ABSTRACT

Title of dissertation	TIME-BASED LOCATION TECHNIQUES USING INEXPENSIVE, UNSYNCHRONIZED CLOCKS IN WIRELESS NETWORKS
	Matthew Yew Mun Mah, Doctor of Philosophy, 2011
Dissertation directed by	Professor Ashok Agrawala
	Department of Computer Science

The ability to measure location using time of flight in IEEE 802.11 networks is impeded by the standard clock resolution, imprecise synchronization of the 802.11 protocol, and the inaccuracy of available clocks. To achieve real-time location with accuracy goals of a few meters, we derive new consensus synchronization techniques for free-running clocks. Using consensus synchronization, we improve existing time of arrival (TOA) techniques and introduce new time difference of arrival (TDOA) techniques. With this common basis, we show how TOA is theoretically superior to TDOA. Using TOA measurements, we can locate wireless nodes that participate in the location system, and using TDOA measurements, we can locate nodes that do not participate. We demonstrate applications using off-the-shelf 802.11 hardware that can determine location to within 3m using simple, existing optimization methods. The synchronization techniques extend existing ones providing distributed synchronization for free-running clocks to cases where send times cannot be controlled and adjusted precisely, as in 802.11 networks. These location and synchronization techniques may be applied to transmitting wireless nodes using any communication protocol where cooperating nodes can produce send and receive timestamps.

TIME-BASED LOCATION TECHNIQUES USING INEXPENSIVE, UNSYNCHRONIZED CLOCKS IN WIRELESS NETWORKS

by

Matthew Yew Mun Mah

Dissertation submitted to the Faculty of the Graduate School of the University of Maryland, College Park in partial fulfillment of the requirements for the degree of Doctor of Philosophy
2011

Contents

Abbreviations

ACK Acknowledgement

AOA Angle Of Arrival

AP Access Point

CTS Clear To Send - response to RTS packet

DOP Dilution Of Precision

GPS Global Positioning System

IEEE Instituite of Electrical and Electronics Engineers

LBS location-based service

LORAN Long Range Aid to Navigation

LOS Line of sight

MAC Medium Access Control

MIMO Multiple-Input Multiple-Output

NTP Network Time Protocol

RBS Reference Broadcast Synchronization

RF Radio Frequency

RTS Request To Send

SIFS Short InterFrame Spacing - minimum time between packets in 802.11 protocol

SNTP Simple Network Time Protocol

TDOA Time Difference of Arrival

TOA Time of Arrival

Chapter 1

Introduction

The context-aware paradigm of computing exploits available context such as user preferences, time, and location by recognizing that context determines what information is relevant to a user [72, 31]. For example, if there is a traffic jam on the freeway, a car commuter may wish to adjust her route, but a subway rider may continue as usual. The general notion of context is intentionally vague to encompass all possible relevant information. Clearly location is a major component of context for most situations. Here we focus on technologies related to location context.

Location-based services (LBS), applications that use location context, are now an important feature of mobile computing for the military, smartphones, and automobiles. Some examples of location-based services are weapon system guidance, ship navigation to prevent running aground in shallow waters, turn-by-turn navigation directions for automobiles, and providing addresses of nearest gas stations or restaurants. Location-based services do not generate location context themselves; they require a separate service to supply location.

The accuracy and coverage of the base location service determine what applications can be supported. Coarse location accuracy is sufficient for applications such as weather forecasts or news feeds, while finer location accuracy is necessary for turn-by-turn navigation directions. Increasing the available accuracy or coverage of location services can improve existing LBS performance and enable new applications. For example, GPS currently assists in-flight aircraft navigation, but its altitude accuracy is not sufficient for aircraft landing systems in poor visibility conditions. Improved differential GPS antennas at airports are expected to improve altitude accuracy to provide this capability [53].

Development of new location-based services is ongoing due to the increasing availability of affordable location services; many of the winners of the first Android Developer Challenge for developing smartphone applications are location-based services [5]. The first tier of challenge winners provide services such as tracking carbon footprints and suggesting transportation alternatives, facilitating cab pickups, managing phone settings to turn off ringers in courtrooms or at bedtime, and finding the best nightlife spots. New applications are expected as the area matures.

Clearly the enabler for such location-based services is the availability of location information. Let us consider how location information can be obtained.

1.1 Location Technologies

Most location technologies require an active device, a target node, whose location is to be determined. The technology may also deploy a number of anchor nodes, devices with known location. The location of a node is determined in a 2D or 3D coordinate system, which may be absolute (e.g. latitude and longitude) or relative, defined for an application locally.

The techniques for finding location can be broadly categorized into angle of arrival (AOA), inertial tracking, signal strength, and time-based techniques. These techniques may be combined to create hybrid systems, but the four categories can be examined independently.

The angle of arrival technique [50, 64] uses triangulation to locate target nodes. Each anchor node measures the angle of arrival for incoming signals from the target node, which defines the direction of a ray starting at that anchor node and extending into space. Intersecting the rays from multiple anchors provides the location of the target node generating the signal. The chief drawbacks of angle of arrival techniques are the expense of the antennas or sensors to measure AOA, the accuracy with which angles can be measured, and the effects of reflections on angle of arrival. Further, for a fixed angle of arrival accuracy, location accuracy decreases with increasing distances between the target node and anchor nodes.

Inertial tracking techniques use accelerometers and gyroscopes to track nodes starting from known locations [39]. Acceleration vectors are integrated over time to compute location. A significant advantage of this approach is that all tracking hardware can be placed in a single, self-contained unit. The main disadvantage is that error in measuring acceleration, particularly from human movement [56, 88], results in location error accumulation over time. For this reason, inertial tracking is often combined with other information sources in hybrid systems [23, 46].

Signal strength techniques to determine location often use fingerprinting [10, 86]. Fingerprinting assumes that the vectors of signal strength values from anchor nodes will be sufficiently different in the area of interest to distinguish between locations. Location from fingerprinting has two phases, the calibration phase and the online phase. In the calibration phase, the vector of signal strength values for signals from anchor nodes is recorded at chosen locations throughout the area of interest to construct a ”radio map”. In the online phase, a target node compares its current vector of signal strength values with the radio map and interpolates to determine its most likely location. The chief drawback of signal strength techniques is the extensive resources required for calibration.

Other signal strength techniques avoid the calibration phase at the expense of using many anchor nodes. The simplest technique places a target node at the location of its closest anchor node, which is determined by the strongest signal strength value at the