offset was $1.1ns$, the variation between replications being $3.1ns^2$.

Unfortunately, despite the remarkable accuracy of their reported results, we must point out that Kannisto's methodology provides almost no information about the measurement error in individual timestamps. Clock synchronization over a long interval is insensitive

to individual timestamp errors, and the symmetric hardware configuration ensures that the timestamping errors will have similar distributions in both directions.[1]

Cooklev et al. [42, 9] measured the one-way network delay between the PHY-MAC interfaces of two Cisco AIRONET series 340 wireless PC cards equipped with the Intersil HFA3861B chipset [10]. To limit the effects of jitter on the air medium (due to changing channel conditions and multipath, etc), the two radios were placed $1m$ apart with clear line-of-sight in an area known to be free of interference in the 2.4 GHz band. This configuration allowed the authors to focus on the jitter induced by the PHY circuitry.

Using an oscilloscope to capture the timing offset between signal transitions at the transmitting and receiving nodes, the authors found the two interface signals had a mean offset of $39.44\mu s$ and standard deviation of $145.6ns$ at the rising edges, compared to a mean offset of $7.35\mu s$ and standard deviation of $594ns$ at the trailing edges, and concluded that the "last-symbol-on-the-air" event is the appropriate timestamp reference point in 802.11 networks. On the other hand, it is important to recognize that these jitter measurements are orders of magnitude larger than the actual signal propagation delay ($3.3ns$) over the $1m$ air gap between the two nodes. Moreover, the spread between the minimum and maximum individual offset values in each experiment – from $39.20\mu s$ to $41.20\mu s$ at the rising edge, and from $-9.95\mu s$ to $9.64\mu s$ at the falling edge – shows how difficult it is to retrofit a timestamp reference point into pre-existing hardware.

[1]Since the variance of the mean of N i.i.d. samples is $1/N$th the population variance, and $N \approx 300$ for one second sampling over a 5 minute experiment, we can use $\sqrt{3.1 \times 300} \approx 30ns$ as a crude estimate for the standard deviation of the individual clock offset samples in Kannisto's experiment – which is remarkably consistent with Cooklev's result of $145.6ns$ for the standard deviation of the individual timing offset samples.

Timestamping Within the PHY Hardware

In the wired Ethernet domain, DP83640 Precision PHYter [41], is a commercially available Ethernet transceiver specially designed to support the IEEE 1588 PTP for real-time industrial applications. In [13], it has been demonstrated that two entities, each equipped with PHYters can be time synchronized to sub-nanosecond accuracy with the "Synchronous Ethernet" mode enabled. The PHYter is capable of timestamping packets very close to the air-interface, immediately after the A/D conversion and symbol detraction. The PHYter contains a local PTP clock operating at $250MHz$, programmable to frequencies obtained by integral division of the base clock, and a counter which is incremented every $8ns$. It is also capable of parsing the packets on-the-go, and triggering timestamps at the A/D sampling stage within the PHY hardware. These timestamps are then inserted into the payload of the packet itself – as a packet is being transmitted onto the medium for sending, and while receiving a packet, before its contents are sent to the higher layers of the protocol stack. This capability enables highly accurate synchronization without using "followup" messages. Commercial transceivers with capabilities like the PHYter are not yet available for 802.11-capable entities, or for entities operating in any other wireless domain.

Summary of the Observations from These Studies

The experimental prototypes discussed in this section are tabulated in Table 4.1. We observe that the possibility of recording accurate timestamps only increases as we move the timestamping point down the protocol stack into the network interface card. The table clearly shows that even if the timestamps are captured in hardware at the PHY-MAC interface, the accuracy achieved is still on the order of μs. Therefore to achieve the desired

accuracy on the order of ns, we must implement timestamping within the PHY hardware, as close as possible to the "PHY-medium interface".

Table 4.1: Comparing Different Implementations of IEEE 1588 PTP

Work by	**Medium**	**Timestamping Point**	**Average Offset**
Kannisto et al. [26]	Ethernet	Device Driver	$\approx 1.8\mu s$
Kannisto et al. [26]	802.11b	Device Driver	$\approx 0.66\mu s$
Cooklev et al. [9]	802.11b	PHY-MAC interface	$\approx 0.2\mu s$
Kannisto et al. [26]	802.11b	PHY-MAC interface	$\approx 1.1ns$
D. Miller [13]	Ethernet	A/D conversion	$< 1ns$

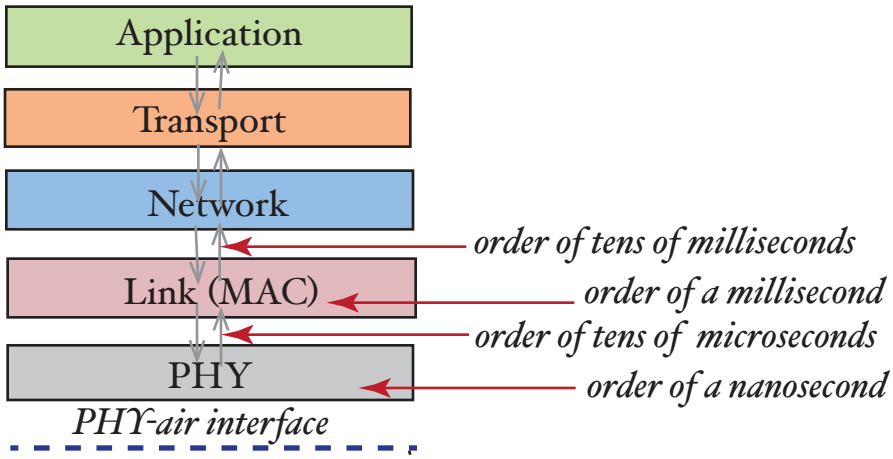


Figure 4.3: Achievable accuracy by varying the timestamping point in the protocol stack – as shown by experimental studies from various sources.

4.3 Difficulty in Controlling the Message Sending Time

A time-based protocol runs in the application layer of the network protocol stack. However, the message arrivals and departures occur in the network transceiver card, at the air interface (antenna). Due to this, there exists a discrepancy in the time at which the application intends to send a packet and the time at which the packet is actually transmitted.

The underlying reason is that any kind of communication data experiences non-deterministic delays in passing through the software (network protocol stack) and hardware of a device before the actual transmission. Sending a message at the exact time decided by the protocol is not feasible due to delays across asynchronous interfaces – from host application to operating system to network interface controller and even to the transceiver. OS scheduling delays in the higher layers of the network protocol stack are a major contributor to this. As experimental proof, we cite experiments conducted by Pasztor and Veitch [47], where packets were scheduled to be sent with a fixed inter-departure time. The measured inter-departure times showed an error in the order of a few milliseconds, even when the real-time linux OS used in the experiments was running the **send()** process as the only user process. In reality however, there are many more processes in addition to the **send()** process would be running simultaneously in a system running normally without experimental control. Therefore even more error can be expected during sending a message.

If an outgoing packet carries a timestamp inserted at the application layer, the value of the timestamp is the *expected time* at which the application layer intends to send the packet. Since the amount of delay experienced as it travels down the network protocol stack is unpredictable, it is not possible for the application layer to insert a corrected timestamp in advance. Techniques can be used to minimize the effect of these non-deterministic delays as the message travels from the application layer, down the protocol stack until it is actually transmitted onto the medium. For example, IEEE 1588 PTP uses "followup messages" (Fig. 4.2b) to minimize error caused by these delays. When a beacon packet (Sync) is sent, a timestamp is captured at a lower layer closer to the actual time of departure, and sent back to the application layer informing it of the correct time of transmission. Then the "followup

message" sent by the "master" entity contains this timestamp in its payload. This helps the "slave" entity to minimize the error introduced due to the discrepancy in the intended and actual sending times of the "sync" packet.

4.4 Timestamps can be Processed Offline

Although secure localization protocols require very accurate timestamps, there is no requirement for real-time processing and the timestamps can be reported to the application after a (reasonably) small delay. This observation is important because it allows for design such that only timestamping unit is implemented in hardware close to the "air-PHY" interface. The timestamps can be processed offline in the application layer of the individual entities participating in the localization protocol. Once the timestamps are captured, they may also be sent to some central server in the system, which handles all the processing required to extract the required timing information.

One example demonstrating this flexibility in design are the Aeroscout System [1]. Another example is the system used in [69], where raw A/D samples collected (at different entities) upon the arrival of a message, are sent to to a central server. The server processes the samples for offline timing alignment via cross correlation. An arrangement of this kind, where a central entity performs the complex timing and signal processing functions may not be feasible in settings where there is no network infrastructure, for example in ad-hoc networks. An alternative for such scenarios could be to capture timestamps in a real-time manner and buffer them, until a higher layer in the protocol stack can retrieve and process them offline. Such a timestamping unit would require a few simple logic blocks in the PHY hardware for buffering, and therefore would be simple to implement.

4.5 Adding Precision Timing to the 802.11 PHY

After studying the available prototype implementations for the PTP, we observed that in order to achieve timestamping on the order of nanoseconds, we must implement timestamping within the PHY hardware. However, we also noted in section 4.4 that only the timestamp capture functionality needs to be in hardware. Other functionalities related to processing the timestamps and extraction of timing information can be at the application layer, or even at a different entity. This is because the target applications under consideration in this dissertation do not need real-time processing of the timestamps.

Separate from the MAC-layer TSF timer, every 802.11 PHY needs a high-precision *reference oscillator* to regulate both the transmit center frequency and symbol clock within its transmit logic. Depending on the chosen combination of modulation scheme and data rate, the specified tolerance[2] for the reference oscillator is never weaker than $\pm 25ppm$ – which is 40 times more strict than the tolerance for the MAC-layer TSF timer! Since the tolerable error in the oscillator is on the order of picoseconds, the symbol times for defined transmit schemes (set modulation and data rate) can be assumed to be known and constant across transmitters and receivers.

To take advantage of the PHY's existing reference oscillator, we now propose to add an "interval timer" to the PHY, i.e., a free-running counter. The counter is clocked at some multiple of a Gigahertz. This can be easily done since GHz frequencies are already generated inside the PHY for transmission of signals across the air medium. Whenever some application at entity v needs to generate high-precision timestamps at packet-boundary

[2]The transmit center frequency tolerance for all versions of the PHY are given as $\pm 25ppm$ except as follows. For 1 Mbps operation, the tolerance is specified as $\pm 60KHz$ on the 2.4 GHz band, which is equivalent to $\pm 25ppm$. For the 5 GHz band, the tolerance is specified as $\pm 20ppm$ for the 20 MHz and 10 MHz sampling rates, and $\pm 10ppm$ for the 5 MHz sampling rate.

events, it would use this PHY counter to emulate $C_v(\cdot)$ – the timestamping clock at v. Otherwise, the PHY counter logic could be disabled to reduce power consumption in the PHY, similar to the optional `MLME-HL-SYNC` capability in the current IEEE 802.11 standard.

It is important to recognize that it is just a simple, uncalibrated interval timer, not a full 1588-style clock, to avoid adding an unreasonable amount of complexity to the PHY. Using a symbol time as the absolute reference, the receiver can count the number of ticks of this counter per incoming symbol, and use it to synchronize its counter to the transmitter's counter.

To avoid the difficulties of attempting to control this interval timer remotely from an application program, we propose that the PHY counter be linked to read-only registers that automatically store its value at the most-recent start-packet or end-packet event, respectively, whenever the PHY counter is enabled. Thereafter, each stored counter value remains in its respective register until it is overwritten by events generated by the next packet. This provides the application program with a (relatively-large) window of time in which to retrieve the stored values of $C_v(e_x^y)$ from the PHY registers, without further degrading the data due to the addition of an offset or some jitter to the retrieved value.

Triggering Timestamps for Packet-Boundary Events

The simplest method for triggering the required timestamps would be to follow the IEEE 802.11 `MLME-HL-SYNC` capability and the experimental studies described in section 4.2.3 in using some existing MAC-PHY interface signals. Even this naïve approach should provide better accuracy than the `MLME-HL-SYNC` capability, because the same PHY logic that issued the `MLME-HL-SYNC.indicate` primitive could simultaneously trigger a timestamp from

the PHY counter, without waiting for the MAC client to respond to this primitive and generate a timestamp from another clock. However, these MAC-PHY interface signals are too far removed from events at the air-PMD interface to provide precision time stamping.

Cooklev et al.'s experiments showed that the jitter in the 802.11 PHY hardware is enough to push the error in the timestamp to the order of a μs. Even an error of $1\mu s$ in measuring the message propagation time between two entities corresponds to an uncertainty of approximately 10 bits in packet length (using a data rate of $11Mb/s$) or $260m$ in the distance between the two nodes! The major reason for this discrepancy is that the start-of-packet event at the air-PMD interface only affects the MAC-PLCP interface signals indirectly, and the offset between the two layers is inherently different for transmitters and receivers and also varies significantly between different combinations of modulation scheme and data rate.

To highlight the issue, let us define an "ideal" timing reference point for the start-of-packet event to occur when the end of the last bit from the Preamble and PLCP Header passes through the air-PMD interface. (A similar argument can be made for the end-of-packet event, but is omitted due to limited space.) From within the PMD, it should be possible (at least in theory) to determine the time of such events with uncertainties on the order of a single sampling period – although the answer may be delayed considerably to allow the PMD to carry out some off-line computations involving many samples. Nevertheless, *even if an oracle could instantly reveal the exact time of this "ideal" start-of-packet event to the PLCP*, neither the transmitter nor the receiver could change the time at which it issues the `PHY-TXSTART.confirm` or `PHY-RXSTART.indication` primitive, respectively.

The reason for this behavior – together with the fact that the IEEE 802.11 stan-

dard does not specify the exact timing of these primitives, relative to events at the air-PMD interface – should be evident from Fig. 4.1. At the transmitter, the `PHY-DATA.confirm` primitive must be issued *before the end* of the header error check (HEC) has been transmitted, but it could be as early as the start of the Preamble. Similarly, the `PHY-DATA.confirm` must be issued *before* any of the associated data is transmitted, and the `PHY-TXEND.confirm` must be issued *after* the transmission of last bit of the packet. Conversely, at the receiver, the `PHY-RXSTART.indication` primitive must be issued *after the end* of the HEC has been received, but it could be much later as long as the PHY has enough buffer space. Similarly, the `PHY-DATA.indicate(DATA)` and `PHY-RXEND.indicate` primitives must be issued *after* the associated data has been received.

Having explained why we cannot use primitives PHY-MAC primitives to trigger timestamp capture, we propose a different way of triggering timestamp capture. During reception, when the packet arrival is detected at the receiver antenna, the signal encounters various signal processing modules within the PMD layer. After analog processing, A/D conversion, and timing recovery, the signal encounters a symbol detection block which is the closest point to the air-interface, where the "ideal" reference symbol can be identified. To minimize the error due to PHY jitter, the symbol detector unit should trigger timestamp capture for a packet arrival event. During transmission, it is easier to identify the reference symbol and trigger a timestamp capture. Within the PMD, the bits received from the MAC layer are packaged into symbols depending on the modulation scheme used for transmission. When the encoding module forms the symbols, it can trigger timestamp capture when the symbol matching the reference symbol is created. Hence, we propose that the timestamps must be triggered corresponding to symbol detection and creation within the PMD.

4.6 Conclusions

In this chapter, we considered the architectural support needed to implement time-based localization protocols like EM in 802.11-based networks. For acceptable accuracy when localization is executed over narrowband RF (the de facto medium of propagation in 802.11-based networks), the entities must be able to timestamp packet events with an accuracy on the order of a nanosecond. The 802.11 standard currently supports timestamping in two ways – with the Time Synchronization Functionality (TSF) and an optional capability called `MLME-HL-SYNC`. Both these features allow 802.11-compatible entities to capture a timestamp in the MAC layer and at best, allow for timestamping accuracy on the order a microsecond. Since this level of accuracy is three orders of magnitude worse than the required accuracy (on the order of a nanosecond) for meaningful localization, timestamping support currently provided by the 802.11 standard if not sufficient for time-based localization.

We proposed hardware-based timestamping within the 802.11 PHY to enable precision timestamping for time-based localization. First, we studied existing prototypes in which non-standard timestamping mechanisms are appended to the standard 802.11 protocol stack for better timestamping accuracy. In some prototypes device driver software is modified to increase the accuracy of the timestamps. Software modifications such as these are easy, however the accuracy obtained with such modifications is still on the order of microseconds. To achieve timestamping accuracy on the order of a nanosecond, it is necessary to capture timestamps in hardware.

Making modifications to the PHY hardware is generally not desirable for the NIC manufacturing community. Hence it was important to consider results from prototypes

where timestamping is hardware-based, but placed at the PHY-MAC interface. Such prototypes have the timestamping logic on custom hardware, most likely an FPGA connected to the PHY-MAC interface. Timestamps are triggered when the `PHY-RXSTART.indication` and the `PHY-RXEND.indicate` primitives are issued across the interface. In work by Cooklev et al., the error introduced due to jitter in the PHY circuitry during transmission and reception, was measured by exposing the PHY-MAC signal transitions. They found that the offset between the signal transition indicating a transmission at the sender and the signal transition indicating reception of the packet at the receiver has a standard deviation of few hundred nanoseconds. This equates to a couple of hundred meters in distance measurements, which is not acceptable for meaningful localization.

Therefore, to achieve timestamping accuracy on the order of a nanosecond, we must capture timestamps *within* the PHY itself, as close as possible to the air-interface. To accomplish this, we propose the addition of a few simple logic blocks including a free running counter clocked by the crystal oscillator, a buffer consisting of some registers, and a supporting mechanism to trigger timestamp capture at symbol detection/creation. Integrating the proposed hardware-based timestamping unit into the PHY would make it possible for the 802.11-compatible entity to timestamp packet arrival and departure events with nanosecond precision (similar to the performance of PHYter, a comparabile timestamping mechanism in the Ethernet domain). Lack of hardware-based precision timestamping support similar to that proposed by us, is currently the biggest impediment to accurate time-based localization in 802.11-based networks. Precision timestamping support is essential for obtaining localization results with accuracy on the order of a few meters in 802.11-based networks.

Chapter 5

Anatomy of the Error Introduced During Message Transfer

In chapter 4, we discussed the architectural support required for localization with 802.11-compatible entities. We showed that timestamps for message arrival and departure events must be captured within the PHY, as close as possible to the air-interface. By studying the interactions between the different modules of the PHY, we also concluded that the appropriate place to trigger timestamp capture within the PHY is when symbols are detected/created. Ethernet transceivers with such hardware-integrated timestamping capability (PHYter [41]) are already available. However, such transceivers are currently not available in the wireless domain. Researchers from the area of precision clock synchronization [13] and time-based RF ranging [9, 45], have highlighted the need for similar hardware-integrated timestamping capability 802.11-compatible transceivers. With the number of applications requiring wireless transceivers capable of precision timestamping, we expect that 802.11-compatible transceivers with capabilities similar to the PHYter, will

be commercially available in the near future.

In this chapter, we take our discussion on achieving "proper timing resolution" a step further. Under the assumption that 802.11-compatible transceivers have the capability to capture timestamps at symbol detection/creation within the PHY, we focus on sources of error during message propagation *between the timestamping points* – at the sender and at the receiver. Fig.5.1 shows the path of the message between these two timestamping points. The path of the message consists of: signal processing blocks in the sender's PHY hardware, channel effects of the wireless medium and the signal processing delays in the receiver's PHY hardware. In addition, clock offset and quantization also introduce error into the measurements. In the first part of this chapter, we present an anatomy of the error introduced during message propagation. We also propose ways to minimize each component of error in order to increase the accuracy of localization.

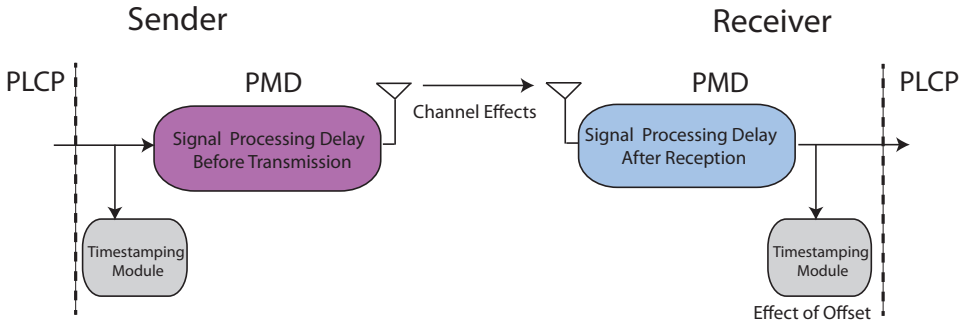


Figure 5.1: Path of the message between timestamping points at the sender and the receiver.

Other than error introduced during message propagation, the accuracy of the timestamps is limited by the resolution of timestamping clock. Throbjornsen et al. [63] proposed a technique to measure the propagation time of messages between a sender entity and a receiver entity with a resolution greater than the time period of the clock on either entity (*subclock accuracy*). Their technique is based on the "vernier effect" – when the

sender and the receiver clocks have the same time period but have a phase offset, multiple measurements can be used to compute the time of flight between them with a virtual resolution, which is greater than the resolution of either clock. In the second part of this chapter we show how Throbjornsen et al.'s technique can be applied to SIMO localization protocols. In particular, we show how the time of flight can be measured with subclok accuracy in both Elliptical Multilateration (EM) and Hyperbolic Multilateration (HM).

We also quantify the effect of clock synchronization on the accuracy of the constraints formed on the prover's location in both EM and HM. In chapters 2 and 3, we mentioned that in current implementations of HM, the verifier clocks are synchronized prior to execution of the challenge-response rounds. However, achieving precision clock synchronization over wireless is very difficult with off-the-shelf transceivers and low frequency clocks. It is also difficult in ad-hoc networks, where synchronization must be achieved in a peer-to-peer manner. As a solution to this, Saha and Molle [54] introduced synchronization-free HM in their protocol *Localization with Witnesses*. Although synchronization-free HM is highly desirable in certain scenarios, it is worth asking *"How does the lack of synchronization amongst verifiers affect the accuracy of localization in Hyperbolic Multilateration?"* We answer this question by use of theoretical analysis and experimental results.

5.1 Error Introduced Due to The Wireless Channel

When message transfer occurs between a sender and a receiver, the signal characteristics like data rate, carrier frequency etc. are selected by the sender, depending on the channel conditions and the communication standard in use. However, some signal characteristics like Signal-to-Noise Ratio (SNR) and phase with respect to the sender's transmit

clock depend on the wireless channel conditions. Therefore, when a signal is received over the wireless medium, the signal processing blocks in the receiver must "learn" the altered characteristics, and tune the parameters of the signal processing algorithms accordingly, to recover the information from the signal. Changing channel conditions introduce variable delay into the measured propagation time of the signal.

Consider the multipath effect, which is caused by separate copies of the same transmission reaching the receiver via different paths. The earliest and strongest component is the line-of-sight (LOS) component, while the signal bouncing off some object, traverses a non-line-of-sight (NLOS) path and reaches the receiver slightly delayed. Due to interference of the LOS and NLOS components, the signal-to-noise ratio changes on a packet-to-packet basis. Multipath effect causes error in detecting the time-of-arrival (ToA) of the signal, irrespective of the method used to detect the ToA.

5.1.1 Mitigating Error Due to Channel Effects

In systems where cross-correlation is used, the time-of-arrival is estimated by detecting the first peak in the correlation function. Interference causes the peak in the correlation function to shift and attenuate. This introduces error into the estimate for the time-of-arrival (ToA) of the signal. In other systems, the ToA is estimated by recording a timestamp for the arrival of a "reference symbol" in the packet. 802.11-based systems use this method. The receiver must "learn" the frequency and phase of the incoming signal before it can make sense of the incoming message and detect the reference symbol. 802.11 packets have 128 sync bits in the preamble to allow the receiver to "learn" and "lock on to" the incoming message before receiving the actual header and payload. Depending on

the channel conditions and multipath, the gain has to be adjusted at the analog front end. These factors also change the delay incurred in the digital timing recovery loop (details in the next section). Since channel effects introduce variable delay in the analog and digital signal processing of the incoming message, the value of timestamp recorded for the time-of-arrival always has a positive error with respect to the true time-of-arrival.

The prototype implementations for time-based localization that we discussed in chapter 2 use a few different methods to mitigate error due to channel effects. In the PinPoint [70] time-based location determination system designed for wireless sensor networks, the effect of multipath is reduced by selecting only the first and strongest chain of received baseband signals for time-of-arrival estimation. Since the NLOS components are weaker and reach the receiver slightly delayed, this method ensures that only the LOS components are considered for the computation. Separating the LOS components from the NLOS components of the signal is easy in the PinPoint system because it uses an an unconventional signaling method consisting of a repetitive pattern of baseband pulses. Such separation may not be possible in generic wireless transceivers including those that comply with the 802.11 standard. Therefore this technique cannot be generalized to all transceivers. Lanzisera et al.'s [30] technique for mitigating error due to multipath is more suited for use in 802.11-based systems. Leveraging the fact that the error due to multipath is frequency dependent, the authors used frequency hopping to measure the time it takes a message to propagate between a sender and a receiver. To use the frequency hopping technique, multiple measurements were made over different carrier frequencies, keeping the positions of the sender and the receiver fixed. The magnitude of error introduced is then characterized as a function of frequency used. Error due to the multipath effect can be isolated during the

actual measurements using this characterization, leading to a better estimate of the true propagation time. This technique can be easily applied in DSSS based 802.11 networks.

Prototype implementations for time-based ranging systems have also demonstrated that that removal of outliers makes a significant difference in the accuracy of the result. The raw data collected in time-based ranging systems in [37] and [30] show that the outliers are generally far from the true trend of the data, and contribute to large errors. Mazomenos et al. [37] improved accuracy by removing all the samples that fell outside of one standard deviation of the computed mean. Sanitizing the data in this manner can be helpful for any time-based ranging system, including time-based localization in 802.11-based networks.

As with any stochastic experiment, simple averaging over a large number of measurements is a proven approach to obtain better accuracy. The time-based ranging systems for WSNs introduced by Thorbjornsen at al. [63], and improved by Mazomenos et al.[37], use averaging to mitigate error due to channel effects. In their results, they stated that it is very difficult to isolate the error due to channel effects in individual measurements. Therefore they approach error mitigation by leveraging the fact that error distribution is gaussian [35], and averaging over a large number of measurements leads to a result which is close to the true propagation time. Averaging over multiple measurements for better accuracy is also used in conventional GPS receivers. As we will observe later in this chapter that averaging not only mitigates error, but also allows entities to make measurements with a resolution greater than that of the clock period used for timestamping.

Overall, error due to channel effects in time-based localization can be mitigated by (i) frequency hopping, if the communication standard in use supports it (ii) removing any samples that fall outside of one standard deviation from the mean, and (iii) averaging

over a large number of measurements.

5.2 Error Due to Signal Processing Before/After Timestamping

In chapter 4, we proposed a hardware-based timestamping mechanism that will allow 802.11-compatible entities to capture timestamps at the symbols detection/creation stage. Although this allows the entities to capture timestamps with an accuracy on the order of nanoseconds, the message still incurs some delay between the timestamping point in the hardware and the air-interface (antenna).

When a message is about to be sent, the timestamp for the departure event is captured by the hardware-integrated timestamping unit as soon as the "reference symbol" is inserted into the outgoing message. The message then passes through various signal processing clocks where it is scrambled, coded and modulated according to the communication standard in use. Therefore, before the message is transmitted by the antenna onto the wireless medium, it incurs some delay after the timestamping point. The delay incurred on the sender side can be quantified when the specifics of the message transfer are known, for example, the data rate, modulation scheme, the length of the message etc.

It is much more difficult to quantify the delay in the signal processing blocks on the receiver side. As explained in the previous section, the receiver must "learn" the characteristics of the incoming signal and perform timing recovery to "lock-on" to the incoming signal for proper reception. In order to analyze the different points at which delay is introduced into the message path during reception, we briefly describe the signal

processing stages in an 802.11b receiver [1]. The following discussion presents a summary of the signal processing steps that the message must undergo between its arrival at the antenna and the timestamp capture corresponding to its arrival at the receiver. Although there are some architectural differences among 802.11a/b/g/n receivers, the techniques that we will describe correct for the error due to signal processing can be applied to a generic 802.11 receiver.

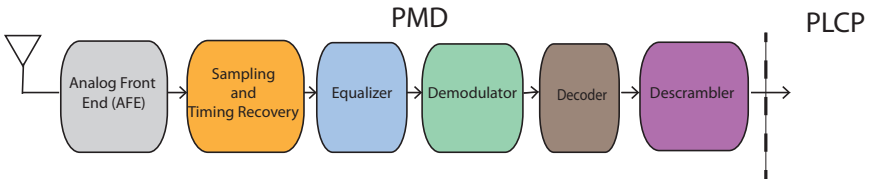


Figure 5.2: Architecture of the PMD in an 802.11b receiver. The message must pass through various analog and digital signal processing blocks between the instant when a message arrives at the antenna and the instant when the reference symbol is detected. The timestamp capture for the arrival event occurs only after the message has gone through these signal processing stages.

Fig. 5.2. shows the a block diagram illustrating the signal processing stages in an 802.11b receiver before the "reference symbol" is detected. Upon detection at the receiver antenna, the signal first passes through the analog front end (AFE) of the receiver. In the AFE, a matched filter compensates for the channel impulse response, and the automatic gain control (AGC) scales the signal to the desired power level. Adjusting the power level is essential for proper operation of the analog to digital converter (ADC) and other downstream modules that the signal will encounter, as it passes through the various signal processing stages in the receiver. We term the delay experienced in this module as the analog processing delay d_{AFE}.

[1] Our treatment of this topic will only be detailed enough for understanding the delays incurred before the timestamping point in the receiver. For an in-depth description of the internals of all-digital receivers (802.11b receivers in particular) and the mathematical aspects of the signal processing, the reader may use [32][12][19][20][7] as references

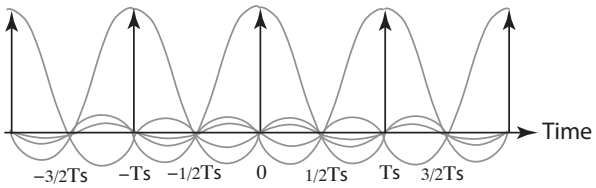


Figure 5.3: Baseband signal comprised of raised cosine pulses, with a time period of T.

Let the baseband signal fed to the analog-to-digital converter (ADC) be represented as

$$z(t) = \sum_{m}^{kT_i} a_m g(t - mT - \epsilon T) \tag{5.1}$$

where g(t) represents the baseband pulse (Fig. 5.3), which has the shape of a raised cosine in 802.11-based transceivers [16] with a period of T. T_i is the symbol period at the output of the interpolator (described later), and ϵ is the fractional delay in terms of a symbol period. This signal is converted to its digital equivalent for the succeeding digital signal processing modules. Sampling of the analog signal occurs at the ADC, which is the first stage of a *digital timing recovery module.*

The timing recovery module in a receiver may be completely analog, completely digital or hybrid (as shown in Fig. 5.4). In analog and hybrid timing recovery, a phase locked loop is used to track the incoming symbol frequency, and controls a voltage controlled oscillator (VCO) to keep the receiver locked on to the incoming symbol rate. The VCO also clocks the ADC in both analog and hybrid receivers. In all-digital timing recovery, the ADC is clocked independently from a source derived from the local crystal oscillator, and the ADC sampling pulses are asynchronous with the incoming symbols. Instead, timing recovery is done exclusively in the digital domain through interpolation. This eliminates

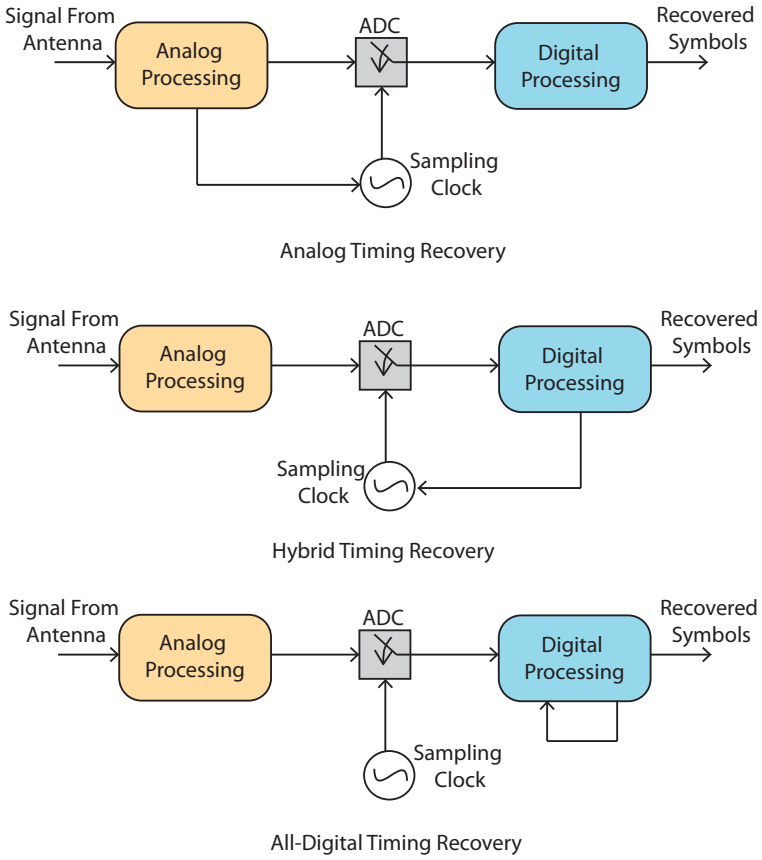


Figure 5.4: Different methods for timing recovery in the receiver. In all-digital receivers, the sampling is clocked by an independent reference derived form the local crystal oscillator.

the need for many analog components including the VCO, which makes the receivers robust against failure, smaller in size, and less expensive. Therefore, all-digital timing recovery has become the de facto standard in modern receivers.

The first stage of timing recovery is A/D convertor. Here, the signal is sampled and converted to its digital equivalent. It is then fed to the succeeding digital signal processing modules. The timing recovery process adds a non-deterministic delay to the time taken by the symbol detector to detect the "reference symbol" after it arrives at the antenna. Studying the specifics of the timing recovery module is important for estimating the amount

of delay incurred and correcting for it.

Consider Fig. 5.5, which shows the baseband signal with the peak power at the center of each raised cosine pulse. The sampling pulses of the ADC are shown in the bottom half of the figure. Depicted in this figure is also ϵT – the fractional symbol period between the closest sample base point and the peak power of the corresponding baseband symbol. The value of ϵ varies from packet to packet. It depends upon the true distance between the sender and the receiver. It is important to note that ϵ is available as a numerical value inside the timing recovery loop, but is not communicated outside of the loop in a generic 802.11 receiver.

If the sampling rate is $1/T_S$, the sampled signal can be represented as

$$x(n) = z(nT_s) \tag{5.2}$$

The sampled signal from the ADC is the input to the all-digital timing recovery loop. The timing recovery loop provides a reconstructed signal as its output.

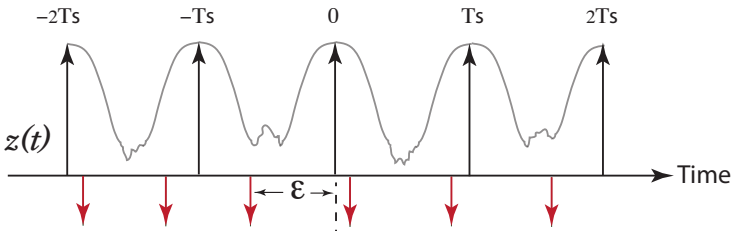


Figure 5.5: Sampling the incoming signal by the ADC. The upper half denotes the incoming baseband signal and the lower half denotes the sampling pulses

Fig. 5.6. shows the timing recovery loop structure used in 802.11b receivers. The interpolator is the first component of the timing recovery loop. It is followed by a decimator,

the output of which are the reconstructed symbols. Further in the loop are the timing error detector, loop filter and the controller for the interpolator.

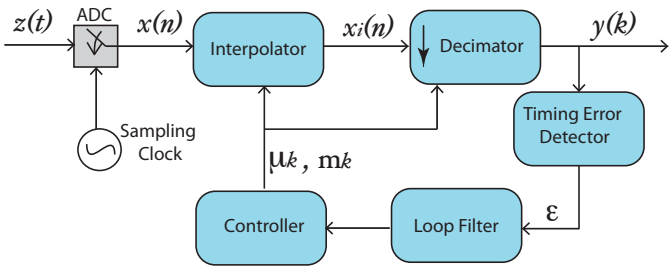


Figure 5.6: An all-digital timing recovery loop in 802.11b receiver

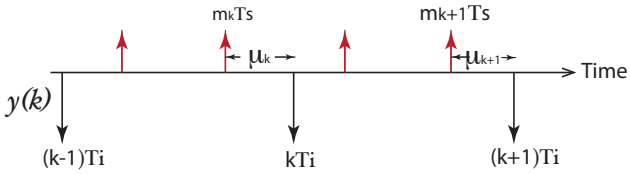


Figure 5.7: Decimation at the appropriate basepoints with corresponding fractional intervals

(i)The *timing error detector* computes an estimate of the fractional delay parameter ϵ. This component in an all-digital timing recovery loop, corresponds to the phase detector in a phase locked loop (PLL) present in analog or hybrid timing recovery modules. It computes ϵ depending on the phase difference between the incoming signal and the local reference clock.

(ii) The *loop filter* forms a control signal by filtering the error signal from the timing error detector by proportional integration.

(iii) The *controller* module's task is to provide the interpolator, decimator and the error detector with the appropriate parameters for optimal signal reconstruction. De-

pending on the estimate of ϵ, it computes the base point index for decimation m_k, and the corresponding fractional interval μ_k. The value of μ_k is fed to the interpolator, and the basepoint index m_k is fed to the decimator. Based on these two values, the controller also selects the correct samples for the interpolator.

(iv)The *interpolator* computes the optimum strobe values to reconstruct the signal. Ideally a symbol should be sampled at certain desirable points. If a single sample is to be taken per symbol, it should correspond to the peak power. However, since sampling is not synchronized to the incoming signal in all-digital receivers, the samples are offset from the ideal sampling points. The task of the interpolator is to reconstruct the optimum strobes as if the signal were sampled at the ideal points, from the series of non-ideal samples that are obtained from the ADC. The other inputs to the interpolator are parameter values from the controller, which it uses for the interpolation process. The output of the interpolator is

$$x_i(n) = z(nT_s + \mu_k T_s) \tag{5.3}$$

(v) The *decimator* downsamples the sampled signal that was fed to the interpolator, at the computed basepoint indices and fractional corresponding intervals computed for each symbol time. In 802.11b receivers, the output from the decimator is one sample per symbol time, representing the reconstructed symbols. The signal reconstructed by interpolation [2] can be expressed as

$$y(k) = x_i(m_k) \tag{5.4}$$

[2]For a comprehensive tutorial on interpolation and decimation of digital signals, and the use of these signal processing techniques in conjunction to reconstruct original signals from samples, the reader may refer to [12].

The reconstructed symbols at the output of the timing recovery module are then processed further by signal processing modules following the timing recovery loop. These modules are the carrier phase recovery module, equalizer, decoder, demodulator and descrambler. The combined delay in all these blocks has been referred to as the digital processing delay in existing literature. Detection of the reference point for timestamping can only occur after the signal has been processed by all these blocks. Upon detection of the reference point, the timestamping module is triggered to capture a timestamp for the arrival event.

5.2.1 Correcting for the Delay Incurred in the Signal Processing Modules

To compensate for the error introduced due to signal processing in the various blocks of the PMD, three individual components of this error must be estimated and subtracted from the timestamp captured for the arrival event. The three components of signal processing delay are: delay in the analog front end (AFE), delay in timing-recovery loop and delay in all other digital signal processing blocks combined. In work by Exel at al. [16], its has been demonstrated that the error in the AFE can be estimated if channel parameters like gain, SNR etc. are known. The manufacturer can provide a characterization of the AFE delay as a function of these parameters. Assuming that these parameters are known, it is possible to estimate the delay in the AFE by looking up the characterization. In our discussion of the digital timing recovery module, we explained that the delay in this module depends on the value of the parameter ϵ. In currently available receivers, ϵ is a temporary variable that is used locally within the timing recovery loop, but not communicated to any module outside of the loop. The only way in which the value of ϵ can be used for error cor-

rection would be to modify the receiver architecture so that its value is communicated to the timestamping module. Although it might seem that modifying the PHY architecture is too difficult, the need for computing this delay is so pronounced for both localization and clock synchronization applications, that new architectures are already emerging to support this feature. For example, a prototype implementation of such a receiver architecture for 802.11 b can be found in Exel's work [16]. The delay in other digital signal processing modules can also be estimated similar to the estimation of the delay in the AFE. The manufacturer can provide a characterization of the delay in each digital signal processing module depending on parameters like packet length, data rate, modulation, frequency etc. When these parameters are known, it is possible to look up the delay in each module. Overall, we find that the delay in the timing recovery module is the unpredictable component. However, it is possible to estimate and correct for this delay if the necessary architectural support is added to 802.11 receivers.

5.3 Error Due To Clock Offset and Quantization

The resolution of a timestamp captured for an arrival or departure process is limited by the frequency of the clock that drives the timestamping unit. In chapter 4, we described how clock for the hardware-integrated timestamping unit can be derived from the physical oscillator in the transceiver. Generic 802.11 transceivers use crystal oscillators which have a frequency of either 20 or 40 MHz. Although a wide range of frequencies starting from the low MHz range to the high GHz range can be generated from the base frequency of the physical oscillator, the GHz frequencies are used only for up-conversion of the baseband signal before it is transmitted. All other logic circuits are clocked in the

lower MHz range to conserve power, which is an important design consideration for mobile wireless entities. This is the reason why it is desirable to clock the timestamping module with the base frequency of the crystal oscillator itself, which is $20MHz$ or $40MHz$ for off-the-shelf 802.11 transceivers.

Consider the radio transceiver of an off-the-shelf entity capable of wireless transmission. By default, the transceiver is in the receive mode, when the antenna constantly senses the medium for an incoming message. During this time the transmit circuitry of the entity is powered off. The transmit circuitry is powered on only when the entity is ready to transmit a message. Due to the intermittent "OFF" states between transmissions, the phase of the transmitted signal varies randomly from transmission to transmission. The radio state control diagram of the TI CC2430 transceiver (used in work by [63] and [37] for accurate time-based localization) is illustrated in Fig. 5.8. The state diagram shows how the transmit circuitry goes into intermittent "OFF" states when it is powered down. Since the transmit circuitry powers up asynchronously to the timestamping clock, the phase of the transmitted message, as perceived by the timestamping clock is random for every message transmission. Therefore, it is not possible to correlate the phases of two transmitted messages, or the phases of a received message and a transmitted message, even for the same transceiver. The timestamping clock at the receiver perceives a random phase offset for every incoming transmission, even when the transmissions were made by a single sender.

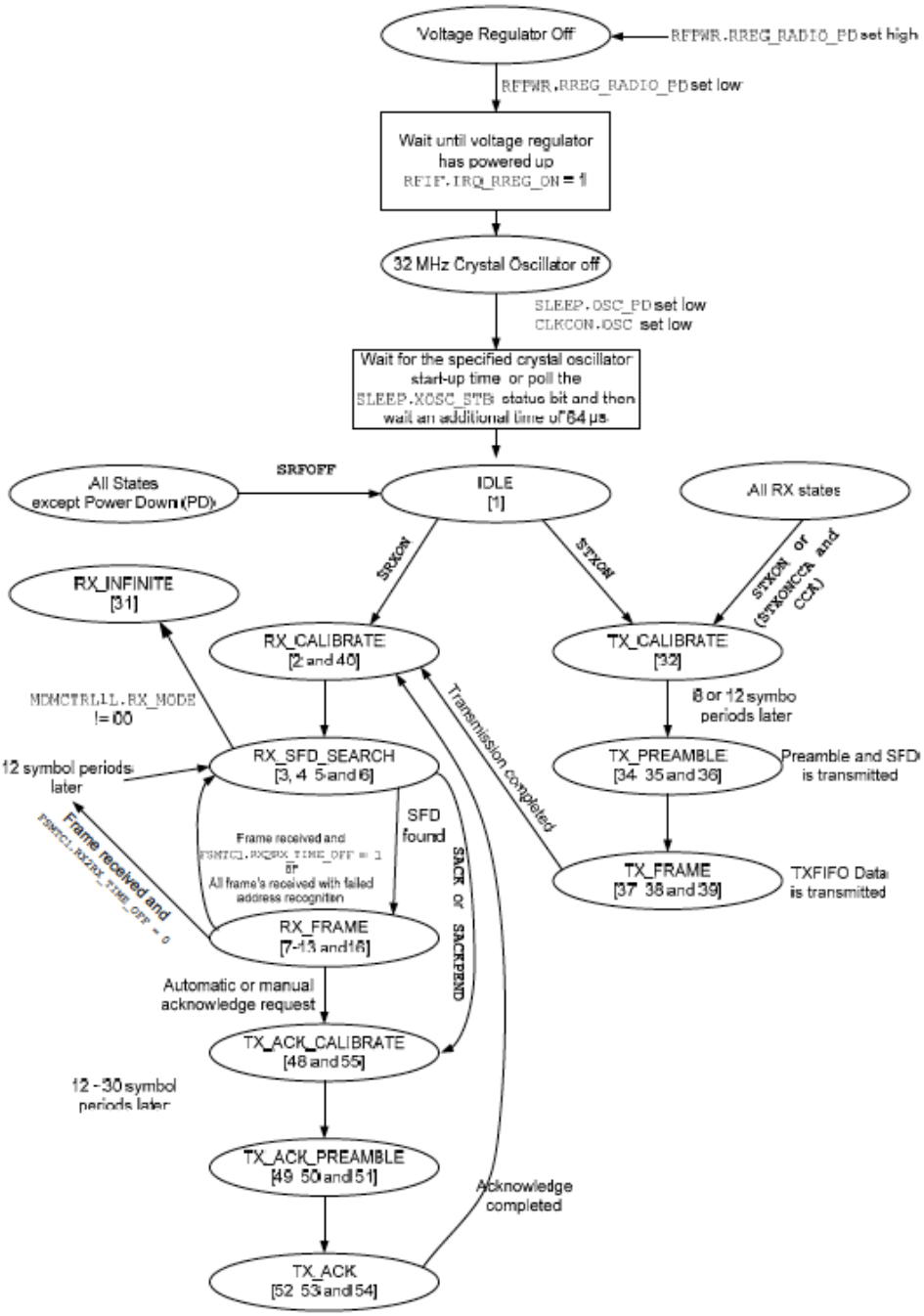


Figure 5.8: Radio Control State Diagram taken from TI CC22430 datasheet [64].

The 802.11 standard specifies the tolerance of the physical oscillator as $\pm 40ppm$. Because of this, we can assume that the drift of the symbol clock will be extremely slow, such that the time period of the symbol clock at sender is equal to the time period of the symbol clock at the receiver. Due to the strict tolerance of the transmit oscillator, we can safely assume that the time periods of the symbols are constant between messages sent in rapid succession, given that the data rate and modulation are the same. However, there exists a phase offset between the symbol clock and the high frequency timestamping clock. Same is the case at the receiver end, where there is a phase offset between the incoming symbol clock and the high frequency timestamping clock. In effect, there exists an unknown phase offset between the timestamping clocks at the sender and the receiver, if the clocks are not synchronized prior to execution of the localization protocol.

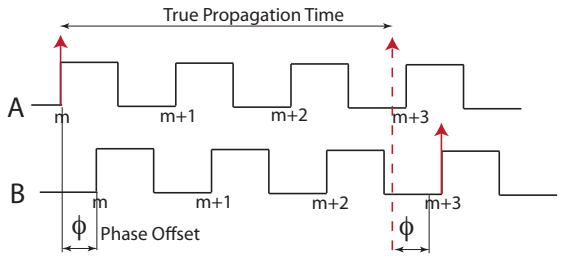


Figure 5.9: Effect of offset between the clocks of the timestamping modules at the sender and receiver.

Suppose an entity B timestamps the arrival of a message sent by entity A to estimate the propagation time of a message between them. Fig. 5.9 illustrates the effect of clock offset on B's measurement for the propagation time. Suppose B's clock has a phase lag of ϕ clock periods with respect to A's clock. If A sends a message at the mth leading edge of its clock, then the timestamp it captures for the departure event of the message

is m. Let the dashed arrow represent the time at which the message actually reaches B. Due to the phase offset, B perceives the arrival time as ϕ clock periods less than the true arrival time. Moreover, the receiver can only detect and timestamp the arrival at a leading edge of its own clock. Hence, timestamps captured at B can only be integer multiples of the clock period. This effect is known as *quantization*. The combined effect of phase offset and quantization cause B to record a timestamp at $m + 3$ clock periods of its own clock, instead of the true time-of-arrival (ToA).

5.3.1 Mitigating Error Due to Clock Offset and Quantizaton

Error due to quantization cannot be avoided. However, the error due to the phase offset can be corrected for, if the timestamping clocks of the sender and the receiver are synchronized. Thorbjornsen et al. [63], introduced a technique for measuring the propagation time of a message between two wireless sensor nodes. The authors showed that when the clocks of two entities have the same time period but a small phase offset, it is possible to obtain ranging measurements with a resolution greater than that allowed by either clock. Their technique requires averaging over a large number of two-way message exchanges.

In the following section, we describe how verifiers in a time-based localization protocol can use this technique to measure propagation times of the challenge and response messages, with a resolution greater than the resolution of the timestamping clock. Further, we also study how the measurement accuracy is affected when Thorbjornsen et al.'s technique is applied with synchronized verifier clocks.

5.4 Effect of Clock Synchonization on the Accuracy of SIMO Localization Protocols

In chapters 2 and 3, we discussed the two subcategories of SIMO localization protocols: (1) Hyperbolic Multilateration (HM) based on the time-difference-of-arrival (TDoA) technique, which is currently used in many applications and (2) Elliptical Multilateration (EM) based on the time-of-arrival (ToA) technique, which we introduced in this work. In chapter 4, we explained that the most dominant factor affecting the accuracy of localization is the placement of the timestamping module relative to the different layers of the network protocol stack. We showed that nanosecond-level accuracy can be achieved only when the timestamps are captured within the PHY hardware, immediately before symbol creation during transmission, and immediately after symbol detection during reception. If timestamps are captured in the manner that we proposed in chapter 4, it is possible to obtain timestamps with an accuracy on the order of nanoseconds.

To limit the error in distance measurements to less than $10m$, we can apply the technique used by Thorbjornsen et al. [63]. The verifiers could execute a large number of challenge-response rounds, and achieve sub-clock accuracy in measuring the message propagation times by simple averaging. In the remaining sections of this chapter, we show how Thorbjornsen's technique works not only with two-way challenge-response echoes (as in their work), but can also be applied to challenge-response relays executed in SIMO localization protocols. We also quantitatively evaluate the error introduced when the verifier clocks are not synchronized.

In chapters 2 and 3, we mentioned that in current implementations of HM, the

verifier clocks are synchronized prior to execution of the challenge-response rounds. In all existing systems however, there is an expensive infrastructure-based mechanism in place for precision synchronization of the verifier clocks. For example, in navigation systems, the devices synchronize to the atomic clocks onboard satellites. In common wireless network scenarios, the verifiers are chosen to be fixed base-station like entities that can be synchronized over a wired ethernet infrastructure. If all the verifiers are mobile and synchronized only over 802.11 message exchanges, synchronization is often "loose" and not as fine-grained as required for limiting the error in localization to the order of one meter. Achieving precision clock synchronization over wireless is very difficult with off the shelf transceivers, using low frequency clocks. It is also difficult in ad-hoc networks where the device clocks must be synchronized soleley in a peer-to-peer manner over the wireless medium. For this reason, Saha and Molle [54] introduced synchronization-free HM in their protocol *Localization with Witnesses*. Although synchronization-free HM is highly desirable in certain scenarios, it is worth asking *"How does the lack of synchronization amongst verifiers affect the accuracy of the localization result in Hyperbolic Multilateration (HM)?"*

In the rest of this chapter, we provide a quantitative analysis of the effect of clock synchronization when Thorbjornsen et al.'s technique is applied to the two types of SIMO localization protocols. We start with the simple case of measuring the one-way propagation delay between two entities. Next, we show how Thorbjornsen et al's technique can also be applied in SIMO localization. We focus our studies on the SIMO class of protocols because their message structure allows for maximum efficiency and accuracy amongst all three classes of time-based localization protocols [44].

5.4.1 Measuring One-Way Propagation Time of a Message

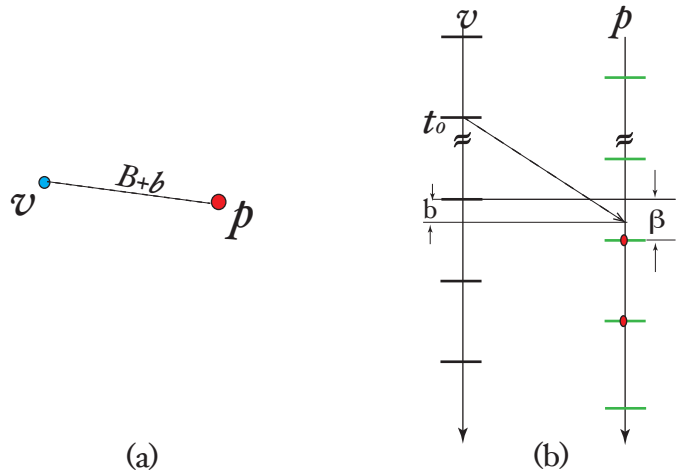


Figure 5.10: Effect of clock offset and quantization on the accuracy of measurements made for one-way message propagation between two entities.

Consider the simple case of one-way propagation of a message from a sender to a receiver. Fig.5.10(a) shows two participants v and p of a time-based ranging protocol. Without loss of generality, we assume that the clock of receiver p has a phase lag of β with respect to sender v's clock, where $\beta \in [0, 1)$. For simplicity, we express the distance $D(v, p)$ between the two entities in terms of the number of clock periods of the timestamping clock. All distances will be denoted in this manner throughout this chapter and the next chapter. Let $D(v, p) \equiv B + b$ clock periods, where B is an integer, and $b \in [0, 1)$ is a fraction, which we will refer to as the *fractional distance*.

Suppose v sends the message at time t_0 according to its own clock. We will denote t_0 as the "start time". The message reaches p after $t_o + B + b$ clock periods, since $D(v, p) \equiv (B + b)$. With digital clocks, events can only be detected synchronous to the leading edges of the clock ticks. In Fig.5.10(b), the leading edges of p's clock marked with the red dots denote the possible time instants at which B detects the arrival of the message

and captures a timestamp for the arrival event. If $b \leq \beta$, then the arrival of the message is detected at the first leading edge following the true arrival time. Instead, if $b > \beta$, then an extra clock period is included in the measurement because the arrival of the message is detected at the second leading edge following the true arrival time of the message. Therefore, the timestamp C_v^p recorded by p, for the arrival of the message is

$$C_v^p = t_o + B + \beta + p_1 \tag{5.5}$$

where

$$p_1 = \begin{cases} 0 & 0 < b \leq \beta \\ 1 & b > \beta \end{cases} \tag{5.6}$$

Depending on the value that the random variable p takes, the number of clock periods measured by p is either B or $B+1$, instead of the true propagation time $B+b$. Therefore, in a *one-way message transfer, the combined effect of the clock offset and quantization causes the measurement (for the propagation time) to take on one of two neighboring integer values.*

If multiple measurements are made keeping the positions of v and p fixed, the value of the fractional distance b is a constant across the measurements, whereas the phase offset between the clocks of the two entities is random for each measurement. If the clocks of the two entities are *not synchronized* prior to making the measurements, then the phase offset is unknown for each measurement. Fig.5.11 shows the probability distribution of the two integer values that the measurements take on, as a function of the phase offset β. From the figure and/or Eq.(5.6), we can compute the expectation $E[p_1 = 1] = b$. Therefore, if p makes enough measurements to obtain a uniform sampling of the phase offset β, it

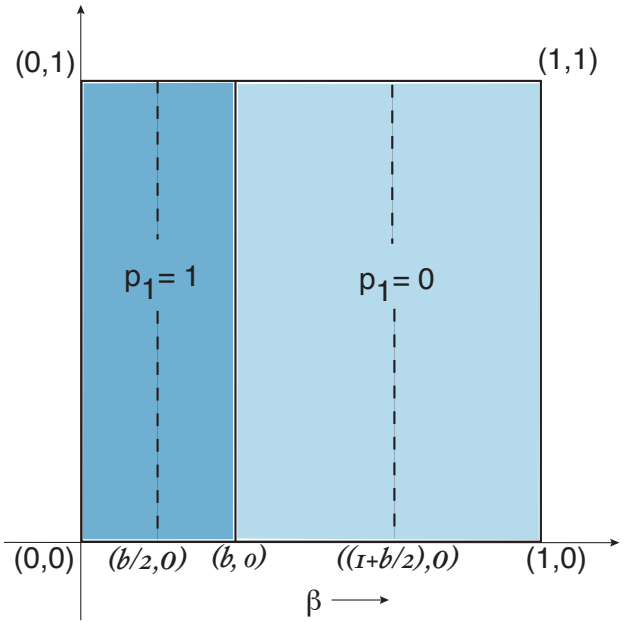


Figure 5.11: Distribution of p_1 depending on the fractional interval b.

can estimate the fractional distance b as follows: While making measurements, p maintains two counters i and j, which are incremented each time the measurement corresponds to $B+1$ and B respectively. After making the measurements, the estimated value of b can be computed as

$$b = \frac{i}{i+j} \cdot 1 + \frac{j}{i+j} \cdot 0 \tag{5.7}$$

As the number of measurements increases, the distribution of the phase offset β obtained from the samples approaches the uniform distribution, therefore the estimated value of b computed from a larger number of samples approaches its true value. In conclusion, in the absence of clock synchronization, the receiver must make a large number of measurements to obtain a good enough estimate of the fractional distance b.

If the clocks of the sender and the receiver *are synchronized*, then the phase offset

is known for every measurement. Compared with the number of measurements required in the absence of synchronization, knowledge of the phase offsets can be used to obtain an estimate of b with the same accuracy, but with fewer measurements. In this case, p maintains two accumulators A_i and A_j in addition to the counters i and j. For each measurement, p increments counter i by one and adds the known value of the phase offset β for that measurement to accumulator A_i if the measurement corresponds to $B+1$. Similarly, it increments counter j and adds the known value of offset to accumulator A_j is the measurement corresponds to B. Let us denote the mean value of β for which the measurements correspond to $B+1$ as $\bar{\beta}_{B+1}$ and the mean value of β for which the measurements correspond to B as $\bar{\beta}_B$. After making all the measurements, the values of $\bar{\beta}_{B+1}$ and $\bar{\beta}_B$, obtained experimentally are

$$\begin{aligned} \bar{\beta}_{B+1} &= A_i/i \\ \bar{\beta}_B &= A_j/j \end{aligned} \tag{5.8}$$

The values for the same variables, computed theoretically from Eq.(5.6), are

$$\begin{aligned} \bar{\beta}_{B+1} &= b/2 \\ \bar{\beta}_B &= (1+b)/2 \end{aligned} \tag{5.9}$$

In the absence of synchronization, p must use the extreme points of the interval along the β-axis i.e., $(0,0)$ and $(1,0)$ in Fig.5.11 to estimate b according to Eq.(5.7). Adding

b to either side of Eq.(5.7), we obtain

$$\begin{aligned} 2b &= \frac{i}{i+j} \cdot 1 + \frac{j}{i+j} \cdot 0 + b \\ 2b &= \frac{(1+b)i}{i+j} + \frac{b \cdot j}{i+j} \\ b &= \frac{1+b}{2} \cdot \frac{i}{i+j} + \frac{b}{2} \cdot \frac{j}{i+j} \end{aligned} \tag{5.10}$$

which shows that in the computation of b, the result does not change if we use the mean values from (5.9) instead of the endpoints of the interval. Recall that the values in (5.8) are experimentally computed values corresponding to the theoretically computed values in (5.9). Therefore, p can substitute the values from (5.8) for the values from (5.9) in Eq.(5.10) to obtain an estimate of b which is even closer to its true value.

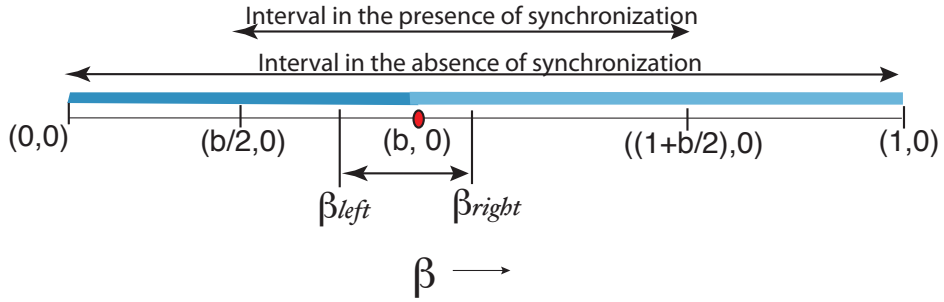


Figure 5.12: When the clocks of the entities are synchronized, the search range for the true value of the fractional distance b can be reduced to half of the search range without synchronization. The search range can be further reduced by bounding it using samples from experimental observation.

An even more accurate estimate of b can be obtained by experimentally bounding the interval in which b lies. Consider Fig.5.12, which shows the end points of the intervals which p must use for the estimation of b when (1) the clocks not synchronized and (2) when

the clocks are synchronized prior to making measurements. The receiver p can find a much tighter interval for b by finding β_{left} – the highest measured value of β which is less than, and β_{right} – the lowest measured value of β which is greater than, the value computed in (5.7). The value of b can then be computed as the mean of β_{left} and β_{right}.

5.4.2 Measuring Round-Trip Time in a Challenge-Response Echo

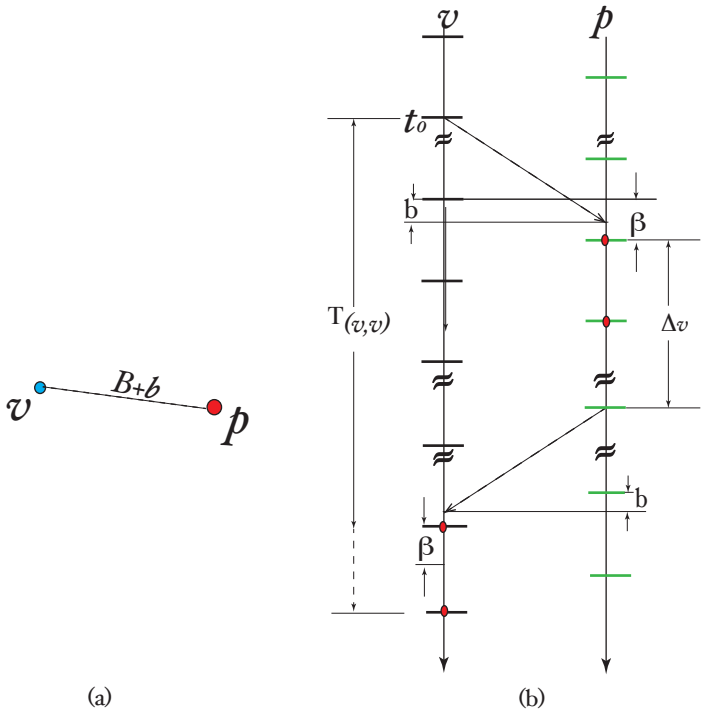


Figure 5.13: Effect of clock offset and quantization on the accuracy of measurements made for the propagation time of a challenge-reponse echo.

Let us consider the case where the sender (verifier v) measures the total time for a two-way message exchange (of the form of a challenge-response echo) between itself and a receiver (prover p). Suppose v sends the message at time t_0, and captures a timestamp $C_v^v = t_o$ for the departure event. As in the case of a one-way message, the timestamp that receiver p records for the arrival event is given by Eq.(5.5). p sends the response after a

delay of Δ_v to reach v at time $t_0 + B + \beta + p_1 + \Delta_v + B + b$ as shown in Fig.5.13. Therefore, the timestamp captured by v for the arrival of the response is

$$C_p^v = t_0 + B + p_1 + \Delta_v + B + v_1 \tag{5.11}$$

where

$$v_1 = \begin{cases} 0 & \beta \leq 1 - b \\ 1 & 1 - b \leq \beta \end{cases} \tag{5.12}$$

Notice that both the departure and arrival events are timestamped by the same entity in this case, therefore the phase offset term β is subtracted away in Eq.(5.11). However, the phase offset *does* have an effect on the measurements because the values of the random variables p_1 and v_1 depend on β. From the timestamps for the arrival and departure events, v can compute the interval $T_p(v, v)$. Assuming that the exact value of the response delay Δ_v is known, the two-way propagation time of the message is $T_p(v, v) - \Delta_v$.

Since the random variables p_1 and v_1 can each take on the values in $\{0, 1\}$, their sum $p_1 + v_1$ can take on one of the three distinct integer values in $\{0, 1, 2\}$. Fig. 5.14 shows the distribution of $p_1 + v_1$ as a function of the phase offset β and the fractional distance b. From the figure, we find that for a fixed value of b, the sum $p_1 + v_1$ can only take on at most two of the three possible integers, i.e, $p_1 + v_1$ can either take on values in $\{0, 1\}$ or in $\{1, 2\}$, except in the special case of $b = 0.5$, when $p_1 + v_1$ can only take on a single value i.e., 1. In Fig. 5.14, it is easy to recognize this, because the dashed line corresponding to $b = y$ passes through regions of at most two distinct colors, except at $y = 0.5$ where the complete line is contained in a single-colored region. *Therefore, in a challenge-response*

relay, the combined effect of clock offset and quantization causes the measurements (for the round-trip propagation time) to take on at most one of two neighboring integer values.

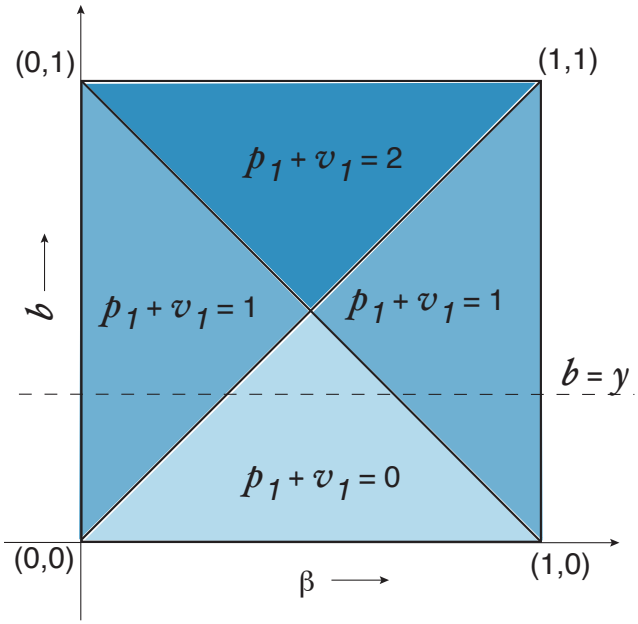


Figure 5.14: Distribution of the values of $p_1 + v_1$ depending on the fractional interval b.

The true distance being $B + b$, the measured values of $T_p(v, v) - \Delta_v$ are

$$Tp(v,v) - \Delta_v \in \begin{cases} \{2 \cdot B + 1, 2 \cdot B + 2\} & b < 0.5 \\ \{2 \cdot B + 2\} & b = 0.5 \\ \{2 \cdot B + 2, 2 \cdot B + 3\} & b > 0.5 \end{cases} \tag{5.13}$$

Let us define x_1 as the probability of occurrence of $2 \cdot B + 2$ in the measurements:

$$x_1 = P(Tp(v,v) - \Delta_v = 2 \cdot B + 2) \tag{5.14}$$

except where $P(Tp(v,v) - \Delta_v \neq 2 \cdot B + 2) \approx 0$, in which case $x_1 = 1$. From (5.13) we find

that the smaller of the two integers in the measurements is odd if $b < 0.5$, and its value is even when $b > 0.5$. It is possible that measurement error may add a clock period to the values in the right hand side of expression (5.13). In that case however, three values for $T_p(v, v) - \Delta_v$ will appear in the measurements, where the largest value will be the erroneous value. Knowing this, we can sanitize the data and compute the probabilities of the two legitimate values. The fractional interval b can then be computed as

$$b = \begin{cases} x_1/2 & b < 0.5 \\ 1 - x_1/2 & b > 0.5 \\ 0.5 & otherwise \end{cases} \tag{5.15}$$

As the number of measurements increases, the distribution of the phase offset β in the sample approaches a uniform distribution, and the computed value of x_1 approaches its true value. When a challenge-response echo is executed for the purpose of secure localization/secure location verification, the prover cannot be trusted to provide correct information about the phase offset of its own clock. We cannot assume that the phase offset of the prover is known to the verifiers. Moreover, b being an unknown, the error in an individual measurement cannot be computed, since it is function of both β and b. Hence accuracy of the measurements can only be improved by computing the solution over a large number of measurements when two-way challenge-response echoes are executed between a verifier and the prover.

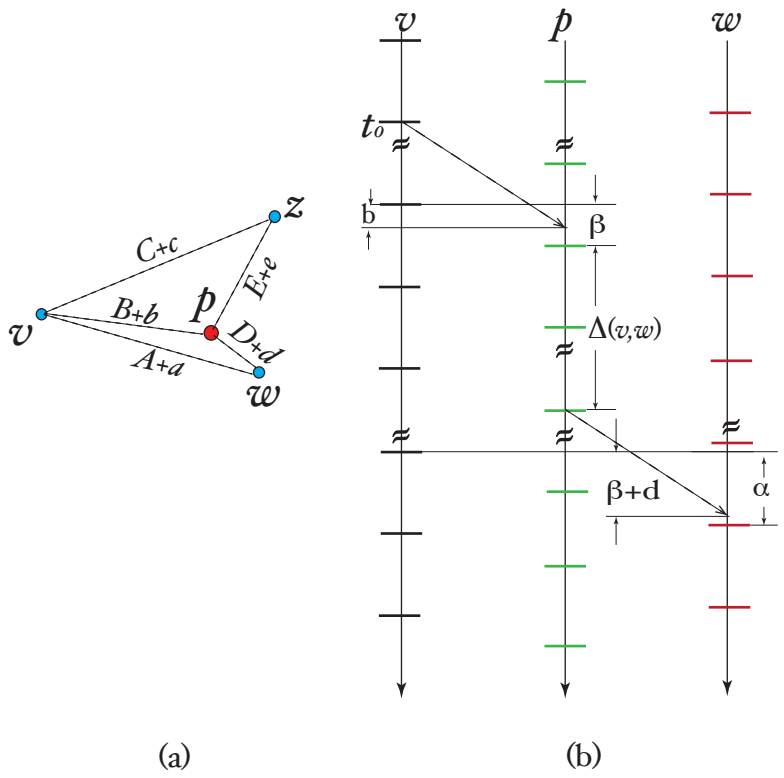


Figure 5.15: Timestamping a two-hop message propagation between the sender v and the receiver w

5.4.3 Measuring Two-Hop Propagation Time in a Challenge-Response Relay

Next, we discuss the effect of clock offset and quantization in the case of a two-hop challenge-response relay. Fig.5.15 shows four entities and the distances between them in terms of the common clock period. A message sent by v (challenge) in the first hop reaches the receiver p (prover) after $t_o + B + b$ clock periods. The timestamp at p is given by Eq.(5.5). The message is also received by other verifiers w and u, whose clocks have lags of α and γ clock periods with respect to v's clock. The timestamp captured by w for the arrival of the challenge is

$$C_v^w = t_o + A + \alpha + w_1 \tag{5.16}$$

where

$$w_1 = \begin{cases} 0 & 0 < a \leq \alpha \\ 1 & a > \alpha \end{cases} \tag{5.17}$$

Similarly, the timestamp recorded by the other witness u is

$$C_v^u = t_o + C + \gamma + u_1 \tag{5.18}$$

where

$$u_1 = \begin{cases} 0 & 0 < c \leq \gamma \\ 1 & c > \gamma \end{cases} \tag{5.19}$$

We assume that p uses a common response delay $\Delta_v = \Delta_{(v,w)} = \Delta_{(v,u)}$ for all the entities. The response sent by p reaches a witness w at

$$C_p^w = t_0 + B + \beta + p_1 + \Delta_{(v,w)} + D + d \tag{5.20}$$

However, due to its phase offset with respect to p, w records the timestamp at

$$C_p^w = t_o + B + p_1 + \Delta_{(v,w)} + D + \alpha + w_2 \tag{5.21}$$

where

$$w_2 = \begin{cases} 0 & \beta \leq \alpha - d \\ 1 & \alpha - d < \beta < 1 + \alpha - d \\ 2 & 1 + \alpha - d < \beta \end{cases} \tag{5.22}$$

Similarly, the timestamp recorded by a different witness, u in this case, is given by

$$C_p^u = t_o + B + p_1 + \Delta p + C + \gamma + u_2 \tag{5.23}$$

where

$$u_2 = \begin{cases} 0 & \beta \leq \gamma - e \\ 1 & \gamma - e < \beta < 1 + \gamma - e \\ 2 & 1 + \gamma - e < \beta \end{cases} \tag{5.24}$$

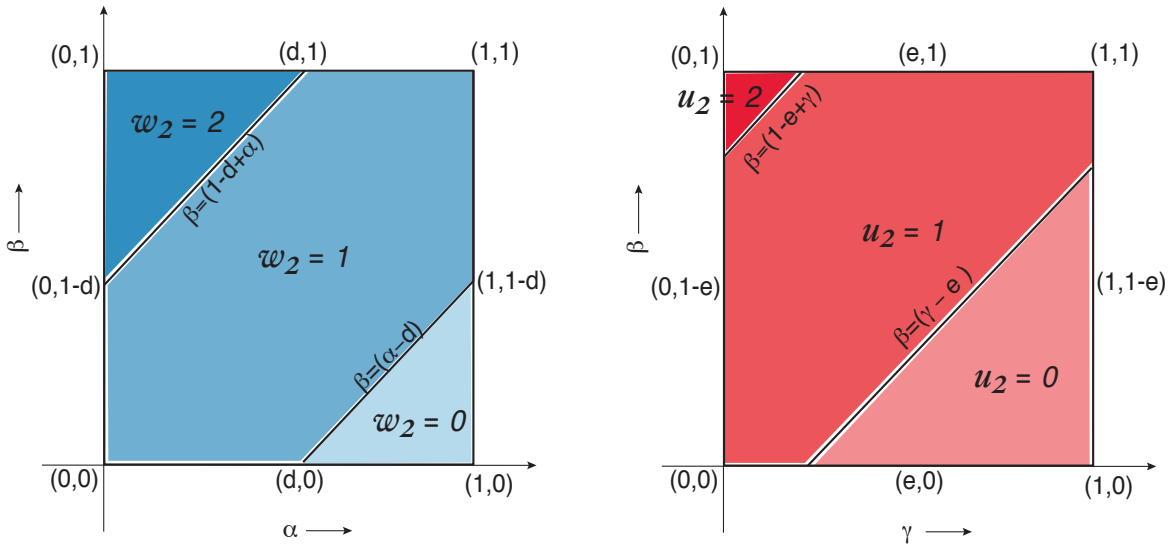


Figure 5.16: Random variables w_2 and u_2 expressed as functions of the phase offsets and fractional distances

In the two known SIMO protocols, EM and HM, while the lead verifier v measures the interval $T_p(v, v)$ as described in section 5.4.2, each passive verifier, say w measures the interval $T_p(v, w) = C_p^w - t_0$ to estimate the distance travelled by the challenge-response relay. Fig.5.16 shows the distributions of the random variables w_2 and u_2 as functions of the phase offsets and the fractional distances between the respective passive verifiers

and p. We find that *in a challenge-response relay, the combined effect of clock offset and quantization causes the measurements (for the two-hop propagation time) to take on one of three neighboring integer values.*

In the subsections that follow, we quantify the error due to clock offset and quantization in both EM and HM. For each protocol, we consider two cases - (i) when the clocks of the verifiers are not synchronized and (ii) when the clocks of the verifiers are synchronized prior to executing the challenge-response message exchanges. We demonstrate through analysis and simulations that in the presence of clock synchronization, the distances along the two-hop paths can be estimated with greater accuracy. We also show that when the exact value of the "start time" is not known to a passive verifier due to the absence of synchronization, the accuracy of the result is significantly lower.

5.4.4 Elliptical Multilateration Without Clock Synchronization

To compute the distance $D_p(v, w)$ from the lead verifier to itself via the prover, a witness w must know the start time t_0. When clock synchronization is not available in the network, w does not know the timing relationship between its own timestamping clock and that of the verifier's. Therefore, even if the lead verifier shares its timestamp for the start time, w cannot tell the exact time at which the lead verifier sent the challenge. It must *estimate* the start time using its own timestamp for the arrival of the challenge, and the known distance between itself and the lead verifier v. The start time, as estimated by w is

$$t_{0\{w\}} = C_v^w - (A + a) \tag{5.25}$$

Substituting from Eq.(5.16), w's estimate of the start time in terms of v's clock is

$$\begin{aligned} t_{0\{w\}} &= t_o + A + \alpha + w_1 - (A + a) \\ &= t_o + \alpha + w_1 - a \end{aligned} \tag{5.26}$$

Therefore, w's estimate for the start time has an error of

$$err_{t_o} = \alpha + w_1 - a \tag{5.27}$$

Since α is a $(0, 1]$ uniform random variable, its expectation, $E[\alpha] = 0.5$. The expectation of the random variable w_1, $E[w_1]$, is

$$\begin{aligned} E[w_1] &= 0 \cdot P(a < \alpha) + 1 \cdot P(a > \alpha) \\ &= a \end{aligned} \tag{5.28}$$

Substituting the expected values of α and w_1 in Eq.(5.27), the expected error

$$E[err_{t_0}] = 0.5 + a - a = 0.5 \tag{5.29}$$

Therefore, *in the absence of precision clock synchronization amongst the verifiers, the estimated start time of EM, as computed by a witness from its measurements, has a mean error of half a clock period.*

After estimating the start time, the witness forms an elliptical constraint on the prover's location by computing the path length along $v \rightarrow p \rightarrow w$. For example, witness

w, uses its timestamp, C_p^w, for the arrival of the response and the estimated start time to compute $T_p(v,w)$

$$\begin{aligned} T_p(v,w) &= C_p^w - t_{0_w} \\ &= B + p_1 + \Delta_{v,w} + D + w_2 - w_1 + a \end{aligned} \tag{5.30}$$

In addition to the error in the estimated start time err_{t_0}, error is also introduced into the measurement due to the terms p_1, w_1 and w_2 which are introduced due to the combined effect of clock offset and quantization. From Fig. 5.15(b), we observe that

$$\begin{aligned} D_p(v,w) + \Delta(v,w) &= D(v,p) + \Delta_{\{v,w\}} + D(p,w) \\ &= B + b + \Delta_{\{v,w\}} + D + d \end{aligned} \tag{5.31}$$

Subtracting Eq(5.31) from Eq.(5.30), gives us the error in the estimated path length

$$err_D = (p_1 + w_2 - w_1 + a) - (b + d) \tag{5.32}$$

Similar to Eq.(5.28), the expectation of variable p, $E[p]$, is given by

$$\begin{aligned} E[p_1] &= 0 \cdot P(p_1 < \beta) + 1 \cdot P(p_1 > \beta) \\ &= b \end{aligned} \tag{5.33}$$

From Fig. 5.16, we can also calculate the expectation $E[w_2]$ as

$$\begin{aligned} E[w_2] &= 0 \cdot A(w_2 = 0) + 1 \cdot A(w_2 = 1) + 2 \cdot A(w_2 = 2) \\ &= 1 \cdot (1 - 0.5(1-d)^2 - 0.5 \cdot d^2) + 2 \cdot (0.5 \cdot d^2) \\ &= d + 0.5 \end{aligned} \tag{5.34}$$

Since the expectation of a sum is the sum of expectations, we can plug in the expected values of p, w_1 and w_2 from Eqs. (5.33), (5.34) and (5.28), to obtain the expectation $E[err_D]$ of the error in the estimated path length.

$$\begin{aligned} E[err_D] &= E[p_1] + E[w_2] - E[w_1] + E[a - (b + d)] \\ &= b + d + 0.5 - a + a - b + d \\ &= 0.5 \end{aligned} \tag{5.35}$$

Therefore, *when Time-of-Arrival EM protocol is executed in the absence of clock synchronization, the individual path length estimates have an error of half a clock period on average. To increase the accuracy of localization, the witness must subtract this value from the mean path length computed over multiple challenge-response rounds, before forming the elliptical constraint.*

Since the response time of p is orders of magnitude greater than the propagation time of the messages, amongst all terms in Eq.(5.30), the response time $\Delta_{\{v,w\}}$ has the greatest magnitude. Therefore, the highest uncertainty in the path length measurement can be attributed to $\Delta_{\{v,w\}}$. SIMO multilateration protocols are designed to detect cheating if

the prover does not adhere to the known value of $\Delta_{\{v,w\}}$. If the prover is honest and adheres to the known value of $\Delta_{\{v,w\}}$, then the accuracy is affected only by the measurement errors for packet arrival events, and making the appropriate correction increases the accuracy of the result.

5.4.5 Elliptical Multilateration With Precision Clock Synchronization

When the verifiers' timestamping clocks are synchronized, the lead verifier v can share with the witnesses, the exact time at which it sent the challenge. If the message exchange structure for the clock synchronization is similar to that in IEEE 1588 PTP [23], then v can capture a timestamp within the transceiver hardware, whose value is close to the true transmission time. It can then share this timestamp for the start time with the witness by sending it in a followup message. Better still, if synchronization is supported by a mechanism similar to the precision PHYter [41], then the exact sending time can be written into the payload of the challenge itself, which avoids the need for a "followup" message transmission.

When the clocks of the verifiers are synchronized, a witness not only knows the exact start time t_0, but also knows the phase offset of its own clock with respect to the lead verifier v's clock. Knowledge of the phase offset at the instant when the timestamp is captured, allows a witness to apply appropriate corrections to the measurements. The timestamp captured by witness w for the arrival of the response is given by Eq.(5.21). Since the exact start time of the protocol is known in this case, the witness can subtract t_0 from C_p^w, instead of subtracting an estimate of t_0. The error introduced due to the estimation of the start time given by Eq.(5.27) is expected to be eliminated from the individual path

length estimated in this case.

$$\begin{aligned} T_p(v,w) &= C_p^w - t_o \\ &= B + p_1 + \Delta_{\{v,w\}} + D + \alpha + w_2 \end{aligned} \tag{5.36}$$

is the expression for time interval measured by w. The error in the measurement can be obtained by subtracting Eq.(5.31) from Eq.(5.36)

$$\begin{aligned} err_D &= (B + p_1 + \Delta_{\{v,w\}} + D + \alpha + w_2) \\ &\quad -(B + b + \Delta_{\{v,w\}} + D + d) \\ &= p_1 + \alpha + w_2 - b - d \end{aligned} \tag{5.37}$$

Plugging in the expected values of p_1, α and w_2 in Eq.(5.37), we find the expectation of the error to be

$$\begin{aligned} E[err_D] &= E[p_1] + E[\alpha] + E[w_2] - E[b + d] \\ &= b + 0.5 + d + 0.5 - b - d \\ &= 1 \end{aligned} \tag{5.38}$$

Therefore, *if the start time is known and no correction is applied for effect of phase offset, the witness must subtract one clock period from the mean path length estimate before forming the elliptical constraint on the prover's location.*

When the start time is known, the error in the measurements can be attributed to p_1, α and w_2, as seen in Eq.(5.37). Of the these three terms, α and w_2 are dependent on

the known phase offset α. To examine the contribution of α to the error introduced by w_2, we uncondition w_2 on β, the unknown phase of the prover. From (5.22), we can rewrite w_2 as

$$w_2 = f(\alpha, \beta, d) = \begin{cases} k-1 & \beta \leq \alpha - d \\ k & \alpha - d < \beta < 1 + \alpha - d \\ k+1 & 1 + \alpha - d < \beta \end{cases} \tag{5.39}$$

where k assumes a default value of 1. Next we uncondition over β.

$$\begin{aligned} \alpha \leq d : f(\alpha, d) &= \int_0^{1-d+\alpha} k \cdot d\beta + \int_{1-d+\alpha}^{1} (k+1) d\beta \\ &= k \cdot (1 - d + \alpha) + (k+1)(1 - (1 - d + \alpha)) \\ &= k + d - \alpha \end{aligned}$$

$$\begin{aligned} \alpha > d : f(\alpha, d) &= \int_0^{\alpha-d} (k-1) d\beta + \int_{\alpha-d}^{1} k \cdot d\beta \\ &= (k-1)(\alpha - d) + k \cdot (1 - (\alpha - d)) \\ &= k + d - \alpha \end{aligned} \tag{5.40}$$

Since $f(\alpha, \beta, d)$ unconditioned over β is a function of α, the individual estimates, even when sampled uniformly across the unknown phase offset of the prover p, have a bias proportional to the known phase offset of the witness. Let us define a new function $g(\alpha, \beta, d)$ which

includes a correction based on α as follows:

$$g(\alpha,\beta,d) = \begin{cases} \alpha - 1 & \beta \leq \alpha - d \\ \alpha & \alpha - d < \beta < 1 + \alpha - d \\ \alpha + 1 & 1 + \alpha - d < \beta \end{cases} \tag{5.41}$$

If we uncondition $g(\alpha,\beta,d)$ over β, we obtain

$$\begin{aligned} \alpha \leq d : g(\alpha,d) &= \int_0^{1-d+\alpha} \alpha \cdot d\beta + \int_{1-d+\alpha}^{1} (\alpha+1) d\beta \\ &= \alpha \cdot (1 - d + \alpha) + (\alpha + 1)(1 - (1 - d + \alpha)) \\ &= \alpha + d - \alpha \\ &= d \end{aligned} \tag{5.42}$$

$$\begin{aligned} \alpha > d : g(\alpha,d) &= \int_0^{\alpha-d} (\alpha - 1) d\beta + \int_{\alpha-d}^{1} \alpha \cdot d\beta \\ &= (\alpha - 1)(\alpha - d) + \alpha \cdot (1 - (\alpha - d)) \\ &= \alpha + d - \alpha \\ &= d \end{aligned} \tag{5.43}$$

We find that subtracting $(1 - \alpha)$ from w_2 makes the individual estimates unbiased. Since the term α is already present in Eq.(5.37), this subtraction equates to reducing the error by one clock period in each individual measurement. With this subtraction made, the expected

error is

$$\begin{aligned} E[err_D] &= E[p_1] + E[\alpha] + E[w_2] - E[b+d] - E[1] \\ &= b + 0.5 + d + 0.5 - b - d - 1 \\ &= 0 \end{aligned} \tag{5.44}$$

Therefore, if we sample uniformly across the unknown phase offset β of the prover p, and make the appropriate correction for the effect of the phase offset of the witness, then the expected error in the measurement is zero.

5.4.6 Hyperbolic Multilateration Without Clock Synchronization

Unlike in EM, where the estimate of $T_p(v, w)$ allows witness w to form an elliptical constraint on the prover's location, the hyperbolic constraints in HM are formed by pairwise combination the measurements made by the verifiers (witnesses). In particular, to form a single hyperbolic constraint on the prover's location, the estimated value of $T_p(v, u)$ computed by witness u is subtracted from the estimated value of $T_p(v, w)$ computed by witness w, to form a difference equation, which is the equation of a hyperbola.

The witness w estimates the time taken along the path $v \to p \to w$ by subtracting Eq.(5.16) from Eq.(5.21) and adding the known distance between itself and the lead verifier v, i.e., $(A + a)$. Thus the estimate is given by

$$\begin{aligned} T_p(v, w) &= C_p^w - t_{0_w} \\ &= B + p_1 + \Delta_{\{v,w\}} + D + w_2 - w_1 + a \end{aligned} \tag{5.45}$$

The error in w's estimate for the time along the challenge-response relay path, given the true path length in Eq.(5.31), is

$$\begin{aligned} err_D &= (B + p_1 + \Delta_{\{v,w\}} + D + w_2 - w_1 + a) \\ &\quad -(B + b + \Delta_{\{v,w\}} + D + d) \\ &= (p + w_2 - w_1 + a) - (b + d) \end{aligned} \tag{5.46}$$

This expression is the same as that in Eq.(5.32). Therefore, when averaged over multiple measurements, the mean error in the estimate of $T_p(v, w)$ is 0.5 clock periods as derived in Eq.(5.35). Although the error in the estimate of a single witness in HM is the same as in the case of EM without clock synchronization, we proceed to show how this error cancels out when two witnesses combine their measurements to form a hyperbolic constraint on the location of the prover. Similar to Eq.(5.45), u's estimate of the time along the challenge-response relay path is

$$\begin{aligned} T_p(v, u) &= C_p^u - t_{0_u} \\ &= B + p_1 + \Delta_{\{v,u\}} + E + u_2 - u_1 + c \end{aligned} \tag{5.47}$$

and similar to Eq.(5.46), u's estimate for the path length averaged over multiple measurements is

$$err_E = (p_1 + u_2 - u_1 + c) - (b + e) \tag{5.48}$$

and also has an error of 0.5 clock periods. Therefore, when the witnesses form a difference

equation by subtracting the respective path length estimates, the error cancels out. Recall that in EM, a correction of 0.5 time periods needs to be applied before forming the elliptical constraint. No such correction is necessary in the case of HM.

5.4.7 Hyperbolic Multilateration With Precision Clock Synchronization

As described earlier, in the presence of clock synchronization, the witnesses know the exact value of the start time, and the phase offsets of their own clocks relative to that of the lead verifier v's clock for each measurement. In TDoA, knowing the exact value of the start time is not important, because the term t_0 is common to the measurements of all witnesses and vanishes in the difference equation. The relative phase offsets however, do effect the measurements. The expressions in Eqs. (5.45) and (5.47) have the terms w_1, w_2, u_1, u_2 and p_1, which in turn are functions of the phase offsets α, β and γ. Knowing the phase offset for each measurement allows the witness to calculate a suitable correction, which when applied to each measurement, leads to better accuracy in the final estimate.

As computed in section 5.4.5, if w applies a correction of $(1-\alpha)$, and u applies a correction of $(1-\gamma)$ to individual measurements, they can correct for the error due to the phase offsets (α and γ) of their own clocks with respect to v's clock. Upon applying the correction, the error in each measurement made by a witness, say w, is

$$err'_D \quad = \quad (p_1 + w_2 - w_1 + a) - (b + e) - (1 - \alpha) \tag{5.49}$$

Substituting the expected values of p_1, w_2, w_1 and α as calculated in sections 5.4.4 and

5.4.5, the expected error in the estimate is

$$\begin{aligned} E[err'_D] &= E[p_1] + E[w_2] - E[w_1] + a - b - d - 1 + E[\alpha] \\ &= b + d + 0.5 - a + a - b + d - 1 + 0.5 \\ &= 0 \end{aligned} \tag{5.50}$$

Similarly if u makes the appropriate correction by subtracting $(1-\gamma)$, the expected error in its estimate is also negligible. Therefore the availability of precision clock synchronization during the execution of HM allows the witnesses to correct for the effect of clock phase offsets. This would allow the witnesses to compute the hyperbolic constraint on p's location with the same accuracy as in section 5.4.6 in fewer rounds.

.

5.5 Simulation Results

To verify the analytical results that we obtained in sections 5.4.4 through 5.4.7, we simulated the execution of both Elliptical Multilateration (EM) and Hyperbolic Multilateration (HM). The code was written in Pyhton. For each protocol, the simulations were done for two different cases: (i) when the verifier clocks are not synchronized prior to protocol execution (ii) when the verifier clocks are synchronized prior to protocol execution. By using the same dataset for both protocols, we were able to quantitatively compare their performance in both cases.

For our simulations, we used a setup with three verifiers $\{v, w, u\} \in V$ and a single resource-constrained prover p. Fig. 5.15(a) shows how we denote the distances between the

participating entities. Each unique permutation of the tuple $\{A+a, B+b, C+c, D+d, E+e\}$ is assumed to represent a unique spatial "placement" of the participating entities. To repeat the experiments for different "placements" of the participating entities, each of the the five values in the tuple is randomly set to a real number in $(0, 100)$[3].

Although the experiments were repeated for many different "placements", it is not possible to include all the results in this dissertation. We have confirmed through extensive simulations with a wide variation of the parameters, that our observations hold irrespective of the "placement" of the participating entities. We present the plots for a randomly chosen "placement" of the participating entities. The representative tuple for which we have presented the results is $\{10.50, 75.29, 43.77, 82.63, 25.21\}$.

For each round of challenge-response, the phase offsets α, β and γ were independently sampled from a uniform random $[0, 1)$distribution. A verifier is picked randomly to act as the lead verifier for the current challenge-response round. The lead verifier sends the challenge and records the running time of the challenge-response echo that is executes with the prover. The other two verifiers observe simultaneously observe the challenge response relay. Assuming v is chosen as the lead verifier for some round, the values of α, β, γ, p_1, w_1, u_1, w_2 and u_2 are added to the dataset for that round. If a different verifier is picked as the lead verifier, then corresponding values of the relevant variables are added to the dataset for that round. The prover's response $\Delta_v = \Delta_{\{v,w\}}$ is set to an integer constant. For each placement, we execute the protocol 100 times, therefore collect 100 datasets. Each execution consists of 100 challenge-response rounds, therefore each dataset consists of 100 entries

[3]This range was picked because commercially available 802.11b/g/n transceivers have a transmission range of $50 - 100m$. The transmission range of 802.11a is generally lower at $20 - 30m$. Our results however, hold despite the receiver architecture, because only the fractional parts of the tuple affect the results. The result is independent of the value of the integer parts of the distances between the participants.

for each of the variables mentioned above. The purpose of collecting so many measurements was to compute confidence intervals for our results.

Fig. 5.21(a) shows a plot of the mean error in the estimated two-hop path length as a function of the number of challenge-response rounds, when EM is executed without clock synchronization. The error is expressed in terms of clock periods of the timestamping clock. Each curve in the plot corresponds to one complete execution of the protocol over a total of 100 challenge-response rounds. As expected, the mean error in the distance measurement decreases non-linearly as the number of challenge- response rounds increases. Fig. 5.21(b) shows the corresponding plot when EM is executed with clock synchronization. Notice that along any curve, the values are more closely spaced between subsequent challenge-response rounds. This is because of the correction made depending on the known value of the phase offset. Notice that in both the plots the mean error is shown to converge to 0 over a large number of challenge-response rounds. Without clock synchronization, this only happens when the bias in the original measurements is corrected. If this correction is not done, then the mean error converges to 0.5 clock periods. With clock synchronization, there is no bias. This further validates that EM executed without clock synchronization leads to lower accuracy than EM executed with clock synchronization, such that the difference in mean error is 0.5 clock periods.

Similar trends are observed for HM executed without and with clock synchronization. This is shown in Figs. 5.22(a) and 5.22(b). However, in the case of HM, the mean error always converges to 0 over a large number of challenge-response rounds irrespective of synchronization. This is because the start time need not be estimated in HM even if there is no clock synchronization. The start time is common for both witnesses that form the

difference equation and is subtracted away in the final equation.

Fig. 5.23 shows the confidence intervals for the mean error in both EM and HM. In either case, the confidence interval reduces to one clock period for twenty of more challenge-response rounds, when the clocks are not synchronized prior to protocol execution. We also find that the gap between the confidence intervals with and without synchronization is greater in the case of EM. This is because of the error in the start time estimate in the case of EM. Overall the confidence intervals in the case of TDoA HM are slightly larger than in the case of ToA EM, although the individual measurements in HM are more accurate. We attribute this to the the following reason: In EM an elliptical constraint is formed using the measurement of a single witness. However, in the case of HM, the hyperbolic constraint is formed from the difference of the measurements made by two witnesses. Since the error in the measurements of two witnesses combined can sometimes be greater than the error in the measurement of a single witness, the confidence interval in HM is slightly wider than the confidence interval in EM.

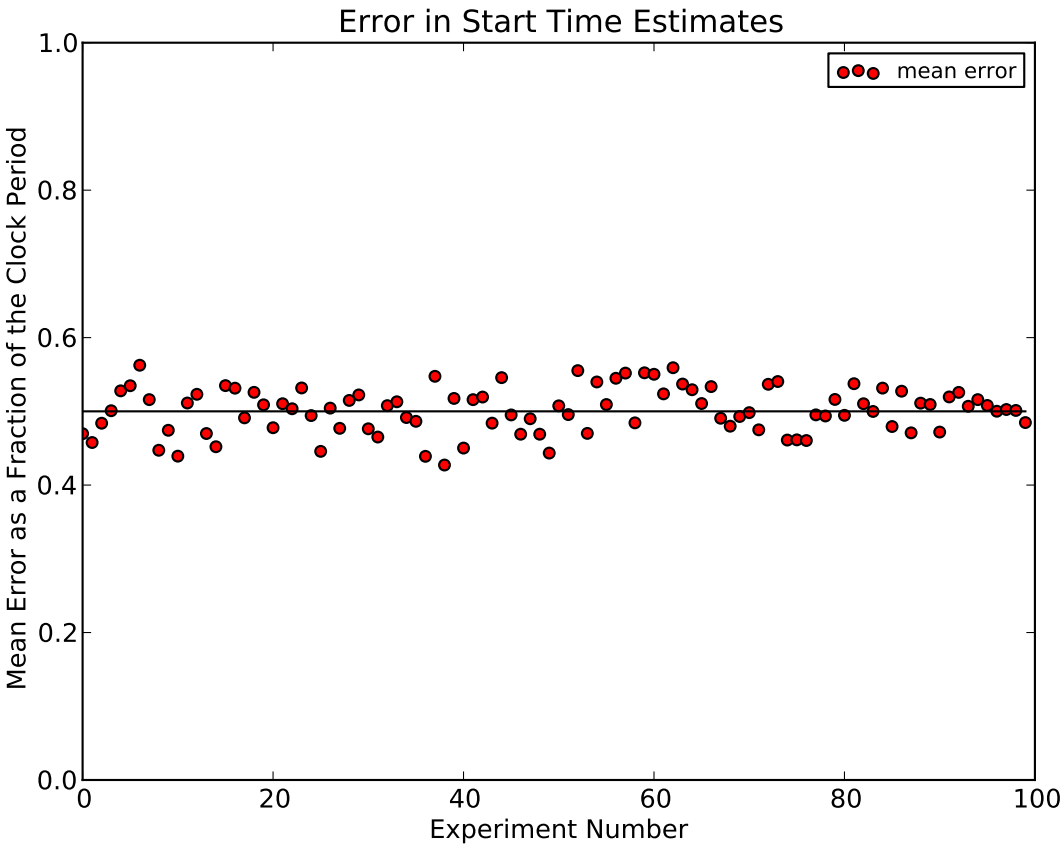
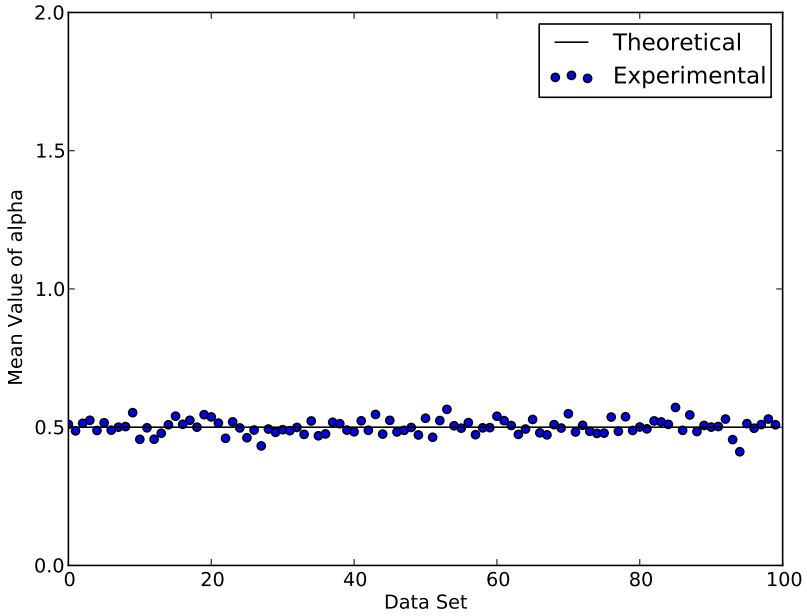
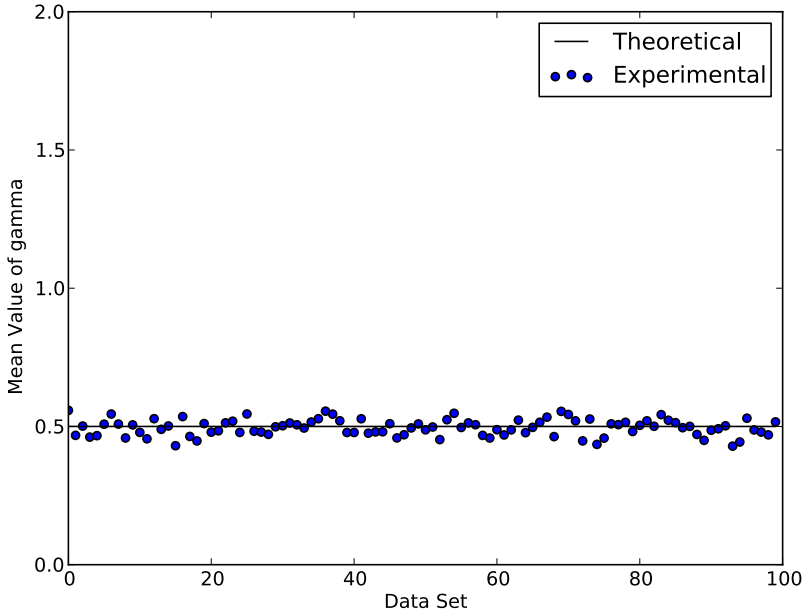


Figure 5.17: Error in the estimate for the start time in EM

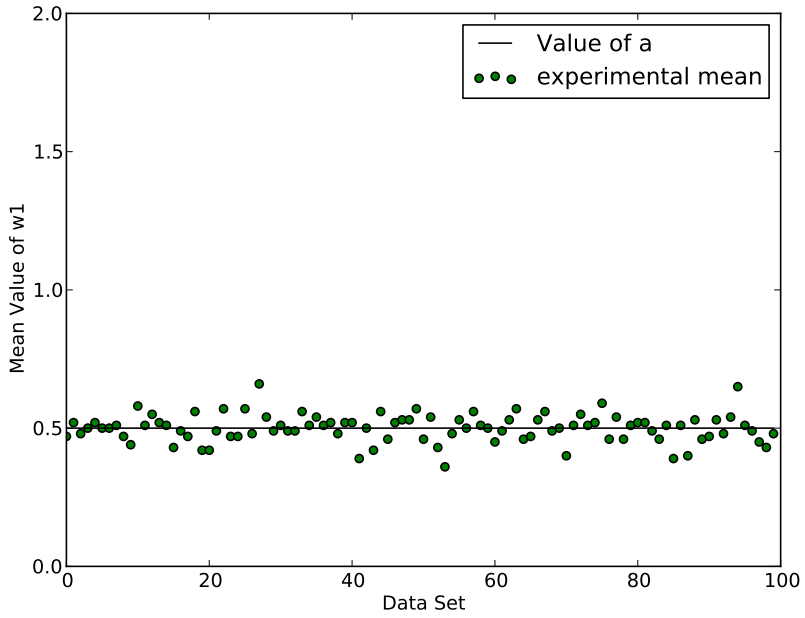


(a) Mean value of α

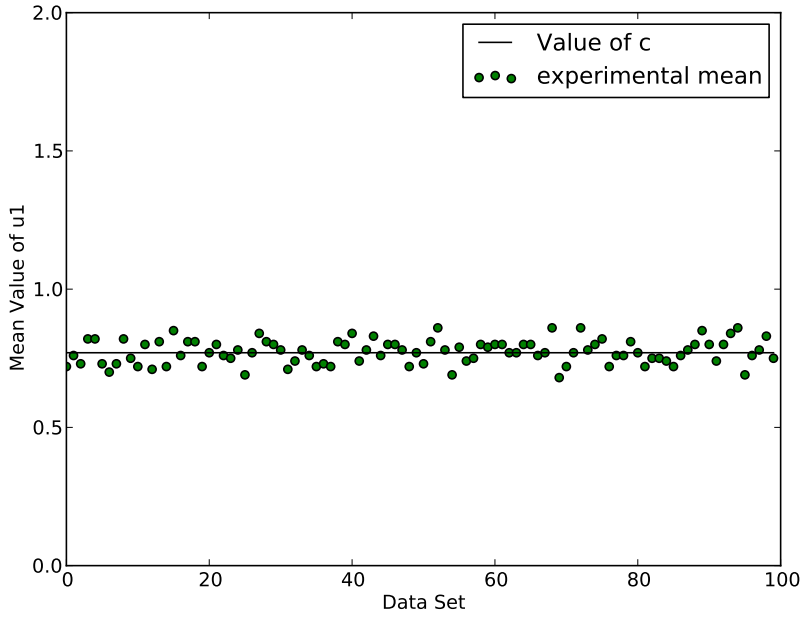


(b) Mean value of γ

Figure 5.18: Comparing the theoretical expectation and experimentally computed mean of the verifiers' clock phase offsets.

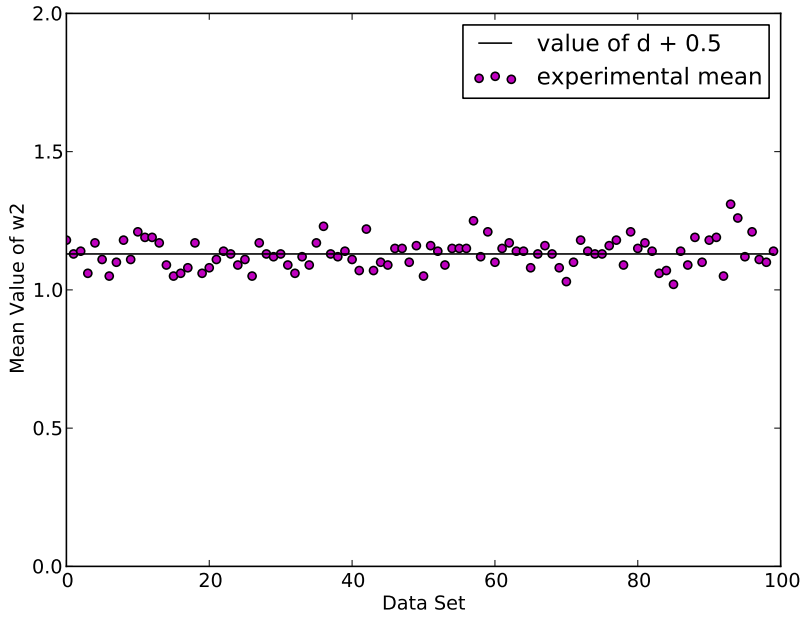


(a) Mean value of w_1

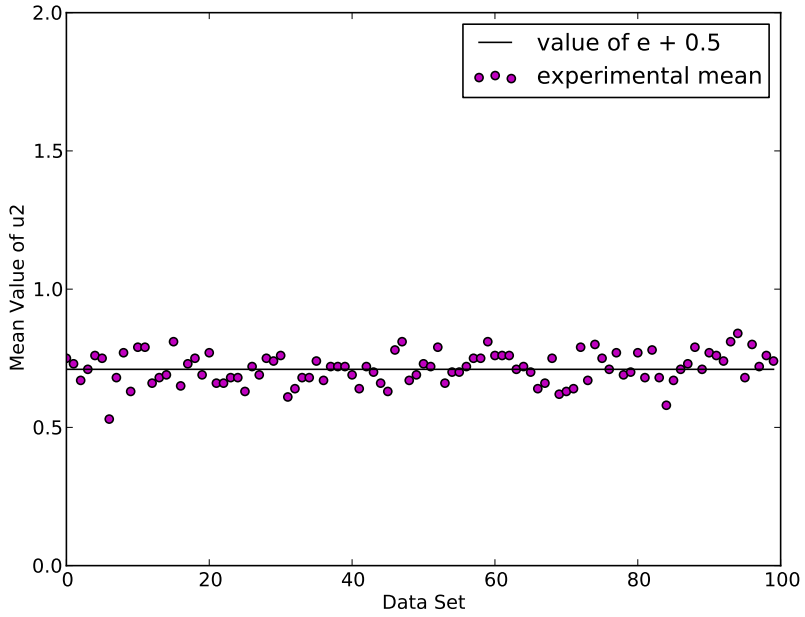


(b) Mean value of u_1

Figure 5.19: Comparing the theoretical expectations and experimentally computed means of the random variables w_1 and u_1, where the theoretical expectation of $w_1 = a$ and of $u_1 = c$.

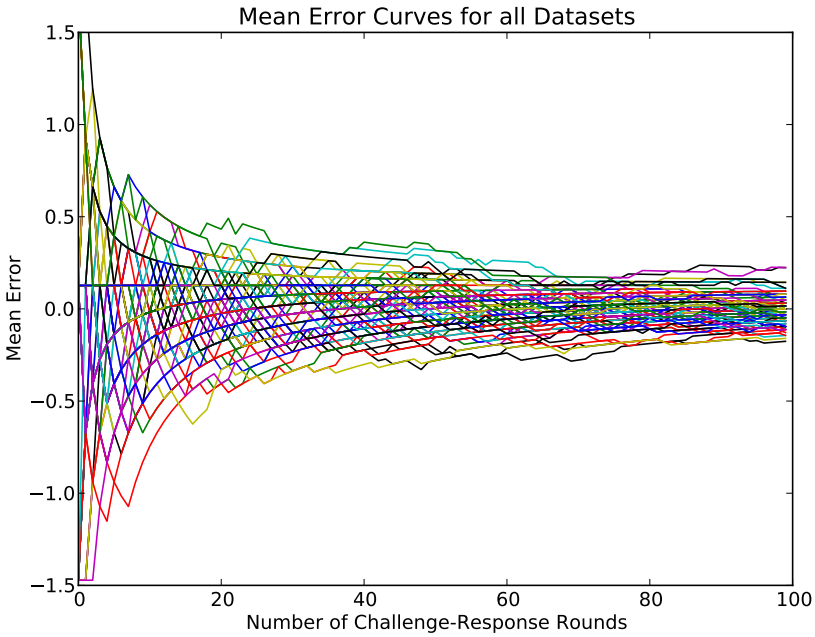


(a) Mean value of w_2

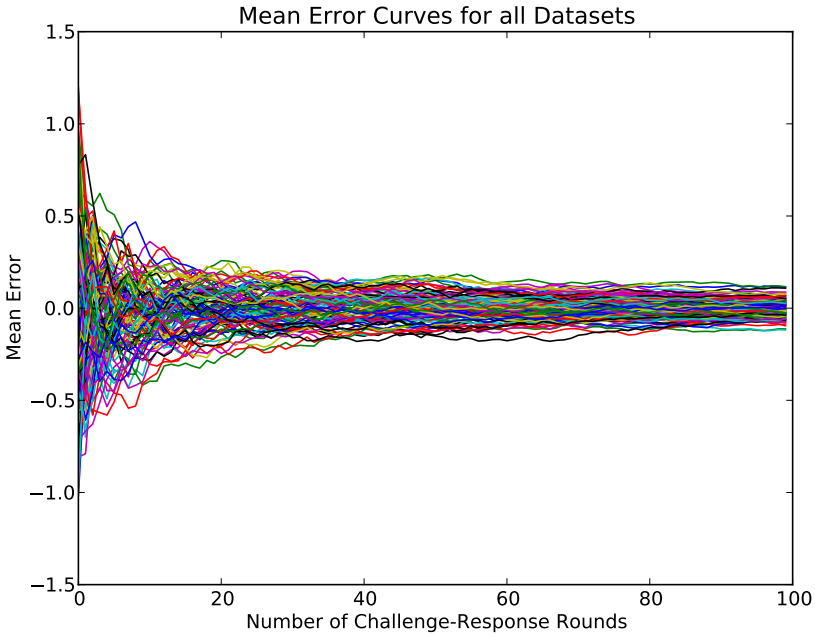


(b) Mean value of u_2

Figure 5.20: Comparing the theoretical expectations and experimentally computed means of the random variables w_2 and u_2, where the theoretical expectation of $w_2 = d + 0.5$ and of $u_2 = e + 0.5$.

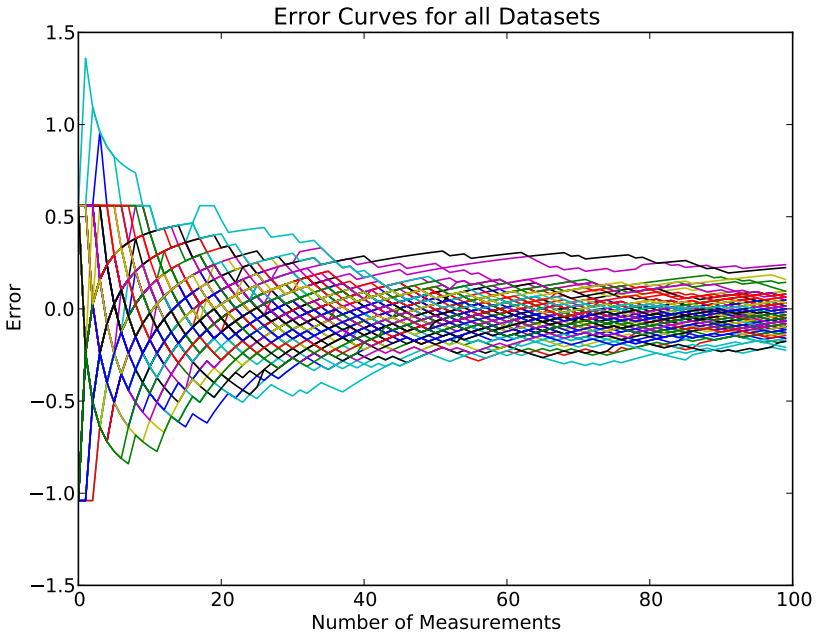


(a) Error in EM in the absence of clock synchronization

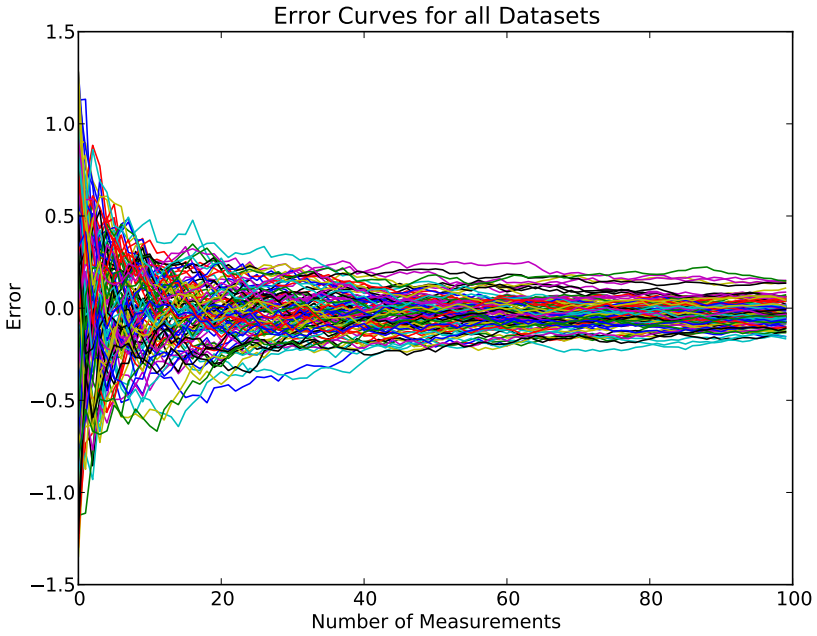


(b) Error in EM in the presence of clock synchronization

Figure 5.21: Comparing the error in the measured running time for a challenge-response relay in Elliptical Multilateration (EM) (a) without and (b) with precision clock synchronization.

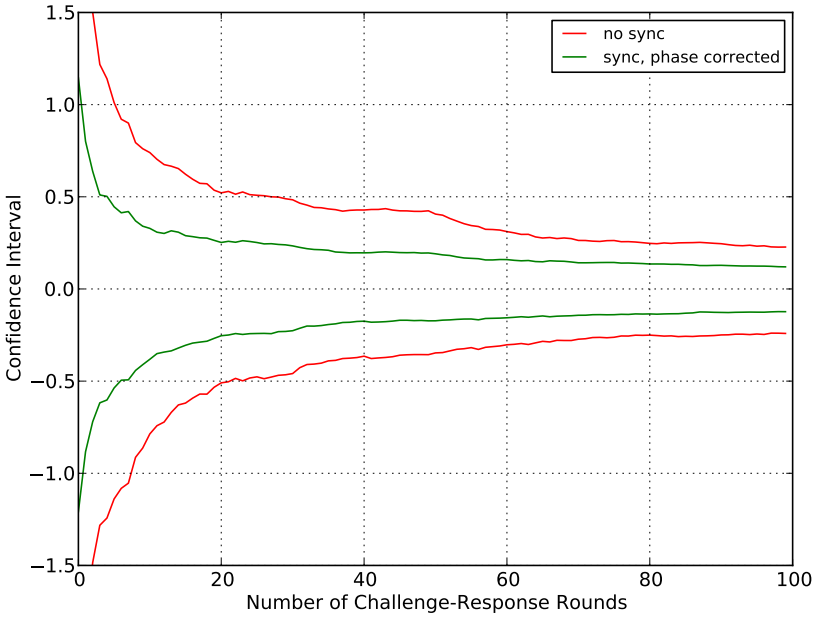


(a) Error in HM in the absence of clock synchronization

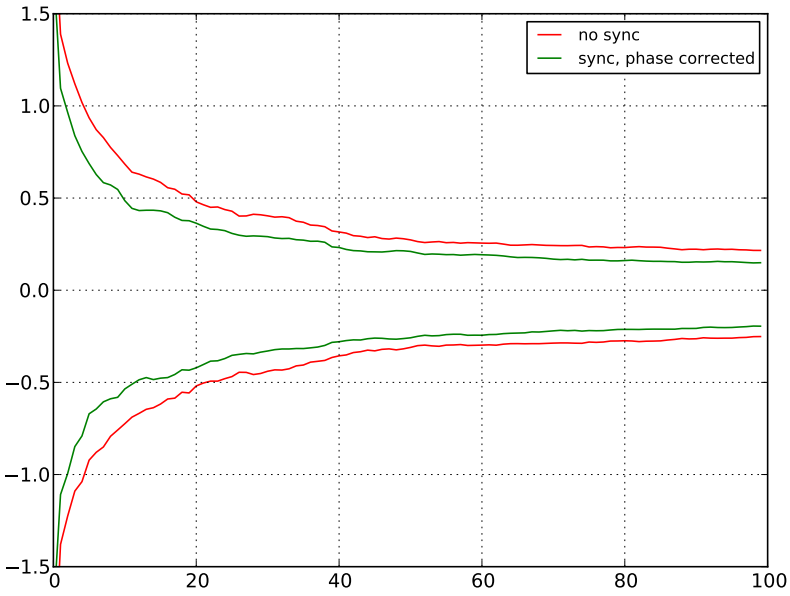


(b) Error in HM in the presence of clock synchronization

Figure 5.22: Comparing the error in the measured running time for a challenge-response relay in Hyperbolic Multilateration (HM) (a) without and (b) with precision clock synchronization.



(a) Confidence interval of error in EM



(b) Confidence interval of error in HM

Figure 5.23: Confidence Intervals for all four cases indicating that the resolution of the measurement is better than the clock period, allowing for sub-clock accuracy.

5.6 Conclusions

In this chapter we presented an anatomy of the error introduced during message transfer, between the hardware-based timestamping points are the sender and the receiver. The components that we identified were error due to channel effects, error due to signal processing delays in the transceiver hardware and error due to clock effects (offset and synchronization). We proposed ways to mitigate error due to each component in order to improve the accuracy of the time-of-flight (consequently, distance) measurements.

Assuming that we place the timestamping unit at symbol detection/creation, and correct for the error incurred in message transfer, the resolution of the timestamps is limited by the time period of the timestamping clock. We showed how we can extend Throbjornsen et al.'s technique, and apply it to SIMO localization protocols like Elliptical Multilateration (EM) and Hyperbolic Multilateration (HM) to capture timestamps with subclock resolution. This allows accuracy on the order of a few meters in localization over narrowband RF.

We also studied the effect of clock synchronization on the accuracy of both EM and HM. Through analysis, we computed the expected error in the geometrical constraint formed on the prover's location, when EM and HM are executed with and without clock synchronization. We showed that absence of clock synchronization has greater effect on the accuracy of geometrical constraints formed in EM as compared to the geometrical constraints formed in HM. This is because of the difference in the way the raw timestamps are used to form the geometrical constraints in either protocol. The other difference when EM and HM are executed is that in EM, the time at which the lead verifier sent the challenge (start time) is very important. The accurate value of the start time must be known. This is not possible when clock synchronization is absent, in which case it has to be estimated

by the witness observing the challenge-response dialog. In contrast, HM does not use the start time in the computation. The geometrical constraints formed in HM are not affected if the start time is not known.

From the confidence intervals for the error in the time-of-flight measurements, we observed that both EM and HM can limit the error to a single clock period or less in twenty or more challenge-response rounds. For either protocol, the number of measurements required to achieve the same accuracy is smaller when the verifier clocks are synchronized prior to protocol execution.

Chapter 6

Fast and Accurate Hyperbolic Multilateration Using Maximum Likelihood Estimation

In chapters 4 and 5, we showed that with appropriate architectural support for timestamping, suitable message structure and by minimizing error, it is possible for time-based localization protocols to locate the prover with an accuracy on the order of a few meters. However, with the technique used in chapter 5, a large number of measurements are required to achieve such accuracy. For a given clock period, its takes at least twenty challenge-response rounds to limit the error to one clock period or less, and at least a hundred rounds to limit the error to a tenth of the clock period. The large number of measurements required in the method described in chapter 5 however, can be a drawback for certain reasons. For example, in situations where security is of concern, a malicious

prover might not cooperate across a large number of message exchanges. A large number of message exchanges for localization are also not desirable since localization messages are an overhead from the perspective of minimizing traffic in the wireless network. Therefore, it is desirable to complete the protocol execution fast, and localize the prover (with the desirable accuracy) in the minimum possible number of message exchanges.

In this chapter, we introduce an alternative method for fast, yet accurate Hyperbolic Multilateration. Our method uses the well-studied technique of maximum likelihood estimation to form the hyperbolic constraint on the prover's location. Using this method, the error in formulating the difference equation can be limited to a tenth of the timestamping clock period or less, using half the number of challenge-response rounds required in the simple averaging method described in chapter 5.

6.1 Observed Time Difference of Arrival in SIMO HM

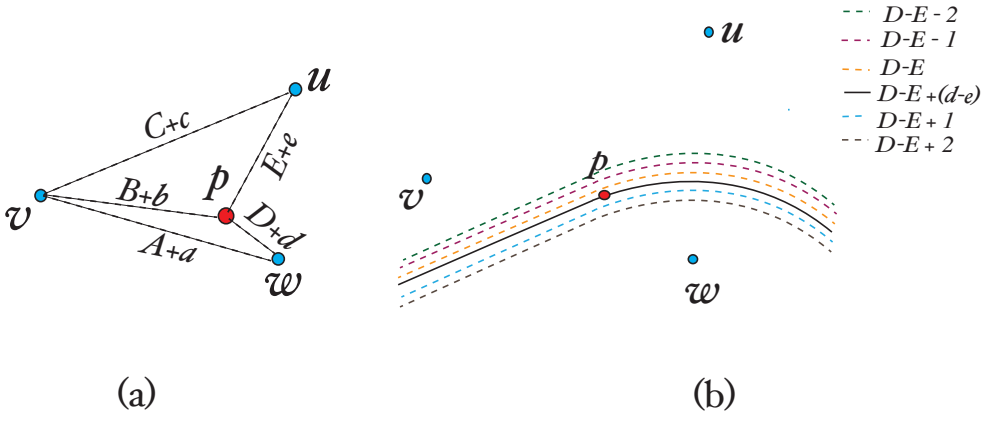


Figure 6.1: (a) The spatial positions of the three verifiers and the prover (b) Hyperbolic constraints formed according to the five different integer values that the measurements take on.

Recall that in SIMO hyperbolic multilateration, the verifiers measure the *difference*

in the arrival time of a message at different entities, the message being the response of the prover in the case of a secure localization by time-difference-of-arrival. The verifiers then mathematically combine these measurements to localize the prover. To understand how the observed difference in the pair-wise measurements of two verifiers differs from the true difference, let us consider the example from section 5.4.3, where three verifiers and a prover participate in SIMO hyperbolic multilateration. In this example, v is the lead verifier that sends the challenge to prover p, while verifiers u and w are passive listeners and act as "witnesses", as shown in Fig. 6.1(a).

As derived in chapter 5, the timestamps captured by witnesses w and u for the arrival of the response are respectively

$$C_p^w = t_o + B + p + \Delta_{\{v,w\}} + D + \alpha + w_2 \tag{6.1}$$

where

$$w_2 = \begin{cases} 0 & \beta \leq \alpha - d \\ 1 & \alpha - d < \beta < 1 + \alpha - d \\ 2 & 1 + \alpha - d < \beta \end{cases} \tag{6.2}$$

and

$$C_p^u = t_o + B + p + \Delta_{\{v,u\}} + C + \gamma + u_2 \tag{6.3}$$

where

$$u_2 = \begin{cases} 0 & \beta \leq \gamma - e \\ 1 & \gamma - e < \beta < 1 + \gamma - e \\ 2 & 1 + \gamma - e < \beta \end{cases} \tag{6.4}$$

The difference in the total propagation time of the challenge-response relay initiated by v and observed by witnesses w and u reduces to the difference in the propagation time of the response message alone, to the two witnesses alone the paths $p \to w$ and $p \to u$. The reason for this claim is as follows: if we consider the propagation time of the challenge-response relay along the two-hop paths $v \to p \to w$ and $v \to p \to u$, The first hop starting at v and ending at p is common for both the paths. Also, because of the assumption that the response time of the prover is same for all the verifiers, $\Delta_{\{v,w\}} = \Delta_{\{v,u\}}$. Therefore,

$$T(p,w) - T(p,u) = T_p(v,w) - T_p(v,u) \tag{6.5}$$

Using Eq.(5.47) and Eq.(5.45) to substitute for $T_p(v,w)$ and $T_p(u,v)$, we have

$$\begin{aligned} T(p,w) - T(p,u) &= D - E + (w_2 - u_2) \\ &= \theta \end{aligned} \tag{6.6}$$

where we use θ to denote the difference of the total propagation time of the challenge-response dialog as observed by the witnesses w and u. Let us also denote $w_2 - u_2$, which is the difference of the two random variables in Eq. (6.6) as a new random variable k. Hence Eq.(6.6) can be rewritten as

$$\theta = D - E + k \tag{6.7}$$

From the definitions of w_2 and u_2 in (6.2) and (6.4), we know that $\{w_2, u_2\} \in \{0,1,2\}$; hence their difference $k \in \{-2,-1,0,1,2\}$. Since D and E in Eq.(6.7) are integers, the observed difference θ must also take on one of five possible consecetive integer values depending on

the value of k.

From Fig.6.1, we find that the true difference in the distances of w and u from prover p is

$$\theta_{true} = D(p,w) - D(p,u) = (D - E) + (d - e) \tag{6.8}$$

where d and e are fractions. Therefore, the true difference in the propagation times of the challenge-response relay to witnesses w and u, is a fractional value in the range $\{(D - E - 1), (D - E + 1)\}$. The observed value of the difference θ, therefore does not equal the true value, rather the five different values that θ can assume are spread around the true difference θ_{true}, such that θ is either slightly greater than, or slightly smaller than θ_{true}.

Since the values of the timestamps in Eqs. (6.2) and (6.4) are affected by the offset and quantization between the participant's clocks, the variation in the observed difference, i.e., the θ values can also be attributed to these two effects. Fig 6.1(b) illustrates the combined effect of clock offset and quantization on the the hyperbolic constraints formed by the witnesses. For clarity, we show only a single lobe of each constraint – the lobe that is closer to the true location of the prover. The solid line which passes through the true location p, represents the hyperbolic constraint corresponding to θ_{true}. The dashed lines represent the constraints formed when θ takes on one of the five possible integer values $D - E + k$, where $k \in \{-2, -1, 0, 1, 2\}$.

Let us examine the effect that this has on the hyperbolic constraint formed on the prover's location. From mathematics of hyperbolic multilateration, it is known that the time-difference-of-arrival, when converted to the equivalent distance, is numerically equal to the vertex separation [1] of the hyperbolic constraint formed from the difference equation.

[1]The points on the two lobes of a hyperbola which are closest to each other are called the vertices of the

If the foci are fixed, increase in the time-difference-of-arrival increases the vertex separation of the hyperbolic constraint formed. As a result, the lobes of the hyperbola are pushed further away from each other along the major axis (the straight line through the foci), and move closer to the foci. Similarly, if the time-difference-of-arrival decreases, the vertex separation becomes smaller, and the lobes of the hyperbola are pushed closer together along the major axis. Therefore, the hyperbolic constraint formed from the estimated value of the time-difference-of-arrival θ, is either closer to, or further away from a witness in comparison to the constraint formed from the true value θ_{true}.

This brings us to the problem that the verifiers must solve in order to compute the prover's true location from the measurements: Given the observed values of $\theta = D - E + k$, and the phase offsets of the witnesses, α and γ for each challenge-response round, across a series of rounds, how can we estimate the true difference $\theta_{true} = (D - E) + (d - e)$?

6.2 Properties of Measurement Data in HM

While visually analyzing the data obtained from simulations for the simple averaging method, we found that in the case of hyperbolic multilateration, the observed θ values exhibit a set pattern. In particular, the occurrence of a specific value of k (the difference of the fractional terms w_2 and u_2 of θ) among the five possible values, in some challenge-response round, is a function of the *phase difference* of the witnesses, $\alpha - \gamma$, in that round, and the value of $(d - e)$ (the difference in the fractional terms of θ_{true}).

Figs. 6.2 and 6.3 illustrate this. In each figure, we plot the observed value of k for the samples as a function of the corresponding phase difference $\alpha - \gamma$. Since $k = \theta - (D - E)$,

hyperbola. The vertices are defined by the intersection of the lobes with the major axis of the hyperbola. The vertex separation is defined as the euclidian distance between the two vertices.

and $(D - E)$ is fixed, these plots essentially show the pattern of occurrence of the θ values as a function of $\alpha - \gamma$. For each plot we picked a different positive value for the true difference $(d - e)$, The $d - e$ value equals the x-intercept of the solid vertical line in each plot. Notice that as the value of the true difference $(d - e)$ increases across subfigures (a)-(d), the relative densities and range of occurrence of samples corresponding to each k value, change significantly. In particular, we observe the following:

Observation 1: The observed samples are grouped into five distinct bands, such that each band corresponds to one possible value of $k \in \{-2, -1, 0, 1, 2\}$.

Observation 2: The range of $\alpha - \gamma$ values across which samples corresponding to a specific k value occur, varies. Furthermore, none of the bands contain samples spanning the complete range of $\alpha - \gamma$ values i.e., from -1 to 1.

Observation 3: For a specific value $\alpha - \gamma$, we have samples corresponding to at most two distinct values of k.

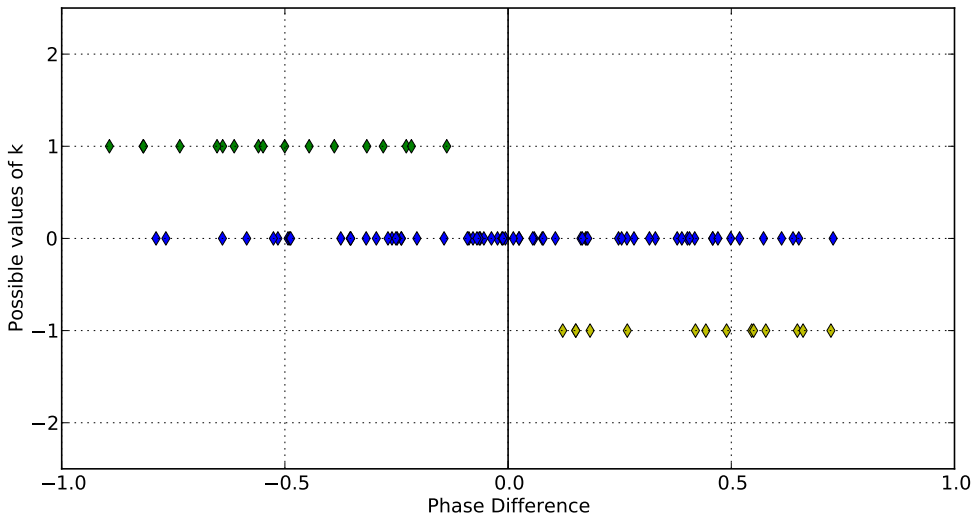
Observation 4: For each k value, the probability of occurrence of samples is highest at the center of the band, and decreases as we move towards the edges.

Observation 5: There is no sample in the band with $k = 1$ for which the corresponding $\alpha - \gamma$ value is greater than $(d - e)$. Similarly, there is no sample in band with $k = -1$ for which the corresponding $\alpha - \gamma$ value is smaller than d-e.

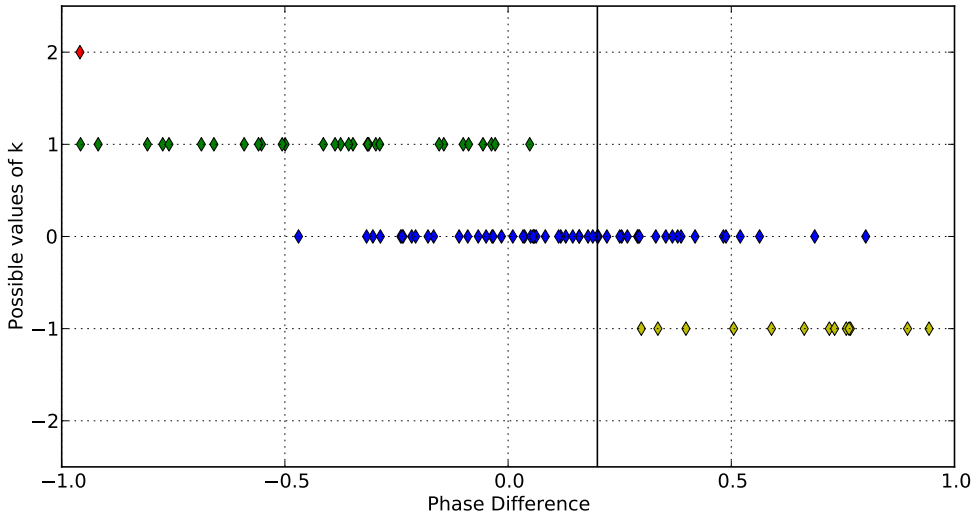
Observation 6: There is no sample in the band with $k = 2$ for which the corresponding $\alpha - \gamma$ value is greater than $(d - e - 1)$. Similarly, there is no sample in band with $k = 0$ for which the corresponding $\alpha - \gamma$ value is smaller than $(d - e - 1)$.

Since the density functions change predictably depending on the value $(d - e)$ in each case, it is possible to formulate a mathematical model that defines the pattern of

occurrence of the samples depending on parameter $(d - e)$. If we can find the appropriate model, the problem of finding the true value of $(d - e)$, and hence θ, can be solved by maximum likelihood estimation, which is a well-studied parameter estimation technique.

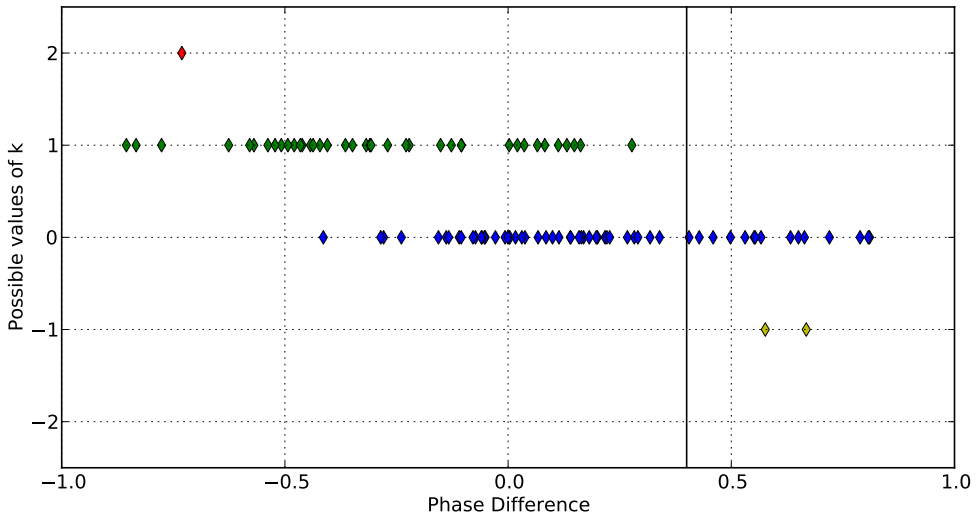


(a) d=0.50, e = 0.50

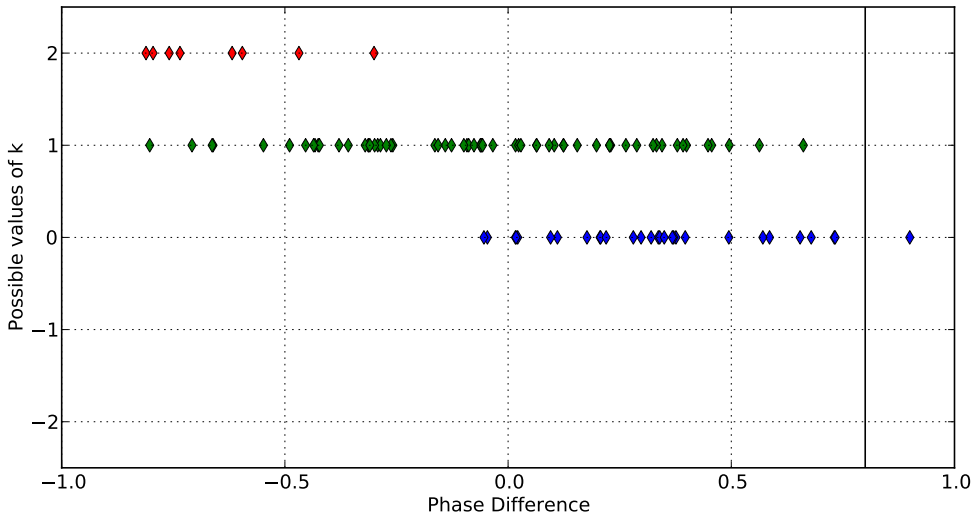


(b) d = 0.60, e = 0.40

Figure 6.2: Two representative plots showing how the pattern of occurrence of samples corresponding to each k value depends on the value of $d-e$. For the upper plot, $d-e = 0.00$, and for the lower plot, $d-e = 0.20$. Note that the vertical line at $\alpha-\gamma = d-e$ demarcates the range of $\alpha-\gamma$ values at which samples corresponding to $k = 1$ and $k = -1$ occur.



(a) d=0.70, e = 0.30



(b) d=0.90, e = 0.10

Figure 6.3: Two more representative plots showing how the pattern of occurrence of samples corresponding to each k value depends on the value of $d-e$. For the upper plot, $d-e = 0.40$, and for the lower plot, $d - e = 0.80$. Similar to the first two plots, the vertical line at $\alpha-\gamma = d-e$ demarcates the range of $\alpha-\gamma$ values at which samples corresponding to $k = 1$ and $k = -1$ occur.

6.3 Mathematical Modeling of the Observed Data

To apply the method of maximum likelihood estimation to the data collected when the entities participate in hyperbolic multilateration (HM), we must find the mathematical model that defines the pattern of occurrence of the five values that θ can take. In Eqs.(6.7), $(D - E)$ is a fixed integer, therefore, the value of θ for some challenge-response round, depends only on the integer random variable $k = w_2 - u_2$ for that round. Hence,

$$P(\theta = D - E + k) = P(w_2 - u_2 = k) \tag{6.9}$$

Similarly, in Eq.(6.8), $D - E$ is the fixed integer common to Eq.(6.7), therefore $(d - e)$ is the parameter that varies, and therefore dictates the observed pattern of occurrence of the samples. The problem of estimating θ_{true} when θ, α and γ are known for each challenge-response round, therefore reduces to the problem of estimating $d - e$ when $w_2 - u_2 = k$, α and γ are known for each round.

Recall Figs. 6.2 and 6.3, which show that the frequency of occurrence of specific values of θ in the observations is a function of *difference* of the phase-offset terms α and γ, conditional on $(d - e)$. If we separate the statistics for the occurrence of each of the five possible values of θ indexed by $\alpha-\gamma$, we can can compute the density function corresponding to each value of θ. The collection of all the five density functions can be denoted as $f(\alpha-\gamma, \theta|d-e)$, which defines the parametric model for the occurrence of θ as a function of $\alpha - \gamma$, conditional on $d - e$. Because of Eq.(6.9), we can replace θ with k in the expression such that the family of density functions denoted as $f(\alpha-\gamma, k|d-e)$ is an alternate definition for the model. Each member of this family is a density function corresponding to a specific

value of $k \in \{-2, -1, 0, 1, 2\}$. The difference $(d - e)$ is the deterministic but unknown parameter of the model, whose value needs to be estimated.

Consider the random variable k whose value is in turn is dependent on the two independent random variables w_2 and u_2. The pattern of occurrence of each of these random variables can also be modeled as a family of density functions. For example, $f(\alpha, j|d)$, where $j \in \{0, 1, 2\}$ represents the family of density functions that models the occurrence of specific values of the w_2. Similarly $f(\gamma, j|e)$, represents the family of density functions that models the occurrence of the specific values of u_2. From an analytical point of view, it is easy to derive the expressions for $f(\alpha, j|d)$ and $f(\gamma, j|e)$. This is because w_2 and u_2 have simple definitions as shown in Eqs. (6.2) and (6.4). In comparison, deriving the expression for $f(\alpha - \gamma, k|d - e)$ analytically is a harder problem. This is because θ is a function of a *difference term* $\alpha - \gamma$, conditional on another *difference term* $d - e$. First, we tackle the easier problem of deriving the expressions for $f(\alpha, j|d)$ and $f(\gamma, j|e)$. We then show how we can build on it to solve the harder problem of expressing $f(\alpha - \gamma, k|d - e)$ in terms of the difference terms $\alpha - \gamma$ and $(d - e)$.

6.3.1 Density Functions for Measurements of Individual Witnesses

To find the expressions for the density functions $f(\alpha, j|d)$ and $f(\gamma, j|e)$, we revisit the definitions of the two random variables w_2 and u_2. First, let us consider the definition of w_2. From Eq.(6.2), we know that w_2 is a function α and β, conditional on d. Therefore,

$$w_2 \quad = \quad F(\alpha, \beta|d) \tag{6.10}$$

where $F(x, y|p)$ denotes a function of two variables x and y, conditional on parameter p.

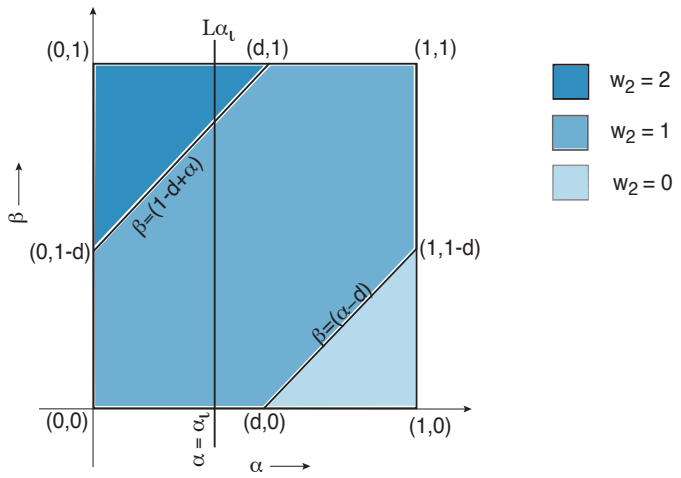


Figure 6.4: Random variable w_2 expressed as functions of the phase offset α and the parameter d

Fig. 6.4 shows a geometrical representation of $F(\alpha, \beta|d)$ in the two-dimensional coordinate space. The domain of $F(\alpha, \beta|d)$ is a unit square $S_w = \{(\alpha, \beta) : 0 \leq \alpha < 1, 0 \leq \beta < 1\}$ defined on the α and β axes. $F(\alpha, \beta|d)$ partitions S_w is into three distinct regions, such that $F(\alpha, \beta|d)$ takes on a unique value in $\{0, 1, 2\}$ in each region. The line $\beta = \alpha - d$ is the boundary between the region where $F(\alpha, \beta|d) = 0$ and the region where $F(\alpha, \beta|d) = 1$. Similarly the line $\beta = \alpha - d + 1$ the boundary between the region where $F(\alpha, \beta|d) = 1$ and the region where $F(\alpha, \beta|d) = 2$. If we separate the statistics of observed values of $F(\alpha, \beta|d)$ indexed by the phase offset α, then all the measurements for which α equals a specific value, say α_i, would be points on the line segment L_{α_i} within S_w, shown in the figure.

From lemma 1, $F(\alpha_i, \beta|d)$ for any point in L_{α_i} can assume only one of two possible values – either $F(\alpha_i, \beta|d)$ takes on values in the set $\{0, 1\}$ when $\alpha_i \leq d$, or it takes on values in the set $\{1, 2\}$ when $\alpha_i > d$. Furthermore, if we sample uniformly across all possible values of β, the probability $P(F(\alpha_i|d) = j)$ where $j \in \{0, 1, 2\}$, is equal to the length of the line segment L_{α_i} contained within the region where $F(\alpha, \beta|d) = j$.

If j_{high} and j_{low} are respectively the larger and smaller of the two values that $F(\alpha_i, \beta|d)$ can assume, then, across a uniform random sampling of β, we have

$$P(F(\alpha_i|d) = j_{high}) = \begin{cases} -(\alpha_i - d) & \alpha_i \leq d \\ 1 - (\alpha_i - d) & \alpha_i > d \end{cases} \tag{6.11}$$

$$P(F(\alpha_i|d) = j_{low}) = \begin{cases} 1 + (\alpha_i - d) & \alpha_i \leq d \\ (\alpha_i - d) & \alpha_i > d \end{cases} \tag{6.12}$$

Generalizing expressions (6.11) and (6.12), we have

$$P(F(\alpha_i, d) = j) = 1 - |d - \alpha_i + (1 - j)| \tag{6.13}$$

Notice that when $\alpha_i = d$, then $P(F(\alpha_i|d) = 1) = 1$.

Therefore, the family of density functions which models the pattern of occurrence of the $w_2 = F(\alpha, \beta|d)$ values across a uniform sampling of β is

$$f(\alpha, j|d) = max\{0, 1 - |d - \alpha + (1 - j)|\} \tag{6.14}$$

Since $F(\alpha_i|d)$ can either be j_{high} or j_{low} along any $\alpha = \alpha_i$, we also have

$$\begin{aligned} f(\alpha_i|d) &= \sum_{j \in \{j_{high}, j_{low}\}} (P(F(\alpha_i|d) = j) \\ &= 1 \end{aligned} \tag{6.15}$$

which is the probability density function of the random variable w_2, assuming a uniform sampling of β.

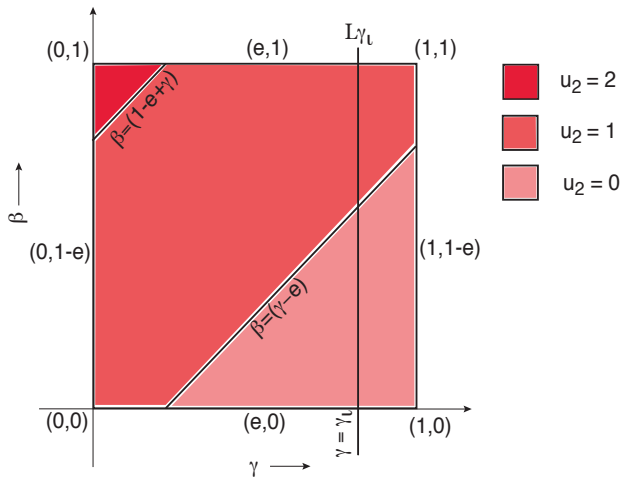


Figure 6.5: Random variable u_2 expressed as functions of the phase offset γ and the parameter e

Next we consider the definition of u_2 in Eq.(6.4). Similar to Eq.(6.10), u_2 can be written as

$$u_2 = F(\gamma, \beta|e) \tag{6.16}$$

By representing u_2 on the two-dimensional coordinate space as shown in Fig. 6.5, through similar analysis, we can show that

$$f(\gamma, j|e) = max\{0, 1 - |e - \gamma + (1 - j)|\} \tag{6.17}$$

is the expression for the density function for a specific value of j, and the probability density function of the random variable u_2 is

$$\begin{aligned} f(\gamma_i|e) &= \sum_{j \in \{j_{high}, j_{low}\}} (P(F(\gamma_i|e) = j) \\ &= 1 \end{aligned} \tag{6.18}$$

Through analysis, we showed how we can compute the density functions which model the pattern of occurrence of the three values $j \in \{0, 1, 2\}$ for each random variable. In particular, we represented each random variable in the two-dimensional coordinate plane, and used geometrical properties to compute the probabilities of occurrence of each possible value of j. In the case of random variable w_2, we derived the expressions for $P(F(\alpha_i|d) = 0), P(F(\alpha_i|d) = 1)$ and $P(F(\alpha_i|d) = 2)$, and subsequently the expressions for $f(\alpha, j|d)$ and $f(\gamma, j|e)$. We also derived similar expressions for the random variable u_2. Our goal however, is to find the expression for $f(\alpha - \gamma, k|d - e)$. Next, we show how we can apply the same technique for the difference of the these two random variables, which in turn is the random variable $k = w_2 - u_2$. First, we suitably represent $F(\alpha - \gamma, k|d - e)$ in the two-dimensional coordinate space. Then, we use geometrical properties to find the expression for the family of density functions $f(\alpha - \gamma, k|d - e)$, which models the pattern of occurrence of the five values that k can assume.

6.3.2 Density Functions for the Difference of Pair-wise Measurements

To find the expression for the family of density functions $f(\alpha - \gamma, k|d - e)$, we consider the difference term $w_2 - u_2$. From the Eqs.(6.10) and (6.16), we have

$$w_2 - u_2 = F(\alpha, \beta|d) - F(\gamma, \beta|e) \tag{6.19}$$

Although the three phase offset terms α, β and γ are independent of each other, the phase offset β of the prover, is common for both the witnesses in any challenge-response round. The fractional distances d and e are unknown but deterministic, hence their difference $d - e$ is also unknown but deterministic. Fig. 6.6(a) shows a representation of both

$F(\alpha, \beta|d)$ and $F(\gamma, \beta|e)$ such that the β-axis is aligned for both the functions. To represent $F(\alpha, \beta|d) - F(\gamma, \beta|e)$ on the two-dimensional coordinate space [2], we overlap the planes containing $F(\alpha, \beta|d)$ and $F(\gamma, \beta|e)$ such that the phase offsets α and γ are both represented along the horizontal axis, while β is represented along the vertical axis. This is illustrated in Fig 6.6(b), which shows us the top view of the overlapping planes. For clarity, we have removed the fill from the figure. Instead, we color lines belonging to a particular plane with the color of the plane.

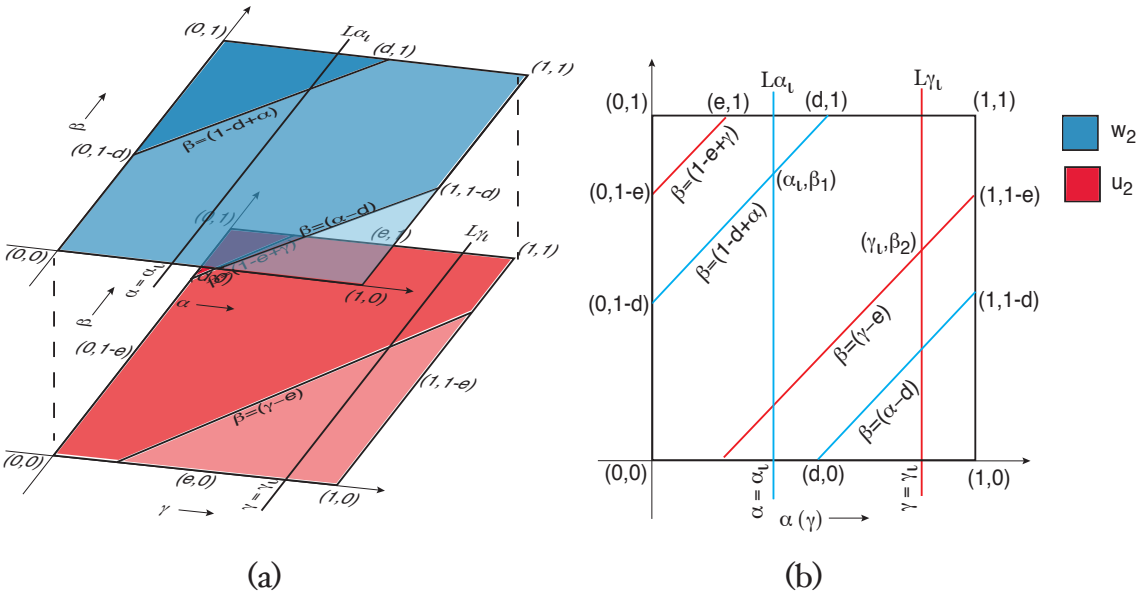


Figure 6.6: (a) Overlapping geometrical representations of $F(\alpha, \beta|d)$ and $F(\gamma, \beta|e)$ by aligning along the β axis. (b) Two-dimentional representation of $F(\alpha - \gamma, \beta|d - e)$. The fill color has been removed for clarity, but the line colors correspond to the color of the original function that they belong to – blue represents $F(\alpha, \beta|d)$ and red represents $F(\gamma, \beta|e)$.

Let us denote the unit square formed by the overlap of S_w and S_u as $S_{\{u,w\}}$. The points on the line segment L_{α_i} represent measurements of witness w for which its phase

[2]Note that other ways to represent $F(\alpha, \beta|d) - F(\gamma, \beta|e)$ in the coordinate space are also possible. For example, if we choose to represent $F(\alpha, \beta|d) - F(\gamma, \beta|e)$ in the three-dimensional coordinate space instead, we can plot each of the phase offset terms α, γ and β on three separate axes. In this case, the domain of $F_d(\alpha, \beta) - F_e(\gamma, \beta)$ will be a unit cube.

offset $\alpha = \alpha_i$. L_{α_i} intersects the line $\beta = \alpha - d + 1$ when $\alpha_i \leq d$, and the line $\beta = \alpha - d$ when $\alpha_i > d$. In Fig.6.6(b), β_1 denotes y coordinate of the point at which L_{α_i} intersects either line, such that

$$\beta_1 \equiv (\alpha - d) \mod 1 \tag{6.20}$$

Similarly, The points on the line segment L_{γ_i} within $S_{\{w,u\}}$ represent measurements of witness u for which its phase offset $\gamma = \gamma_i$. L_{γ_i} intersects the line $\beta = \gamma - e + 1$ when $\gamma_i \leq e$, and the line $\beta = \gamma - d$ when $\gamma_i > e$. In the same figure, β_2 denotes the y coordinate of the point at which L_{γ_i} intersects either line, such that

$$\beta_2 \equiv (\gamma - e) \mod 1 \tag{6.21}$$

From Lemma 2, we know that for any α_i and γ_i, the difference $F_d(\alpha, \beta) - F_e(\gamma, \beta)$ can assume only one of two possible values in $\{-2, -1, 0, 1, 2\}$ if $\beta_1 \neq \beta_2$, and only a single value when $\beta_1 = \beta_2$. Let us denote the larger of the two possible values as k_{high} and the smaller value as k_{low}. From Observation 2, we also know that across a uniform sampling of the prover's phase offset β, the probability of k_{high} is equal to the fraction of β values which satisfy $min\{\beta_1, \beta_2\} \leq \beta < max\{\beta_1, \beta_2\}$.

$$P(F(\alpha_i|d) - F(\gamma|e) = k_{high}) = |\beta_1 - \beta_2| \tag{6.22}$$

The probability of k_{low} is therefore,

$$P(F(\alpha_i|d) - F(\gamma|e) = k_{low}) = 1 - |\beta_1 - \beta_2| \tag{6.23}$$

The probability of occurrence of a specific value of $k \in \{-2, -1, 0, 1, 2\}$ can therefore be computed given any 4-tuple $\{\alpha_i, \gamma_i, d, e\}$.

Theorem 1: Given 4-tuples $\{\alpha, \gamma, d, e\}$ belonging to the equivalence class defined in Lemma 3, where $\alpha_i - \gamma_i = \sigma$ and $d - e = \rho$, the probability of occurrence of a specific value of $k \in \{-2, -1, 0, 1, 2\}$ is

$$P(F(\alpha, \beta|d) - F(\gamma, \beta|e) = k) = (1 - |\sigma|) * max\{0, 1 - |\rho - \sigma - k|\}$$

Proof:

To derive the expressions for the desired probabilities, we start with a sample measurement from a single round of challenge-response where $\alpha = \alpha_i$ and $\gamma = \gamma_i$. A sample measurement from the ith challenge-response round can therefore be represented by the four tuple $\{\alpha_i, \gamma_i, d, e\}$. Since the spatial positions of the participants does not change across a series of challenge-response rounds in raid succession, d and e are constant across different rounds for one complete execution of hyperbolic multilateration. Since $\sigma = \alpha_i - \gamma_i$ and $\rho = d - e$, each of these two variables can independently be positive or negative in the range $(-1, 1)$, depending on the values of α_i, γ_i, d and e.

From observation 3, we know that keeping the value of σ and ρ fixed, we can apply a suitable parameter shift m to both the phase offset terms α_i and γ_i, and a suitable parameter shift n to both fractional distance terms d and e, without effecting the probabilities $P\{F(\alpha_i, \beta|d) - F(\gamma_i, \beta|e) = k\}$. To derive the expressions for the desired probabilities, we consider four separate cases as tabulated in Table 6.1. We divide the range each of the two variables σ and ρ into two parts – the range over which the variable is negative, and

the range it is positive. Each row in the table corresponds to a unique combination of the possible ranges of σ and ρ, each being either positive or negative, as shown in the second and third columns of he table.

To simplify the analysis for each case, we apply suitable parameter shifts: m to the $\{\alpha_i, \gamma_i\}$ pair, and n to the $\{d, e\}$ pair, such that one of the new values in each pair is 0, while preserving the signs of both σ and ρ. The fourth and fifth columns of Table 6.1 show the parameter shifts applied to the values in the 4-tuple $\{\alpha_i, \gamma_i, d, e\}$ depending on the values of σ and ρ. The new 4-tuples formed are shown in the sixth column of Table 6.1. The derivation for the desired probabilities is as follows:

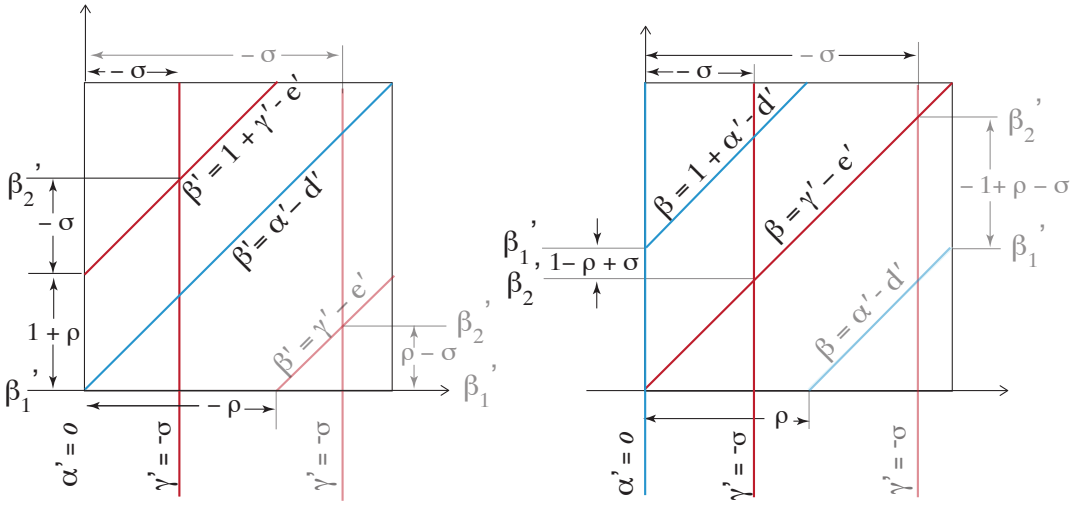


Figure 6.7: Geometrical representation of the 4-tuple $\{\alpha_i', \gamma_i', d', e'\}$ formed after the suitable parameter shifts have been applied in case (i)(a) and in case (i)(b). The blue lines correspond to the function $F(\alpha, \beta|d)$, while the red lines correspond to the function $F(\gamma, \beta|e)$.

Case (i)a If we apply Lemma 3 to the 4-tuple $\{\alpha_i, \gamma_i, d, e\}$ such that $m = -\alpha_i$ and $n = -d$ as shown in the first row of Table 6.1, the new four tuple formed is $\{0, \gamma_i', 0, e'\}$,

Case	Range of σ	Range of ρ	Shift $\{\alpha_i, \gamma_i\}$ by $m =$	Shift $\{d, e\}$ by $n =$	New 4-tuple
(i)a	$-1 < \sigma \leq 0$	$-1 < \rho \leq 0$	$-\alpha_i$	$-d$	$\{0, \gamma_i', 0, e'\}$
(i)b	$-1 < \sigma \leq 0$	$0 < \rho \leq 1$	$-\alpha_i$	$-e$	$\{0, \gamma_i', d', 0\}$
(ii)a	$0 < \sigma \leq 1$	$-1 < \rho \leq 0$	$-\gamma_i$	$-d$	$\{\alpha_i', 0, 0, e'\}$
(ii)b	$0 < \sigma \leq 1$	$0 < \rho \leq 1$	$-\gamma_i$	$-e$	$\{\alpha_i', 0, d', 0\}$

Table 6.1: Parameter shifts applied to the original 4-tuple for simplifying the analysis

where $\gamma_i' = -\sigma$ and $e' = -\rho$. This 4-tuple is illustrated in Fig.6.7(a). Substituting $\alpha = 0$ and $d = 0$ in Eq.(6.20), we have

$$\beta_1' = 0 \tag{6.24}$$

and substituting $\gamma = -\sigma$ and $e = -\rho$ in Eq.(6.21), we have

$$\begin{aligned} \beta_2' &= (-\sigma + \rho) \bmod 1 \\ &= \begin{cases} 1 + \rho - \sigma & \gamma_i' \leq e' \\ \rho - \sigma & \gamma_i' > e' \end{cases} \end{aligned} \tag{6.25}$$

Therefore, from Eqs. (6.24) and (6.25), we have

$$|\beta_1' - \beta_2'| = \begin{cases} 1 + \rho - \sigma & \gamma_i' \leq e' \\ \rho - \sigma & \gamma_i' > e' \end{cases} \tag{6.26}$$

From Lemma 1, we know that the function $F(\alpha, \beta|d)$ can take on one of only two values, i.e., $F(\alpha_i', \beta|d) \in \{F(\alpha, 0|d), F(\alpha, 1^-|d)\}$. Similarly, $F(\gamma, \beta|e) \in \{F(\gamma, 0|e), F(\gamma, 1^-|d)\}$. Substituting β_1' from Eq.(6.24) for y_1, β_2' from Eq.(6.25) for y_2, α_i' for a, and γ_i' for b in

Eq.(II.16), we have

$$F(\alpha_i', \beta|d) - F(\gamma_i', \beta|e) = \begin{cases} 0 & 1+\rho-\sigma & \gamma_i' \le e' \\ -1 & \sigma-\rho & \gamma_i' \le e' \\ 1 & \rho-\sigma & \gamma_i' > e' \\ 0 & 1-\sigma+\rho & \gamma_i' > e' \end{cases} \tag{6.27}$$

Therefore, when when both σ and ρ are negative, $k \in \{-1, 0, 1\}$ with the respective probabilities shown in Eq.(6.27).

Case (i)(b) When we apply Lemma 3 to the 4-tuple $\{\alpha_i, \gamma_i, d, e\}$ such that $m = -\alpha$ and $n = -\gamma$ as shown in the second row of Table 6.1, the new 4-tuple formed is $\{0, \gamma_i', d', 0\}$, where $\gamma_i' = -\sigma$ and $d' = \rho$. This 4-tuple is illustrated in Fig. 6.7(b). Substituting $\alpha = 0$ and $d = \rho$ in Eq.(6.20), we have

$$\beta_1' = (0 - \rho) \bmod 1 = 1 - \rho \tag{6.28}$$

and substituting $\gamma = -\sigma$ and $e = 0$ in Eq.(6.21), we have

$$\beta_2' = (-\sigma - 0) \bmod 1 = -\sigma \tag{6.29}$$

Therefore, from Eqs.(6.28) and (6.29), we have

$$|\beta_1' - \beta_2'| = \begin{cases} 1-\rho+\sigma & \gamma_i' \le d' \\ -1+\rho-\sigma & \gamma_i' > d' \end{cases} \tag{6.30}$$

Similar to Eq.(6.27), we can substitute β_1' from from Eq.(6.28) for y_1, β_2' from Eq.(6.29) for y_2, α_i' for a, and γ_i' for b in Eq.(II.16) from to obtain

$$F(\alpha_i, \beta|d) - F(\gamma_i, \beta|e) = \begin{cases} 0 & 1-\rho+\sigma & \gamma_i' \leq d' \\ 1 & \rho-\sigma & \gamma_i' \leq d' \\ 2 & -1+\rho-\sigma & \gamma_i' > d' \\ 1 & 2-\rho+\sigma & \gamma_i' > d' \end{cases} \tag{6.31}$$

Therefore, when σ is negative, but ρ is positive, $k \in \{0, 1, 2\}$, with the respective probabilities shown in Eq.(6.31).

From Lemma 3, we know that the 4-tuples $\{0, \gamma_i', 0, e'\}$ and $\{0, \gamma_i', d', 0\}$ in **Cases (i)(a)** and **(i)(b)** belong to the same equivalence class. Hence we can apply Observation 4 to aggregate over the range $[-\sigma, 1)$ for each $k \in \{-2, -1, 0, 1, 2\}$.

$$\begin{aligned} \int_{-\sigma}^{1} P(k|\alpha_i', \gamma_i', d, e) d(\alpha_i' - \gamma_i') &= \int_{-\sigma}^{1} P(F(\alpha_i'|, d) - F(\gamma_i'|e) = k) d(\alpha_i' - \gamma_i') \\ &= (1+\sigma)(1 - |\rho - \sigma - k|) \end{aligned} \tag{6.32}$$

for all $\alpha_i' - \gamma_i' = \sigma$.

Case (ii)a In this case, we apply Lemma 3 to the 4-tuple $\{\alpha_i, \gamma_i, d, e\}$ such that $m = -\gamma_i$ and $n = -d$ as shown in the third row of Table 6.1. The new 4-tuple formed is $\{\alpha_i', 0, 0, e'\}$, where $\alpha_i' = \sigma$ and $e' = -\rho$. The new 4-tuple is illustrated in Fig. 6.8(a).

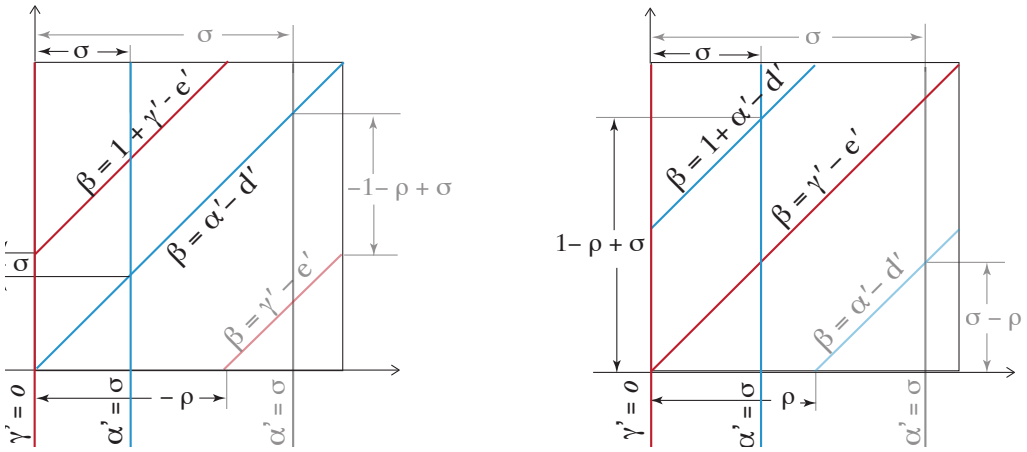


Figure 6.8: Geometrical representation of the 4-tuple $\{\alpha_i', \gamma_i', d', e'\}$ formed after the suitable parameter shifts have been applied in case (ii)(a) and in case (ii)(b). The blue lines correspond to the function $F(\alpha, \beta|d)$, while the red lines correspond to the function $F(\gamma, \beta|e)$.

Substituting σ for a and 0 for d in Eq.(6.20), we have

$$\beta_1' = \sigma \bmod 1 = \sigma \tag{6.33}$$

and substituting 0 for γ, and $-\rho$ for e in Eq.(6.21), we have

$$\beta_2' = \rho \bmod 1 = 1 + \rho \tag{6.34}$$

Therefore, from Eqs.(6.33) and (6.34), we have

$$|\beta_1' - \beta_2'| = \begin{cases} 1 + \rho - \sigma & \alpha_i' \leq e' \\ -1 - \rho + \sigma & \alpha_i' > e' \end{cases} \tag{6.35}$$

From Lemma 1, we know that $F(\alpha, \beta|d) \in \{F(\alpha, 0|d), F(\alpha, 1^-|d)\}$ and $F(\gamma, \beta|e) \in$

$\{F(\gamma, 0|e), F(\gamma, 1^-|d)\}$. Substituting β_1' from Eq.(6.33) for y_1, β_2' from Eq.(6.34) for y_2, α_i' for a, and γ_i' for b in Eq.(II.16), we have

$$F(\alpha_i', \beta|d) - F(\gamma_i', \beta|e) = \begin{cases} 0 & 1+\rho-\sigma \quad \alpha_i' \le e' \\ -1 & \sigma-\rho \quad \alpha_i' \le e' \\ -2 & -1-\rho+\sigma \quad \alpha_i' > e' \\ -1 & 2-\rho-\sigma \quad \alpha_i' > e' \end{cases} \tag{6.36}$$

Therefore, when σ is positive and ρ is negative, $k \in \{-2, -1, 0\}$, with the corresponding probabilities shown in Eq.(6.36).

Case (ii)b When we apply Lemma 3 to the 4-tuple $\{\alpha_i, \gamma_i, d, e\}$ such that $m = -\gamma_i$ and $n = -e$ as shown in the fourth row of Table 6.1, the new 4-tuple formed is $\{\alpha_i', 0, d', 0\}$, where $\alpha_i' = \sigma$ and $e' = \rho$. The new 4-tuple is shown in Fig. 6.8(b). Substituting σ for a and ρ for d in Eq.(6.20), we have

$$\begin{aligned} \beta_1' &= (\sigma - \rho) \bmod 1 \\ &= \begin{cases} 1-\rho+\sigma & \alpha_i' \le d' \\ \sigma-\rho & \alpha_i' > d' \end{cases} \end{aligned} \tag{6.37}$$

and substituting 0 for both γ and e in Eq.(6.21)

$$\beta_2' = 0 \bmod 1 = 0 \tag{6.38}$$

From Eqs.(6.37) and (6.38)

$$|\beta'_1 - \beta'_2| = \begin{cases} 1-\rho+\sigma & \alpha'_i \leq d' \\ \sigma-\rho & \alpha'_i > d' \end{cases} \tag{6.39}$$

From Lemma 1, we know that $F(\alpha, \beta|d)$ and $F(\gamma, \beta|e)$ can each take only one of two values. Substituting β'_1 from Eq.(6.37) and β'_2 from Eq.(6.38)

$$F(\alpha'_i, \beta|d) - F(\gamma'_i, \beta|e) = \begin{cases} 0 & 1-\rho+\sigma & \alpha'_i \leq d' \\ 1 & \rho-\sigma & \alpha'_i \leq d' \\ -1 & \sigma-\rho & \alpha'_i > d' \\ 0 & 1+\rho-\sigma & \alpha'_i > d' \end{cases} \tag{6.40}$$

Therefore, when σ and ρ are both positive, $k \in \{-1, 0, 1\}$, with the corresponding probabilities as shown in Eq.(6.40).

Since the 4-tuples $\{\alpha', 0, 0, e'\}$ and $\{\alpha', 0, d', 0\}$ in **Cases (ii)(a)** and **(ii)(b)** belong to the same equivalence class, as defined in Lemma 3, we can apply Observation 4 to aggregate over the range $[\sigma, 1)$ for each $k \in \{-2, -1, 0, 1, 2\}$.

$$\begin{aligned} \int_{\sigma}^{1} P(k|\alpha'_i, \gamma'_i, d, e) d(\alpha'_i - \gamma'_i) &= \int_{\sigma}^{1} P(F(\alpha'_i|, d) - F(\gamma'_i|e) = k) d(\alpha'_i - \gamma'_i) \\ &= (1-\sigma)(1-|\rho-\sigma-k|) \end{aligned} \tag{6.41}$$

for all $\alpha_i' - \gamma_i' = \sigma$.

Generalizing over all cases, from Eqs.(6.32) and (6.41), we have

$$\begin{aligned} f(\alpha - \gamma, k|d - e) &= (1 - |\sigma|)\{max(0, 1 - |\rho - \sigma - k|)\} \\ &= (1 - |\alpha - \gamma|)\{max(0, 1 - |(d - e) - (\alpha - \gamma) - k|)\} \end{aligned} \quad (6.42)$$

which is the expression for the family of density functions that defines our model.

6.4 Computing the Maximum Likelihood Estimate

In the previous section, we derived the expressions for the fractional density functions for each possible value of k. By doing so, we formulated a mathematical model for the pattern of occurrence of the five possible θ values when multiple challenge-response rounds are executed in SIMO Hyperbolic Multilateration. Given the family of density functions defined by the expression (6.42), we can now apply the maximum likelihood estimation method to estimate the true value of the parameter $(d - e)$.

6.4.1 Making Measurements and Collecting the Input Data

Our model requires as input, pair-wise differences between the phase offsets of the witnesses, and their "corrected" timestamps for each round of challenge-response. In each round, the measurements are made as follows: witness w records the phase offset α of its local clock with respect to the lead verifier v's clock, and a "corrected" timestamp $(C_p^w - \alpha)$, by subtracting α from the timestamp it records for the arrival of the response. Similarly witness u records its phase offset γ, and a corrected timestamp $(C_p^u - \gamma)$. The value of θ for

each round is computed by subtracting the "corrected" timestamp of one witness from that of the other's, i.e., $\theta = (C_p^w - \alpha) - (C_p^u - \gamma)$. Since θ values have a one-on-one mapping to the k values, the corresponding k values are known when θ is known. The difference in the individual phase offsets of the witnesses $\alpha - \gamma$ is also computed for each round. The sequence of tuples $\{\alpha - \gamma, k\}$ from all the challenge-response rounds executed, is the input data vector fed to the maximum likelihood estimation algorithm. Table **??** further illustrates how the input data vector is obtained.

6.4.2 Expression for The Log-Likelihood Function

The input data vector formed by measurements from all challenge-response rounds can be represented as

$$(\boldsymbol{\alpha - \gamma, k}) = \{((\alpha - \gamma)_1, k_1), ((\alpha - \gamma)_2, k_2), ((\alpha - \gamma)_3, k_3), \cdots ((\alpha - \gamma)_n, k_n)\} \tag{6.43}$$

where the individual elements $((\alpha - \gamma)_i, k_i)$ are statistically independent because the phase offsets of the witnesses vary randomly across different rounds. Due to this property, we have

$$f(\boldsymbol{\alpha - \gamma, k}|d-e) = f((\alpha-\gamma)_1, k_1|d-e) \cdot f((\alpha-\gamma)_2, k_2|d-e) \cdots f((\alpha-\gamma)_n, k_n|d-e) \tag{6.44}$$

Therefore, the likelihood function is

$$\begin{aligned} L(d-e|\boldsymbol{\alpha}-\boldsymbol{\gamma},\boldsymbol{k}) &= f(\boldsymbol{\alpha}-\boldsymbol{\gamma},\boldsymbol{k}|d-e) \\ &= f((\alpha-\gamma)_1,k_1|d-e)\cdot f((\alpha-\gamma)_2,k_2|d-e)\cdots f((\alpha-\gamma)_n,k_n|d-e) \end{aligned} \tag{6.45}$$

Taking the logarithm on both sides, we have

$$\begin{aligned} log(L(d-e|\boldsymbol{\alpha}-\boldsymbol{\gamma},\boldsymbol{k})) &= log(f(\boldsymbol{\alpha}-\boldsymbol{\gamma},\boldsymbol{k}|d-e)) \\ &= log(f((\alpha-\gamma)_1,k_1|d-e)) + log(f((\alpha-\gamma)_2,k_2|d-e)) + \cdots \\ & \quad log(f((\alpha-\gamma)_n,k_n|d-e)) \end{aligned} \tag{6.46}$$

This function is a convex function and attains a single maxima in the entire range of $\alpha-\gamma$ values. The maximum likelihood estimate for (d-e) is the $\alpha-\gamma$ value for which the expression (6.46) attains its maximum value.

6.4.3 Minimizing the Search Range for the Maximum Likelihood Estimate

To compute the maximum likelihood estimate for $(d-e)$, we must evaluate the value of the log-likelihood function at test values across the entire possible range of $(d-e)$. If n denotes the number of test points where we evaluate the log-likelihood function, then the accuracy of the solution depends on the value of n, as well as the range of $(d-e)$ values across which we search for the maximum likelihood estimate. Therefore, the accuracy of the

estimate can be improved either by increasing the number of test points, or by minimizing the search range of $(d-e)$. Since using fewer test points over a smaller range of values is a better approach in terms of the amount of computation required and the accuracy obtained, we investigated if we can narrow down the range of $(d-e)$ values over which we search for the maximum likelihood estimate.

We plotted the theoretically-derived densities for occurrence of k values (equivalently θ values) as a function of $\alpha-\gamma$. Figs. 6.9 and 6.10 show plots the four representative $(d-e)$ values from section 6.2. Across all plots, e use a consistent one-on-one mapping for the color of a density curve to the k value that it corresponds to. Table 6.2 shows the mapping of the colors to the specific values of k.

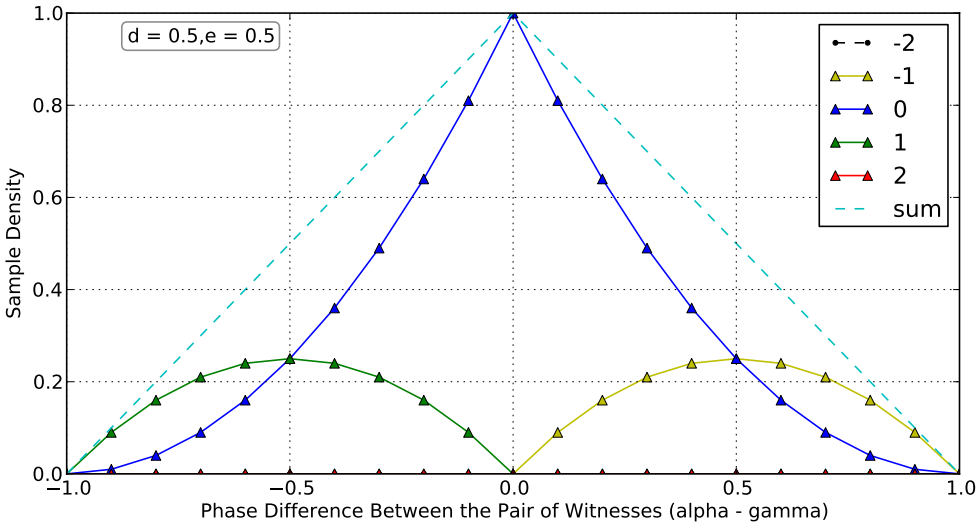
k =	-2	-1	0	1	2
color	black	yellow	blue	green	red

Table 6.2: Mapping k values to the colors of the density curves.

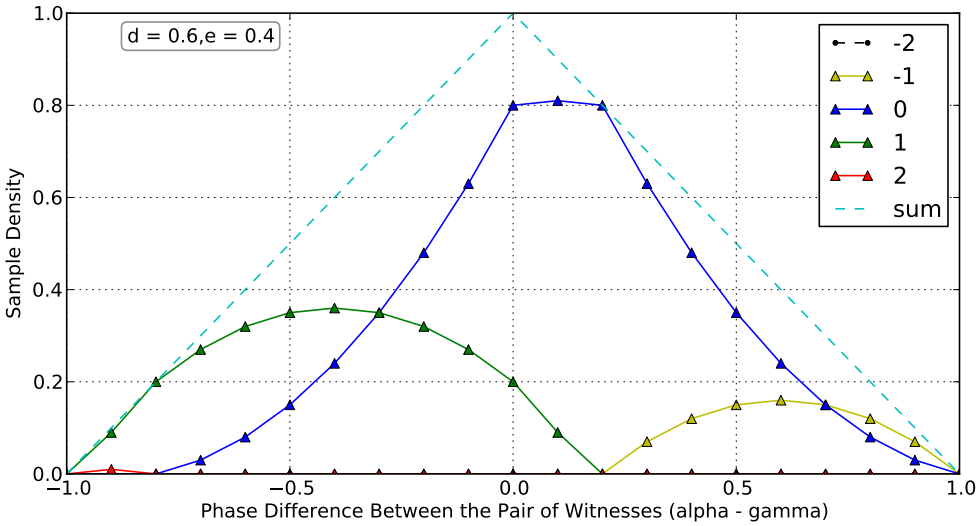
Notice that in each plot, the true value of (d-e) equals the value of the phase difference $\alpha-\gamma$ where the probability of obtaining a yellow sample, and the probability of obtaining a green sample, both become zero. Furthermore, the value of the phase difference $\alpha-\gamma$ where the probability of obtaining a blue sample and the probability of obtaining a red sample, both become zero, equals $(d-e-1)$. Next, we describe how we used these two observations to narrow down the search range for the maximum likelihood estimate. Consider the samples that belong to the green curve, where $k=1$, and the yellow curve, where $k=-1$. Because of **observation 5** from section 6.2, we know that the largest $\alpha-\gamma$ value, say $(\alpha-\gamma)_l$, corresponding to a sample belonging to the green curve, must be smaller than the true value of $(d-e)$. Similarly, the smallest value of $\alpha-\gamma$, say $(\alpha-\gamma)_u$,

corresponding to a sample belonging to the yellow curve, must be greater than the true value of (d-e). Therefore, $(\alpha-\gamma)_l$ and $(\alpha-\gamma)_u$ respectively are a lower and an upper bound on the true value of $(d-e)$. Now consider the samples belonging to the blue curve, where $k = 0$, and the red curve, where $k = 2$. Because of **observation 6** from section 6.2, the largest $\alpha - \gamma$ value corresponding to a sample from the red curve, say $(\alpha - \gamma)'_l$, is smaller than (d-e-1). Therefore we can reset the lower bound of the search range to $max\{(\alpha - \gamma)'_l + 1, (\alpha - \gamma)_l\}$. Also, the smallest $(\alpha - \gamma)$ value corresponding to a sample belonging to the blue curve, say $(\alpha-\gamma)'_u$, is greater than (d-e-1). Similar to resetting the lower bound, we can also reset the upper bound of the search range to $min\{(\alpha - \gamma)'_u + 1, (\alpha - \gamma)_u\}$.

Therefore, the deterministic and finite spread of the density functions, which depends on the true value of $(d - e)$, allows us to significantly narrow down the search range for the maximum likelihood estimate. Computing these bounds from the empirical samples, must therefore be incorporated as a preprocessing step in the execution of the maximum likelihood estimation algorithm.

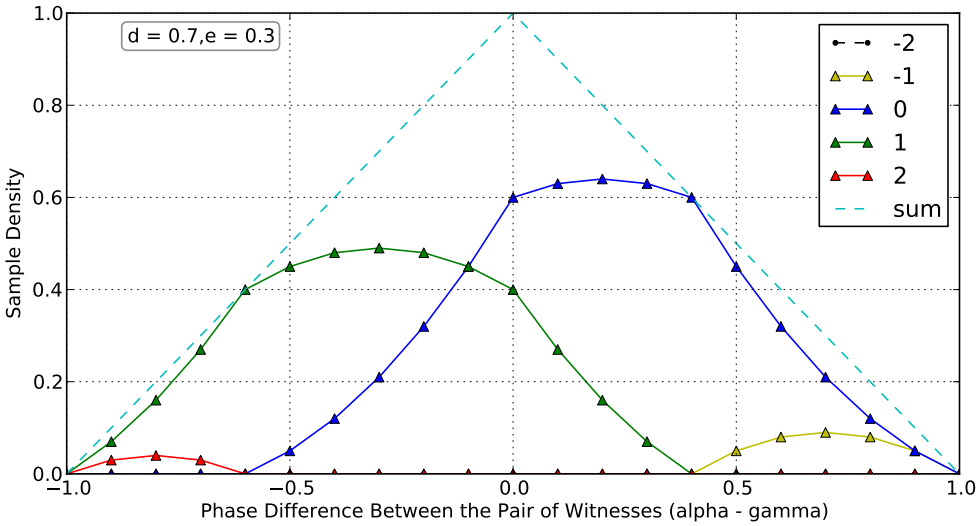


(a) d=0.50, e = 0.50

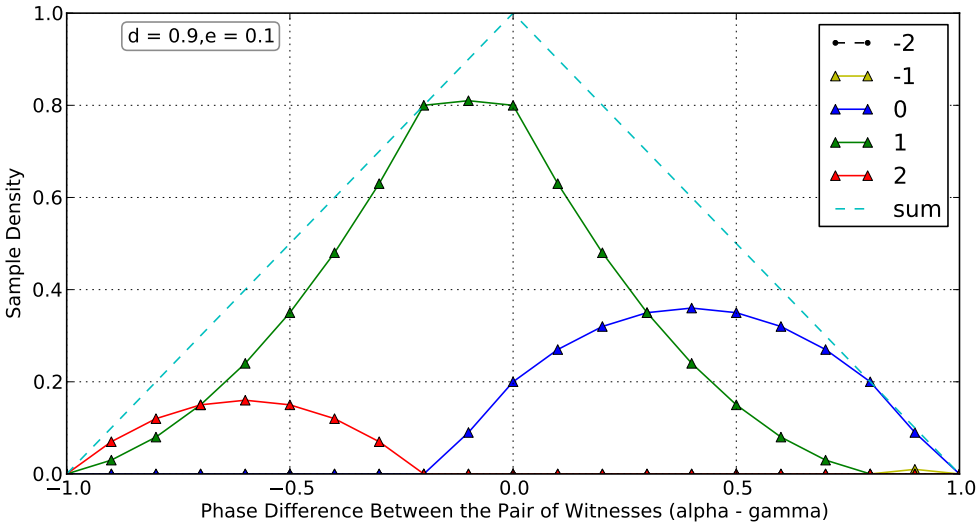


(b) d = 0.60, e = 0.40

Figure 6.9: Two representative plots showing the theoretically derived density functions for representative $d - e$ values used before. For the upper plot, $d - e = 0.0$, and for the lower plot, $d - e = 0.20$



(a) d=0.70, e = 0.30



(b) d=0.90, e = 0.10

Figure 6.10: Two more representative plots showing theoretically derived density functions for the representative $d - e$ values used before. For the upper plot, $d - e = 0.40$, and for the lower plot, $d - e = 0.80$

6.5 Simulation Results

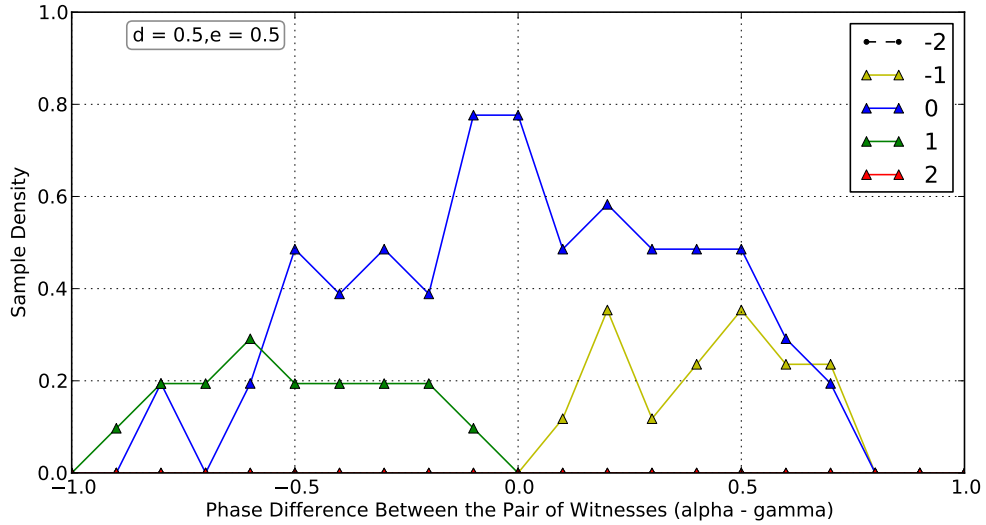
To evaluate the performance of our algorithm in comparison with the simple averaging technique mentioned in chapter 5, we performed simulations using three verifiers and a prover as before.

For a given "placement" represented by some tuple $\{A + a, B + b, C + c, D + e, E + e\}$, a single execution of HM consisted of 100 challenge-response rounds. Keeping the "placement" fixed, we conducted a total of 100 executions of HM to plot the confidence interval of the error.

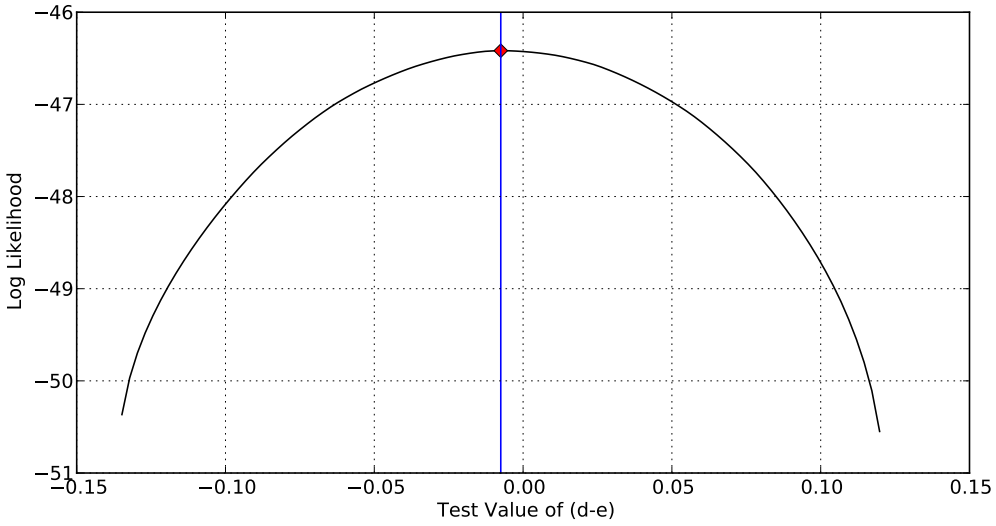
For our first set of experiments, we used the same representative tuple from the simulations of chapter 5. Keeping the distances of the witnesses and the prover from the lead verifier constant, we evaluated the accuracy of the hyperbolic constraint formed as the fractional difference $d - e$ changes. The measurements from each challenge-response round are binned into one of five bins corresponding to each k value. After collecting measurements over some number of challenge-response rounds, it is possible to plot the experimentally observed densities of each k value as a function of the phase difference $(\alpha - \gamma)$. The spread of each density function along the $(\alpha - \gamma)$ axis is also estimated by finding the maximum and minimum $(\alpha - \gamma)$ values corresponding to a sample with the k value for which the range is being computed. Following this, we bound the search space for the maximum likelihood estimate by extracting the maximum and minimum $\alpha - \gamma$ values corresponding to samples belonging to each bin, and follow the procedure described in section 6.4.3. Sometimes, it might not be possible to narrow down the search range while approaching the solution from wither side. This may happen when there are not enough samples, and two or more bins are empty. We address this problem by simply setting the lower bound to the default -1 when

we cannot find a better lower bound based on the samples, and the upper bound to the default 1 when we cannot find a better upper bound based on the samples. This results in searching for the solution over a much wider range, yet the accuracy does not degrade much, as we will see in the plots. The interval between the lower and upper bounds is then divided into 10 test points, uniformly distributed throughout the range. Notice that increasing the number of test points should theoretically lead to a more accurate solution because the likelihood function will be smoother. However, through experimentation we found that few test points (10 for our experiments) suffice and increasing the number beyond that does not lead to significant increase in the accuracy of the estimate.

Figs. 6.11 - 6.14 show the simulation results for the four representative $(d - e)$ values for which we have used in previous sections. The second plot in each figure shows the log-likelihood function for each case. As expected, the log-likelihood function is a smooth convex function, with a unique maxima. The maximum likelihood estimate for $(d - e)$ is shown by the solid vertical line in each plot. Despite the fact that experimentally observed sample distribution is significantly distorted compared to the theoretical distribution (due to the small number of samples collected), the maximum likelihood estimate is very accurate. This is true even if the sample count is very low, irrespective of the true value of $(d - e)$. To quantify the accuracy of the maximum likelihood estimate, we plotted its confidence interval as a function of the number of challenge-response rounds executed. We found that the confidence interval converges to less than 0.2 clock periods in at most 50 measurements.

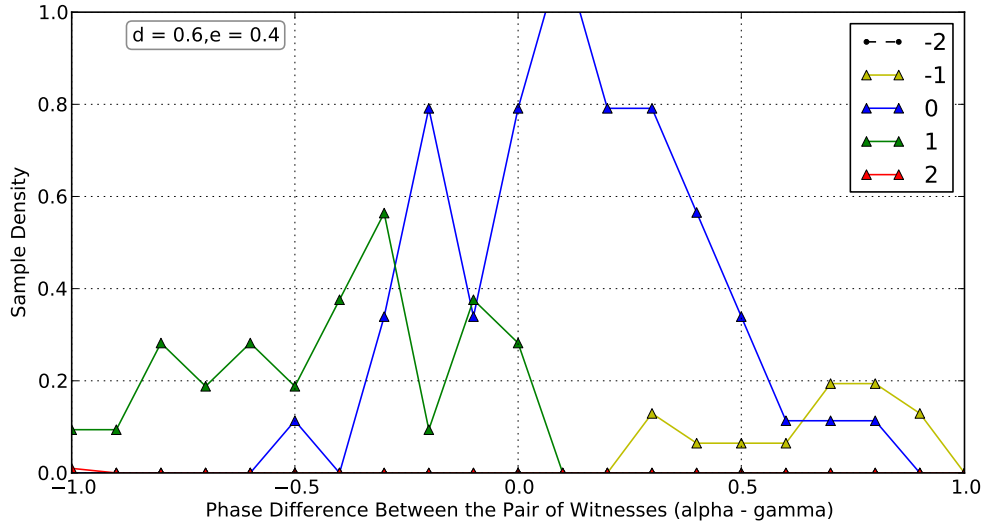


(a) Density functions of the observed samples when d=0.50, e = 0.50

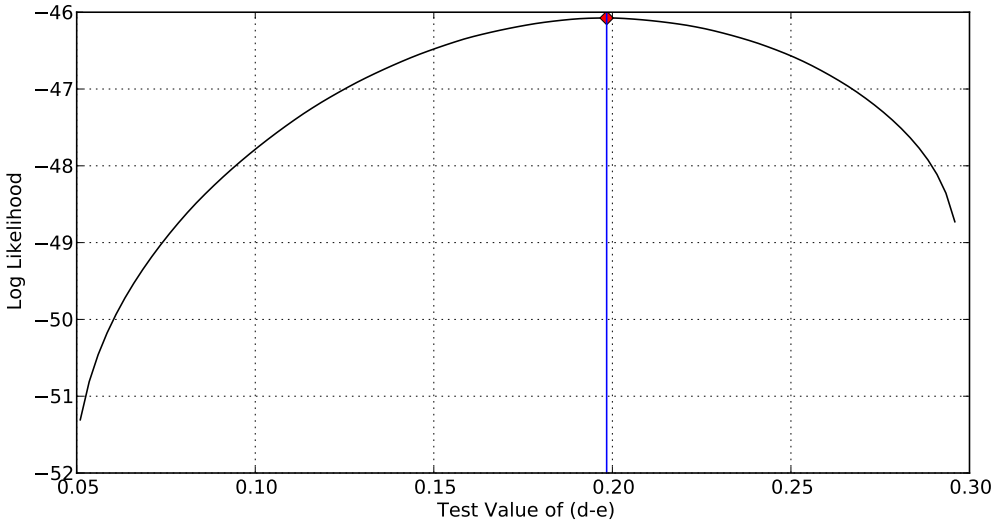


(b) Log-likelihood function and the maximum likelihood estimate for $(d-e)$, when the true value of $(d-e) = 0.0$

Figure 6.11: Experimentally observed density functions and the maximum likelihood estimate for the representative tuple when $(d - e) = 0.0$

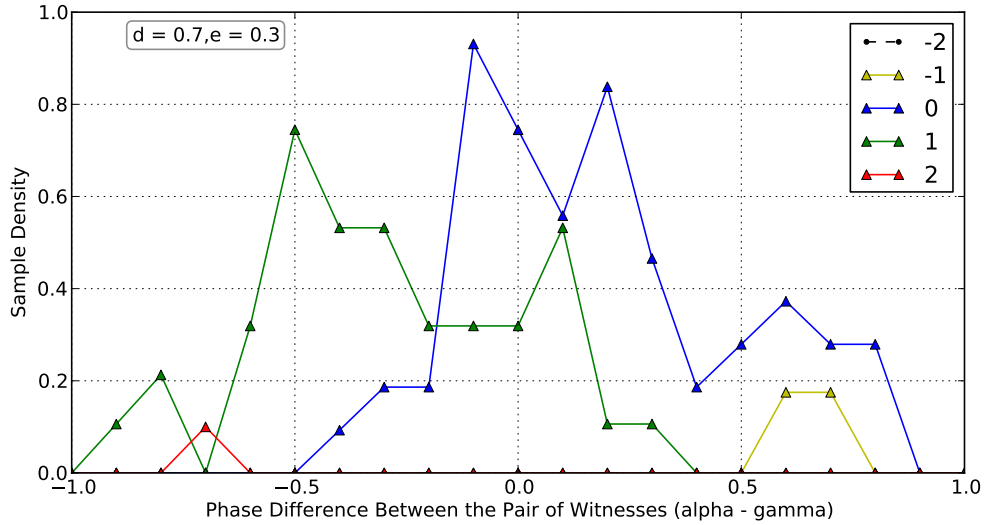


(a) Density functions of the observed samples when d=0.60, e = 0.40

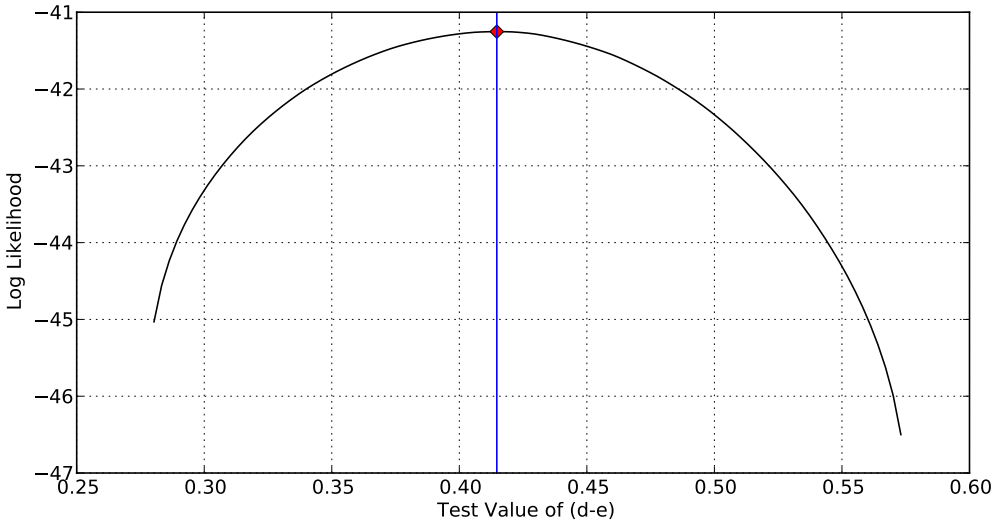


(b) Log-likelihood function and the computed estimate for $(d-e)$, when the true value of $(d-e) = 0.2$

Figure 6.12: Experimentally observed density functions and the maximum likelihood estimate for the representative tuple when value of $(d-e) = 0.2$

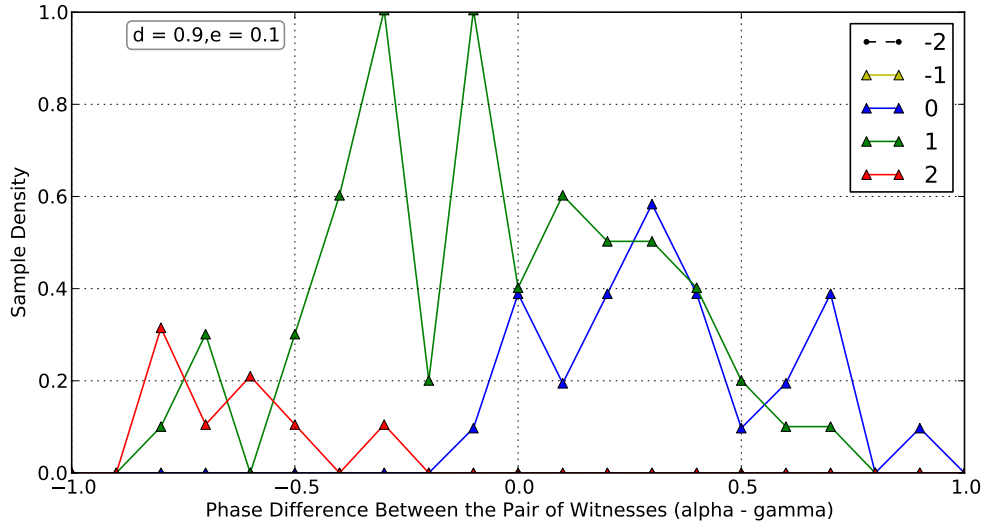


(a) Density functions of the observed samples when d=0.70, e = 0.30

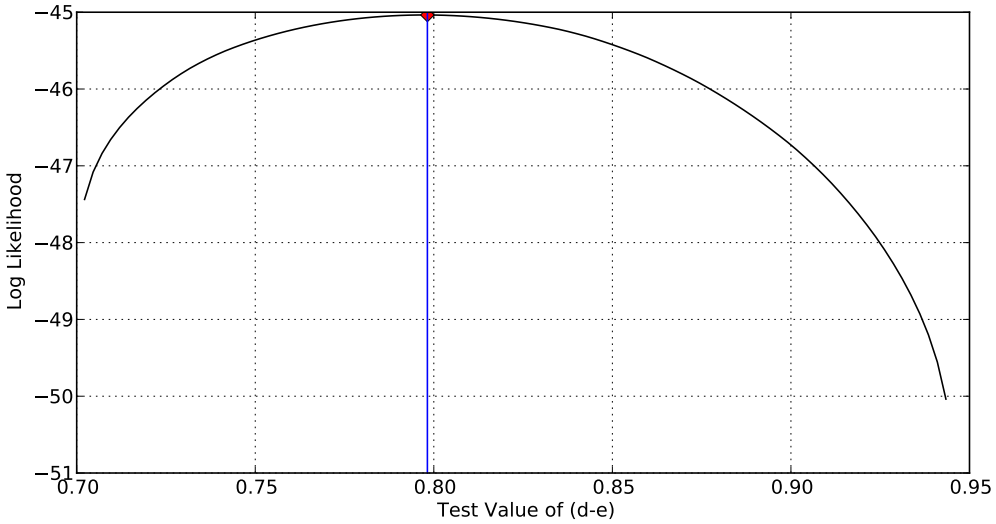


(b) Log-likelihood function and the computed estimate for $(d-e)$, when the true value of $(d-e) = 0.4$

Figure 6.13: Experimentally observed density functions and the maximum likelihood estimate for the representative tuple when value of $(d-e) = 0.4$

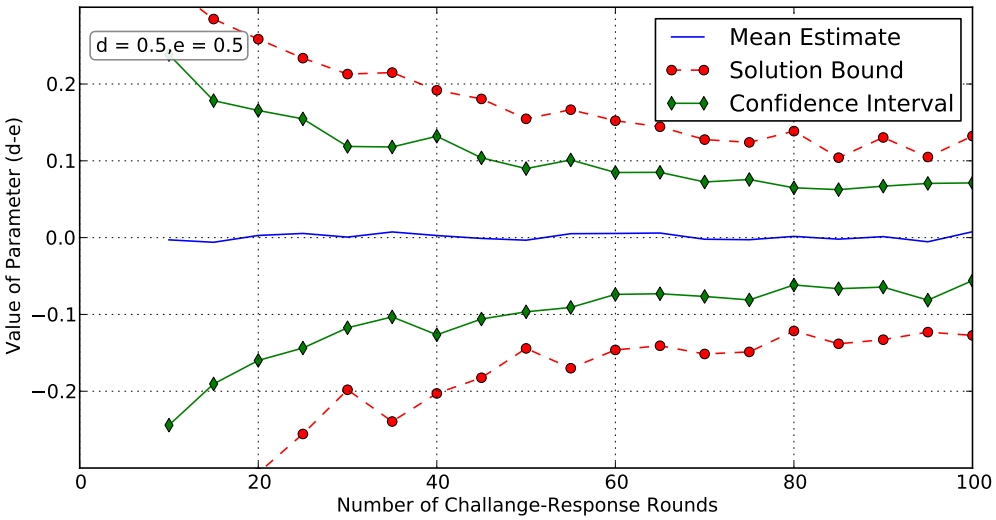


(a) Density functions of the observed samples when d=0.90, e = 0.10

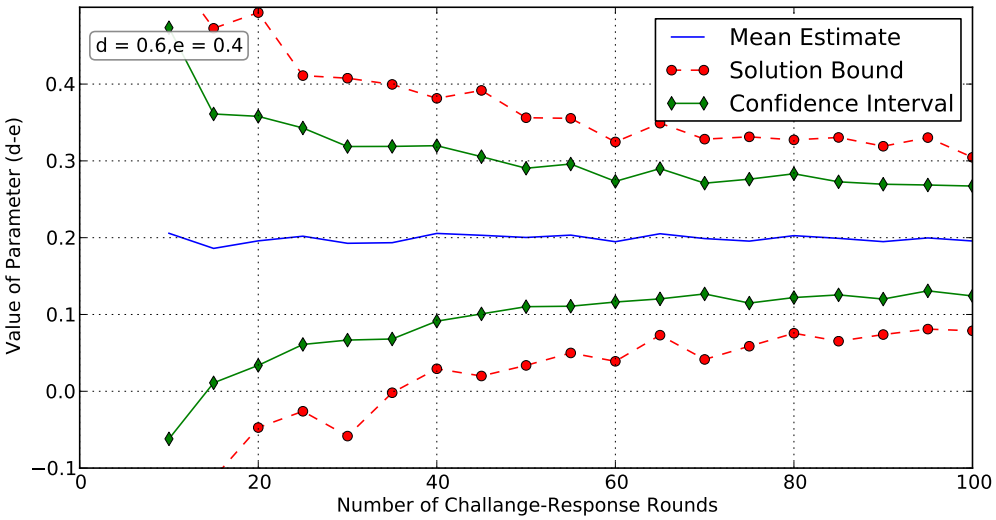


(b) Log-likelihood function and the computed estimate for $(d-e)$, when the true value of $(d-e) = 0.8$

Figure 6.14: Experimentally observed density functions and the maximum likelihood estimate for the representative tuple when value of $(d-e) = 0.8$

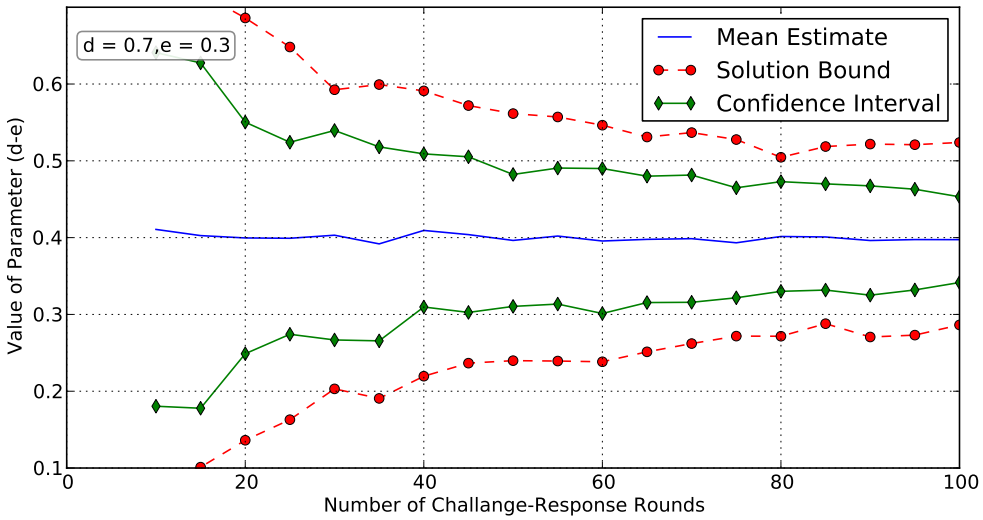


(a) Confidence Interval of estimate for $(d-e)$ when true value of d=0.50 and e = 0.50

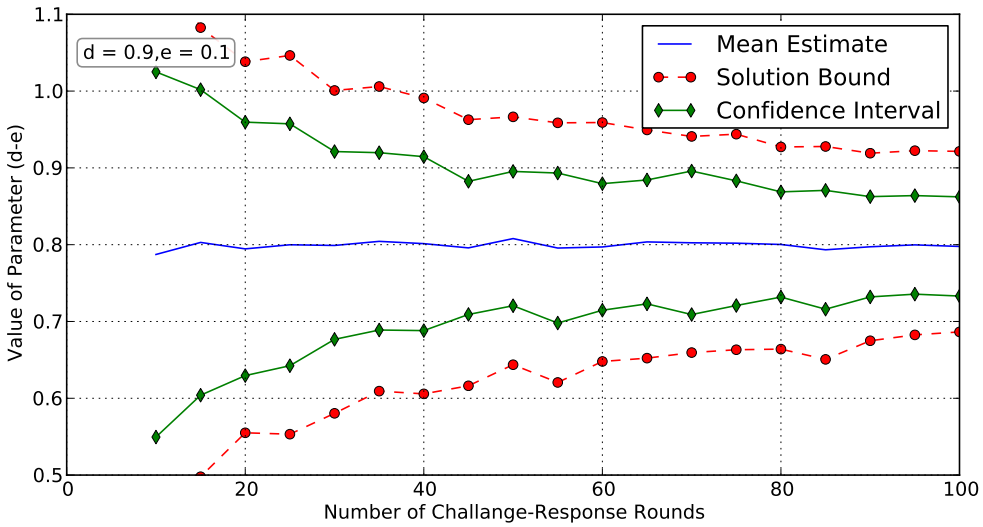


(b) Confidence Interval of estimate for $(d-e)$ when true value of d=0.60 and e = 0.40

Figure 6.15: Confidence intervals for the estimated value of $(d-e)$ for the first two cases.



(a) Confidence Interval of estimate for $(d-e)$ when true value of d=0.70 and e = 0.30



(b) Confidence Interval of estimate for $(d-e)$ when true value of d=0.90 and e = 0.10

Figure 6.16: Confidence intervals for the estimated value of $(d-e)$ for the last two cases.

6.6 Conclusions

In this chapter, we developed a new method for constructing the hyperbolic constraints in Hyperbolic Multilateration (HM) accurately. For a given accuracy, our method requires significantly fewer measurements in comparison with the simple averaging method in chapter 5. We identified the unique geometrical properties of measurement data from time-difference-of-arrival (TDoA) HM. By manipulating the data in different ways, we identified that the problem can be best solved by modeling it as a maximum likelihood estimation problem.

A difficult step in arriving at the proposed method was to reduce the dimensionality of the data so that we could find the mathematical model analytically. Having reduced the dimensionality, we used analysis to compute the expression for the family of density functions that represent the model.

We performed simulations where our method was applied to estimate the time-difference- of-arrival. Our experiments showed that our method can limit the error in the hyperbolic constraint to to 0.2 clock periods in at most 50 challenge-response rounds. Therefore, applying our technique for constructing the hyperbolic constraints is much better than simple averaging. We advocate the use of our method instead of the traditional approach to compute the hyperbolic constraints in HM.

Bibliography

[1] AeroScout. http://www.aeroscout.com, retrieved May 2012.

[2] BBN ADROIT Project. http://gnuradio.org/redmine/projects/gnuradio/wiki/BBN_Technologies_Internetwork_Research_ADROIT_Project, Retreived May 2012.

[3] G. Bella. What is Correctness of Security Protocols? *Journal of Universal Computer Science*, 14:2083–2106, 2008.

[4] S. Brands and D. Chaum. Distance-bounding protocols. In *Advances in Cryptology – EUROCRYPT '93*, volume 765 of *Lecture Notes in Computer Science*, pages 344–359. International Association for Cryptologic Research, Springer-Verlag, Berlin Germany, 1994.

[5] S. Capkun, M. Cagalj, and M. Srivastava. Secure Localization with Hidden and Mobile Base Stations. In *IEEE INFOCOM*, April 2006.

[6] S. Capkun and J. P. Hubaux. Secure Positioning of Wireless Devices with Application to Sensor Networks. In *IEEE INFOCOM*, March 2005.

[7] D. Cardenas and G. Arevalo. All Digital Timing Recovery and FPGA Implementation. *Jornadas de Ingeniera Electrica y Electronica (FIEE)*, November 2010.

[8] J. T. Chiang, J. J. Haas, and Y. Hu. Secure and Precise Location Verification Using Simultaneous Distance Bounding and Simulataneous Multilateration. In *second ACM Conference on Wireless Network Security*, March 2009.

[9] T. Cooklev, J. C. Eidson, and A. Pakdaman. An implementation of IEEE 1588 Over IEEE 802.11b for Synchronization of Wireless Local Area Network Nodes. *IEEE Trans. on Instrumentation and Measurement*, 56:1632–1639, 2007.

[10] I. Corp. HFA3861B Direct Sequence Spread Spectrum Baseband Processor. Technical Report Data Sheet FN4816.2, February 2002.

[11] C. Cremers, K. B. Rasmussen, and S. Capkun. Distane Hijacking Attacks on Distance Bounding Protocols. In *Cryptology ePrint Archive*, August 2011.

[12] R. E. Crochiere and L. R. Rabiner. Interpolation and Decimation of Digital Signals - A Tutorial Review. *Proceedings of the IEEE*, 69:300–331, 1981.

[13] DP83640 Synchronous Ethernet Mode: Achieveing Sub-nanosecond Accuracy in PTP Applications. In *Application Note 1730*. National Semiconductor, September 2007.

[14] G. Durisi and G. Romano. Simulation Analysis and Performance Evaluation of an UWB System in Indoor Multipath Channel. In *IEEE Conference on Ultra Wideband Systems and Technologies*, pages 255–258, 2002.

[15] R. Exel, G. Gaderer, and P. Loschmidt. A Novel, High-Precision Timestamping Platform for Wireless Networks. In *International Conference on Emerging Technologies and Factory Automation*, pages 1–8, September 2009.

[16] R. Exel, G. Gaderer, and P. Loschmidt. Localization of Wireless LAN Nodes Using Accurate TDoA Measurements. In *IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6, April 2010.

[17] R. J. Fontana and S. Gunderson. Ultra Wideband Precision Asset Location System. In *Conference on Ultra Wideband Systems and Technologies*, pages 147–150, May 2002.

[18] R. J. Fontana, E. Richley, and J. Barney. Commercialization of an Ultra Wideband Precision Asset Location System. In *IEEE Conference on Ultra Wideband Systems and Technologies*, pages 369–373, November 2003.

[19] F. M. Gardener. Interpolation in Digital Modems – Part I: Fundamentals. *IEEE Transactions on Communications*, 41:501–507, 1993.

[20] M. S. Gast. *802.11 Wireless Networks The Definitive Guide, O'Rielly*. April 2005.

[21] D. J. Geiger. High Resolution Time Difference of Arrival Using Timestamps for Localization in 802.11b/g Wireless Networks. In *Wireless Communications and Networking Conference (WCNC)*, pages 1–6, April 2010.

[22] IEEE. *IEEE Standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) And Physical Layer (PHY) Specifications*. Number 802.11-2007. Piscataway, NJ, June 2007.

[23] IEEE. *IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*. Number 1588-2008. Piscataway, NJ, Jan 2008.

[24] P. Jain and S. S. Lam. Modeling and Verification of Real Time Protocols for Broadcast Networks. *IEEE Transactions on Software Engineering*, SE-13:924–937, 1987.

[25] Juniper Research. Mobile Application Based Services – Applications, Forecasts and Opportunities. http://juniperresearch.com/reports.php?id=213, retrieved May 2012.

[26] J. Kannisto, T. Vanhatupa, M. Hannikainen, and T. Hamalainen. Software and Hardware Prototypes of the IEEE 1588 Precision Time Protocol on Wireless LAN. In *14th IEEE Workshop on Local and Metropolitan Area Networks*, September 2005.

[27] T. C. Karalar and J. Rabey. An RF ToF Based Ranging Implementation for Sensor Networks. In *IEEE International Communications Conference*, volume 7, pages 3347–3352, June 2006.

[28] S. Ko, J. Takayama, and S. Ohyama. Proposal of Generalized Vernier Effect and its Practical Advantage for RF Time-of-Flight Ranaging Betweeen Sensor Nodes in Wireless Sensor Networks. *Sensors and Actuators A: Physical*, 167:537–547, 2011.

[29] K. W. Kolodziej and J. Hjelm. *Local Positioning Systems*. CRC Press Taylor and Francis Group, 6000 Broken Sound Parkway NW, Boca Raton, FL, 2006.

[30] S. Lanzisera, D. Lin, and K. Pister. RF Time of Flight Ranging for Wireless Sensor Netowrk Localization. In *International Workshop on Intelligent Solutions in Embedded Systems*, June 2006.

[31] X. Li, K. Pahlavan, M. Latva-Aho, and M. Ylianttila. Comparison of Indoor Geolocation Methods in DSSS and OFDM Wireless LAN Systems. In *IEEE Vehicular Technology Conference (VTS)*, pages 3015–3020, September 2000.

[32] L. Litwin. Matched Filtering and Timing Recovery in Digital Receivers, A Practical Look at Methods for Signal Detection and Symbol Synchronization. In *Mobile Dev and Design Magazine*. http://mobiledevdesign.com/tutorials, retrieved September 2011.

[33] X. Liu and M. Ren. Location Cheating: A Security Challenge to Location-Based Social Network Services. In *International Conference on Distributed Computing Systems (ICDCS)*, pages 740–749, June 2011.

[34] P. Loschmidt, G. Gaderer, and T. Sauter. Clock Synchronization for Wireless Positioning of COTS Mobile Nodes. In *IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication*, pages 64–69, October 2007.

[35] D. D. MacCrady, L.Doyle, H. Forstrom, T. Dempsey, and M. Martorana. Mobile Ranging Using Low Accuracy Clocks. In *IEEE Transactions on Microwave Theory and Techniques*, pages 951–958, June 2000.

[36] T. Manodham, L. Loyola, and T. Miki. A Novel Wireless Positioning System for Seamless Internet Connectivity Based on the WLAN Infrastructure. *International Journal of Wireless Personal Communications*, 44:295–309, 2008.

[37] E. B. Mazomenos, D. D. Jager, J. S. Reeve, and N. M. White. A Two-Way Time-of-Flight Ranging Scheme for Wireless Sensor Networks. In *the 8th European Conference on Wireless Sensor Networks (EWSN)*, February 2011.

[38] J. Micheel, S. Donnelly, and I. Graham. Precision Timestamping of Network Packets. In *ACM SIGCOMM INternet Measurement Workshop IMW*, November 2001.

[39] D. Moore, J. Leonard, D. Rus, and S. Teller. Robust Distibuted Network Localization With Noisy Range Measurements. In *ACM SenSys*, pages 50–61, November 2004.

[40] D. Munoz, F. B. Lara, C. Vargas, and R.Enriquez-Caldera. *Position Location Techniques and Applications*. Elsevier Science, ISBN:01237435332, Philadelphia, PA, April 2009.

[41] Precision PHYter. http://www.national.com/pf/DP/DP83630.html, retrieved April 2011.

[42] A. Pakdaman, T.Cooklev, and J.C.Eidson. IEEE 1588 over IEEE 802.11b for. Synchronization of Wireless Local. Area Network Nodes. In *2004 Conference on IEEE 1588, Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, pages 116–127, November 2004.

[43] S. Parichha. Ubiquitous Peer-Proximity Awareness in Mobile Environments. In *Mobile Future Forward Summit, Student Paper Contest*, September 2010.

[44] S. Parichha. On Secure Localization Without Simultaneous Challenges. Master's thesis, University of California, Riverside, USA, 2011. Adviser-Molle, Mart.

[45] S. Parichha and M. Molle. Localization and Clock Synchronization Need Similar Hardware Support in Wireless LANs. In *IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication*, September 2008.

[46] S. Parichha and M. Molle. More (Messages) is Less (Accuracy) in Localization. In *Military Communications Conference (MILCOM)*, November 2011.

[47] A. Pasztor and D. Veitch. High Precision Active Probing for Internet Measurement. In *INET, The Internet Society*, 2001.

[48] N. B. Prityantha, A. Chakraborty, and H. Balakrishnan. The Cricket Location-Support System. In *ACM Anhual International Conference on Mobile Computing and Ntework-ing*, pages 32–43, August 2000.

[49] M. A. Rahman. Wireless Disaster Area Emergency Network (W-DAEN) to Save Human Lives. In *IEEE Presidents' Change the World Competition – Outstanding Humanitarian Project*, 2011.

[50] K. B. Rasmussen and S. Capkun. Location Privacy of Distance Bounding Protocols. In *ACM Conference on Computer and Communications Security*, pages 149–160, October 2008.

[51] K. B. Rasmussen and S. Capkun. Realization of RF Distance Bounding. In *USENIX Security Symposium*, August 2010.

[52] Ekahau. http://www.ekahau.com, retrieved May 2012.

[53] D. Rosselot. Simple, Accurate Time Synchronization in an Ethernet Physical Layer Device. In *IEEE International Symposium for Precision Clock Synchronization for Measurement, Control and Communication (ISPCS 2007)*, October 2007.

[54] A. Saha and M. Molle. Localization with Witnesses. In *Proc. 1st International Conference on New Technologies, Mobility and Security (NTMS 2007)*, May 2007.

[55] A. K. Saha. *Cross Layer Techniques to Secure Peer-to-Peer Protocols for Location, Adjacency, and Identity Verification.* PhD thesis, USA, 2006. Adviser-Molle, Mart.

[56] S. Saroiu and A. Wolman. Enabling New Mobile Applications with Location Proofs. In *Proc. of 10th Workshop on Mobile Computing Systems and Applications*, February 2009.

[57] Clock Solutions for WiFi (IEEE 802.11). In *Application Note 70.* http://www.pericom.com/pdf/applications/AN070.pdf, Retreived May 2012.

[58] N. Sastry, U. Shankar, and D. Wagner. Secure Verification of Location Claims. In *ACM workshop on Wireless Security (WiSe 2003)*, pages 1–10. ACM Press, 2003.

[59] A. Savvides, C. C. Han, and M. Srivastava. Dynamic Fine-Grained Localization in Ad-hoc Networks of Sensors. In *ACM International Conference on Mobile Computing and Networking (MOBICOM)*, pages 166–169, July 2001.

[60] A. Savvides, H. Park, and M. Srivastava. The Bits and Flops of the N-hop Multilateration Primitive for Node Localization Problems. In *First ACM International Workshop on Sensor Networks and Applications*, September 2002.

[61] V. Shmatikov and M. Wang. Secure Location Verification of Location Claims with Simultaneous Distance Modification. In *12th Asian Computing Science Conference on Advances in Computer Science: Computer and Network Security*, 2007.

[62] D. Singelee and B. Preneel. Security Analysis of the Rasmussen-Capkun CRCS Distance Bounding Protocol. Technical report, Computer Security and Industrial Cryptography Group, KU Leuven, 2011.

[63] B. Thorbjornsen, N. M. White, A. D. Brown, and J. S. Reeve. Radio Frequency (RF) Time-of-Flight Ranging for Wireless Sensor Networks. *Measurement Science and Technology*, 21(3), 2010.

[64] A True System-on-Chip Solution for 2.4 GHz IEEE 802.15.4/Zigbee. In *Datasheet for the TI CC2430.* Texas Instruments, Retreived May 2012.

[65] N. Tippenhauer and S. Capkun. ID-Based Secure Distance Bounding and Loclaization. *Lecture Notes in Computer Science*, 5789:621–636, 1981.

[66] J. Werb, M. Newman, V. Berry, S. Lamb, D. Sexton, and M. Lapinski. Improved Quality of Service in IEEE 802.15.4 Mesh Networks. In *International Workshop on Wireless Industrial Automation*, March 2005.

[67] Multilateration. In *Wikipedia, The Free Encyclopedia.* http://en.wikipedia.org/wiki/Multilateration, retrieved December 2010.

[68] B. Xu, R. Yu, G. Sun, and Z. Yang. Whistle: Synchronization-Free TDOA for Localization. In *International Conference on Distributed Computing Systems (ICDCS)*, June 2011.

[69] R. Yamasaki, A. Ogino, T. Tamaki, T. Uta, N. Matsuzawa, and T. Kato. TDoA Location System for IEEE 802.11b WLAN. In *IEEE Wireless Communications and Networking Conference (WCNC)*, 2005.

[70] M. Youssef, A. Youssef, C. Reiger, U. Shankar, and A. Agrawala. PinPoint: An Asynchronous Time-Based Location Determination System. In *MobiSys*, 2006.

Appendix I

Let $\mathcal{R}$ be the planar region containing all the verifiers in V, and the prover p. A triangle $\langle x, y, z\rangle$ is defined as a verification triangle if all points in a circle with center p and radius ϵ, for $\epsilon > 0$, are strictly inside $\langle x, y, z\rangle$.

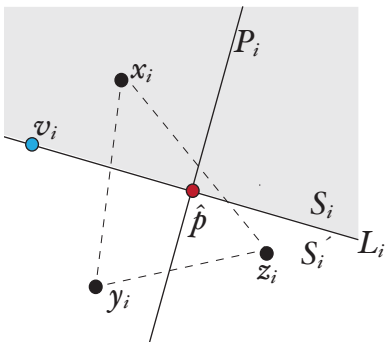
Lemma 1 *During the execution of elliptical multilateration (EM), let v_i be the lead verifier in the ith challenge-response round, and L_i be the straight line passing through v_i and p. Let $\langle x_i, y_i, z_i\rangle$ be a verification triangle such that p passes the δ-test with it. If P_i the perpendicular to L_i at p, then verifiers in $\langle x_i, y_i, z_i\rangle$ can constrain p's location to the half-plane (defined by P_i) containing v_i.*

Proof: Consider the two half-planes on either side of the line L_i. Let us denote one of the half planes as S_i, and the opposite half-plane as S_i' as shown in Fig I.1(a). Since p is contained within triangle $\langle x_i, y_i, z_i\rangle$, each of the two half planes must contain at least one vertex of $\langle x_i, y_i, z_i\rangle$. Suppose vertex x_i is in the half plane S_i, and vertex y_i is in the half plane S_i' as shown in Fig. I.1(a). Although the third vertex z_i is shown to be in S_i' in the figure, it may instead be in S_i. The location of the third vertex does not affect this proof.

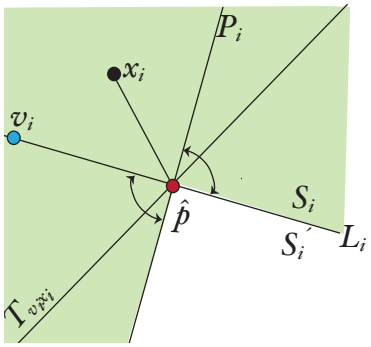
In elliptical multilateration (EM), each witness (passive verifier) uses its measurement to impose an elliptical constraint on p's location. The elliptical constraint passes through the claimed location $\hat{p}$, and its foci coincide with the locations of the lead verifier and the witness itself. Let us consider the witness x_i from $\langle x_i, y_i, z_i \rangle$, which is in the half-plane S_i. Using its measurement from the ith round of challenge-response, x_i can constrain the prover to an ellipse $\mathcal{E}_{v_i x_i}$ with foci at x_i and v_i. Note that for all possible positions of x_i in the S_i half plane, the angle $\angle v_i p x_i$ can vary between 0° and 180°. Let the tangent to $\mathcal{E}_{v_i x_i}$ at p be denoted as $T_{v_i x_i}$, as shown in Fig I.1(b). From the geometric properties of a tangent to an ellipse, we know that $T_{v_i x_i}$ is perpendicular to the bisector of $\angle v_i p x_i$. When $\angle v_i p x_i = 0°$, then $T_{v_i x_i}$ is coincident with P_i. When $\angle v_i p x_i = 180°$, then $T_{v_i x_i}$ is coincident with L_i. Since ellipse $\mathcal{E}_{v_i x_i}$ must always be on the side of $T_{v_i x_i}$ that contains the lead verifier v_i, $\mathcal{E}_{v_i x_i}$ must lie in the shaded region shown in Fig. I.1(b).

Next , we consider witness y_i which is in the half plane S_i'. Using its measurement in the ith round of challenge-response, y_i can impose an elliptical constraint $\mathcal{E}_{v_i y_i}$ on the prover's location, such that $\mathcal{E}_{v_i y_i}$ passes through $\hat{p}$ and its foci coincide with the locations of v_i and y_i. Let $T_{v_i y_i}$ be the tangent to $\mathcal{E}_{v_i y_i}$ at $\hat{p}$. Similar to the case for witness x_i, depending on the location of y_i in the S_i' half plane, the angle $\angle v_i p y_i$ can vary between 0° and 180°. When $\angle v_i p y_i = 0°$, then $T_{v_i y_i}$ is coincident with P_i. When $\angle v_i p y_i = 180°$, then $T_{v_i y_i}$ is coincident with L_i. Since ellipse $\mathcal{E}_{v_i y_i}$ must always be on the side of $T_{v_i y_i}$ that contains the lead verifier v_i, $\mathcal{E}_{v_i y_i}$ must lie in the shaded region shown in Fig. I.1(c).

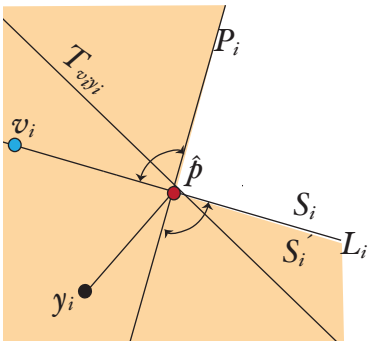
Since the third witness z_i must either be in S_i or in S_i', by similar reasoning, the ellipse $\mathcal{E}_{v_i z_i}$ must either be in the shaded region shown in Fig I.1(b) or in the shaded region shown in Fig. I.1(c). The intersection of the shaded regions in Fig. I.1(b) and Fig. I.1(c)



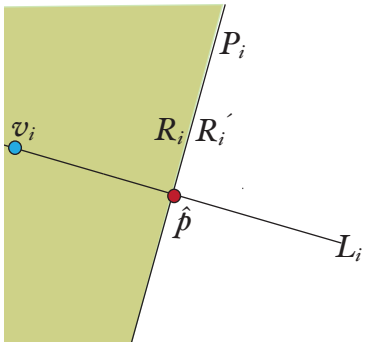
(a) Regions S_i and S_i' defined by line L_i



(b) Shaded region shows constraint on p's location formed by x_i



(c) Shaded region shows constraint on p's location formed by y_i



(d) Verifiers in $\langle x_i, y_i, z_i \rangle$ constrain p to the shaded half-plane R_i

Figure I.1: Each verifier in the verification triangle can constrain the prover to a half plane. This half plane is defined by the tangent at $\hat{p}$, to the elliptical constraint that the verifier forms with the lead verifier.

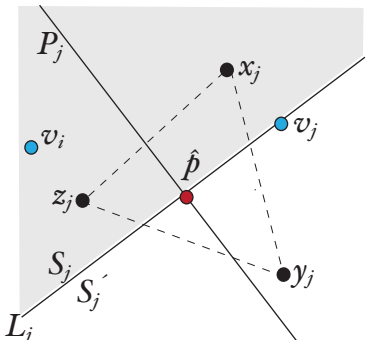
is the half plane R_i defined by the line P_i, which contains the lead verifier v_i. R_i is shown in Fig. I.1(d). QED.

Lemma 2 *Let the lead verifiers in the ith and jth challenge-response rounds of EM, be v_i and v_j. If $\langle x_i, y_i, z_i \rangle$ and $\langle x_j, y_j, z_j \rangle$ are valid verification triangles for the ith and jth rounds, then two witnesses from $\langle x_i, y_i, z_i \rangle$ and two witnesses from $\langle x_j, y_j, z_j \rangle$ can together contain the prover to a region $\mathcal{R}_{\rangle|}$ within (the non-reflex angle) $\angle v_i p v_j$.*

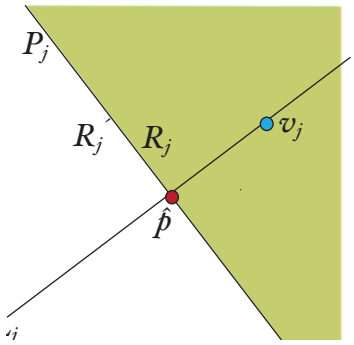
Proof: First, let us consider the ith challenge-response round in which verifier v_i acts as the lead verifier. From Lemma 1, we know that the two witnesses in $\langle x_i, y_i, z_i \rangle$ can together constrain p to the half-plane R_i as shown in Fig. I.1(d). Next we consider the jth challenge-response round in which a different verifier v_j acts as the lead verifier. Let the line joining v_j and $\hat{p}$ be L_j, such that L_j partitions $\mathcal{R}$ into two half planes: S_j and S'_j, as shown in Fig. I.2(a).

Let P_j is the perpendicular to L_j at p. Similar to Lemma 1, two witnesses in $\langle x_j, y_j, z_j \rangle$, located on either side of L_j, can constrain the prover to the shaded half plane $\mathcal{R}_j$ defined by P_j, which contains the lead verifier v_j.

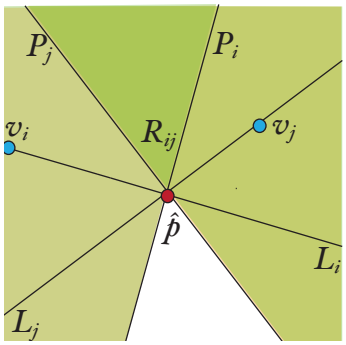
The superimposition of the constraints formed in the ith and jth round are shown in Fig. I.2(c). The intersection of the two-half planes $\mathcal{R}_i$ and $\mathcal{R}_j$ is the region $\mathcal{R}_{ij}$ bounded by the the lines P_i and P_j as shown in Fig. I.2(d). Notice that $\mathcal{R}_{ij}$ is entirely within the non-reflex angle $\angle v_1 p v_2$ formed by the prover and the two lead verifiers from both rounds. QED



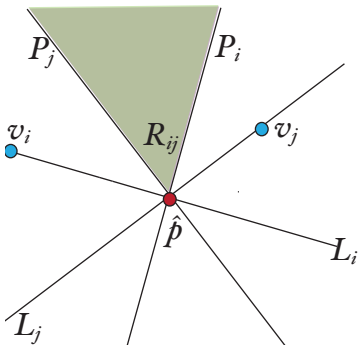
(a) Regions S_j and S'_j defined by line L_j



(b) Prover is constrained to the shaded half plane R_j in the jth round by two witnesses in $\langle x_j, y_j, z_j \rangle$.



(c) Superposition of constraints formed by the witnesses in the ith and jth rounds.



(d) Over two different challenge-response rounds, the verifiers can constrain the prover to the shaded region $\mathcal{R}_{ij}$ within the non-reflex angle $\angle v_i p v_j$.

Figure I.2: The intersection of the constraints formed by the verifiers from two verification triangles in two separate challenge-response rounds is contained in the non-reflex angle formed by $\hat{p}$ with the two lead verifiers.

Appendix II

Let $S = \{(x, y) : 0 \leq x < 1, 0 \leq y < 1\}$ be a unit square in the two-dimensional coordinate space. For any $r, 0 \leq r < 1$, the function

$$F_r(x, y) = \begin{cases} 0 & 0 \leq y \leq x - r \\ 1 & x - r < y \leq 1 + x - r \\ 2 & 1 + x - r < y < 1 \end{cases} \tag{II.1}$$

partitions S into three distinct regions.

Lemma 3 *For any $c \in [0, 1)$, let $L_c = \{(c, y) : 0 \leq y < 1\}$ be a vertical line segment in S. Then, (a) we cannot find two points (c, y_0) and (c, y_2) on L_c, such that $F_r(c, y_0) = 0$ and $F_r(c, y_2) = 2$ (b) $F(c, 0) \neq F(c, 1^-)$ (c) it is not possible to have $F_r(c, 0) = 2$.*

Proof for (a):

Let us assume we can find two points: (c, y_2) and (c, y_0) on line segment L such that $F_r(c, y_0) = 0$ and $F_r(c, y_2) = 2$.

Since $F_r(c, y_0) = 0$, we must have

$$0 \leq y_0 \leq c - r \tag{II.2}$$

Similarly, because $F_r(c, y_2) = 2$, *we must have*

$$1 + c - r < y_2 < 1 \tag{II.3}$$

Substituting for $c - r$ *from Eq.(II.2) in Eq.(II.3), we obtain*

$$1 + y_0 < y_2 \tag{II.4}$$

Since $0 \leq y_0$ *and* $y_2 < 1$, *inequality (II.4) cannot be true, which proves (a).*

Proof for (b):

We consider two cases:

(i) *Suppose* $F_r(c, 0) = 0$. *Then, we have*

$$0 \leq c - r \tag{II.5}$$

Since $\{c, r\} \in [0, 1)$, *we also have*

$$-1 < c - r < 1 \tag{II.6}$$

Combining inequalities (II.5) and (II.6), we get

$$0 \leq (c - r) < 1 \tag{II.7}$$

Therefore, we can find an $\epsilon_0 > 0$ *such that* $c - r = 1 - \epsilon_0$. *Let* $y = 1^- = 1 - \epsilon$, *where*

$0 < \epsilon < \epsilon_0$. *We have*

$$1 - \epsilon_0 < 1 - \epsilon \Rightarrow c - r < y = 1 - \epsilon_0 \tag{II.8}$$

which implies $F_r(c, 1^-) \neq 0 = F_r(c, 0)$.

(ii) *If* $F_r(c, 0) = 1$, *then we have*

$$c - r < 0 \leq 1 + c - r \tag{II.9}$$

Since $c - r < 0$, *for any* $c - r$, *we can find an* $\epsilon_1 > 0$ *such that* $c - r = -\epsilon_1$. *Therefore, II.9 can be rewritten as*

$$-\epsilon_1 < y \leq 1 - \epsilon_1 \tag{II.10}$$

Let $y = 1^- = 1 - \sigma$, *where* $0 < \sigma < \epsilon_1$. *If* $F_r(c, 1^-) = 1$ *is true, from II.10, we must have*

$$1 - \sigma = y \leq 1 + c - r \Rightarrow 1 - \sigma \leq 1 - \epsilon_1 \tag{II.11}$$

which is a contradiction. Therefore, $F_r(c, 1^-) \neq 1 \equiv F_r(c, 0)$.

Proof for (c):

Let $F_r(c, 0) = 2$. *Then, we must have*

$$1 + c - r < 0 \Rightarrow c - r < -1 \tag{II.12}$$

which contradicts inequality (II.6).

Observation 1 *We proved that either* $F_r(c, 0) = 0$ *and* $F_r(c, 1^-) = 1$, *or,* $F_r(c, 0) = 1$ *and*

$F_r(c, 1^-) = 2$. *Therefore,*

$$F_r(c, 1^-) - F_r(c, 0) = 1 \tag{II.13}$$

Lemma 4 *Let L_a and L_b be two vertical line segments in S. For all $y \in [0, 1)$, the expression $F_r(a, y) - F_q(b, y)$ can take on (a) one of only two distinct values if $(a - b) \bmod 1 \neq (r - q) \bmod 1$ (b) and only a single value if $(a - b) \bmod 1 \equiv (r - q) \bmod 1$.*

Proof:

Let

$$y_1 = max\{y : F_r(a, y) = F_r(a, 0)\}$$

From Lemma:1, we know that $F_r(a, 0) \neq 2$, in which case we can apply Eq. (II.1) to show that

$$y_1 = \begin{cases} a - r & 0 \leq a - r \\ 1 + a - r & a - r < 0 \end{cases}$$

and thus,

$$y_1 \equiv (a - r) \bmod 1 \tag{II.14}$$

Similarly,

$$\begin{aligned} y_2 &= max\{y : F_q(b, y) = F_q(b, 0)\} \\ &\equiv (b - q) \bmod 1 \end{aligned} \tag{II.15}$$

From Lemma:1, we also know that

$$F_r(x,y) \in \{F_r(a,0), F_r(a,1^-)\}$$

Therefore,

$$F_r(a,y) - F_q(b,y) = \begin{cases} F_r(a,1^-) - F_q(b,1^-) & max\{y_1,y_2\} < y(a) \\ F_r(a,1^-) - F_q(b,0) & y_1 < y \leq y_2(b) \\ F_r(a,0) - F_q(b,1^-) & y_2 < y \leq y_1(c) \\ F_r(a,0) - F_q(b,0) & y \leq min\{y_1,y_2\}(d) \end{cases} \tag{II.16}$$

Proof for (a):

Expressions II.16(a) and II.16(d) evaluate to the same value because of (II.13) in observation 1. Furthermore, we find that expressions II.16(b) and II.16(c) are mutually contradictory. So both cannot be true. Therefore, $F_r(a,y) - F_q(b,y)$ can take on only one of two distinct values when $y_1 \neq y_2$.

Proof for (b):

If $y_1 = y_2$, then neither condition II.16(b) nor condition II.16(c) can be true. This occurs when

$$(a-b) \bmod 1 \equiv (r-q) \bmod 1$$

Therefore,

$$F_r(a,y) - F_q(b,y) = \begin{cases} v_1 & min\{y_1,y_2\} < y \leq max\{y_1,y_2\} \\ v_0 & otherwise \end{cases} \tag{II.17}$$

where v_1 and v_0 are the two possible values for a given four tuple $\{a,b,r,q\}$. QED.

Observation 2 *If y is chosen as a uniform $[0,1)$ random variable that is independent of a, b, r and q, then the probability of occurrence of v_1 is given by*

$$P(v_1|a,b,r,q) = |y_1 - y_2| \tag{II.18}$$

and that of v_0 is given by

$$P(v_0|a,b,r,q) = 1 - |y_1 - y_2| \tag{II.19}$$

Lemma 5 *Let the constants m and n satisfy:*

$$-min(a,b) \leq m < 1 - max(a,b)$$

$$-min(r,q) \leq n < 1 - max(r,q)$$

$$-min(y_1,y_2) \leq m - n < 1 - max(y_1,y_2)$$

Then, the values $a' = a + m$, $b' = b + m$, $r' = r + n$ and $q' = q + n$ are all within the range $[0,1)$. Furthermore, if we apply Lemma 2 to the system ${L_a}', {L_b}', {F_r}'(x,y), {F_q}'(x,y)$ to obtain y_1' and y_2', then $y_1 - y_2 = y_1' - y_2'$. In this case, we say $\{a,b,r,q\}$ and $\{a',b',r',q'\}$ belong to the same equivalence class of 4-tuples.

Proof: *From (II.14) we have*

$$\begin{aligned} {y_1}' &\equiv ((a+m) - (r+n)) \bmod 1 \\ &\equiv ((a-r) \bmod 1 + (m-n)) \end{aligned}$$

Substituting $y_1 \equiv (a - r) \bmod 1$*, we obtain*

$$y_1{}' \equiv (y_1 + (m - n)) \bmod 1$$

Similarly,

$$y_2{}' \equiv (y_2 + (m - n)) \bmod 1$$

If

$$(y_1 + (m - n)) \bmod 1 \equiv y_1 + (m - n) \tag{II.20}$$

and

$$(y_2 + (m - n)) \bmod 1 \equiv y_2 + (m - n) \tag{II.21}$$

then

$$\begin{aligned} y_1{}' - y_2{}' &= y_1 + (m - n) - y_2 - (m - n) \\ &= y_1 - y_2 \end{aligned} \tag{II.22}$$

as required to satisfy the Lemma.

But condition (II.20) is satisfied if and only if

$$0 \leq y_1 + (m - n) < 1$$

$$\Rightarrow -y_1 \leq m - n < 1 - y_1 \tag{II.23}$$

Similarly, condition (II.21) is satisfied if and only if

$$-y_2 \leq m - n < 1 - y_2 \tag{II.24}$$

Thus, to satisfy both conditions (II.23) and (II.24), we require

$$-min(y_1, y_2) \leq m - n < 1 - max(y_1, y_2)$$

QED.

Observation 3 *If m and n satisfy the conditions of Lemma 3, then*

$$P(v_1|a, b, r, q) = P(v_1|a', b', r', q')$$

and

$$P(v_0|a, b, r, q) = P(v_0|a', b', r', q')$$

Thus, the probabilities of occurrence for v_1 and v_0 depend only on the differences $(a-b)$ and $(r-q)$, and are independent of their individual values. This allows us to apply convenient parameter shift to the $4-$tuple $\{a, b, r, q\}$ to simplify the task of evaluating these probabilities and therefore, our solution will apply equally well to every $4-$tuple where m and n satisfy the conditions stated in Lemma 3.

Observation 4 *Because these parameter shifts do not change the probabilities of occurrences for v_1 and v_0, we can combine 4-tuples from the same class. In particular, if every point in the region $\mathbb{R} = \{(m, n) : m_1 \leq m < m_2, n_1 \leq n < n_2\}$ satisfies the conditions of*

Lemma 3, then

$$\int_{(m,n)\in\mathbb{R}} P(v_1|a',b',r',q') = |\mathbb{R}| \cdot P(v_1|a,b,r,q) \tag{II.25}$$

which is the product of aggregation range and the probability for a representative 4*-tuple from that range.*

Glossary

C_x^y Timestamp for arrival of a message sent from x at y.

$D(x, y)$ Time normalized distance between entities x and y.

$T_p(v, v)$ Total time for executing a challenge-response echo initiated by verifier v. The challenge is sent by v, prover p receives the challenge, and computes the response, and v observes the response.

$T_p(v, w)$ Total time for executing a challenge-response relay. The challenge is sent by v, prover p receives the challenge, and computes the response, and a different verifier w observes the response.

Δ_v Time taken by prover p to respond in a challenge-response echo initiated by verifier v.

$\Delta_{\{v,w\}}$ Time taken by the prover p to respond in a challenge-response relay, where verifier v sends the challenge, and verifier w observes the response.

$\bar{v}$ The verifier chosen to send the challenge in some round of a SIMO localization protocol.

δ Measurement error in the running time of a challenge-response dialog.

$\hat{T}(v, v)$ Expected time for executing a challenge-response echo by verifier v, when the claimed prover location is $\hat{p}$.

$\hat{\Delta}$ The minimum possible response delay of a prover.

$\hat{p}$ The location claimed by prover(s).

$\mathcal{C}_v$ Circular constraint formed by verifier v on the prover's location after it executed a challenge-response echo with it.

$\mathcal{E}_{vw}$ Elliptical constraint on the prover's location formed by verifiers v and w when they observe the prover's response simultaneously in some challenge-response round of EM.

$\mathcal{H}_{vw}$ Hyperbolic constraint on the prover's location formed by verifiers v and w when they observe the prover's response simultaneously in some challenge-response round of HM.

$\mathcal{R}$ Planar region containing all the verifiers in V, the prover p and the claimed location $\hat{p}$.

∇_v The response delay of the prover as perceived by verifier v, in a challenge-response echo initiated by v.

$\{u, v, w, z, ...\} \in V$ verifiers in V, where V represents the set of verifiers.

e_x^y Event of arrival of a message sent by entity x, received by entity y.

p Resource-constrained prover who possesses a single radio and an omnidirectional antenna.

p^* Resourceful prover who possesses multiple radios with directional antennas.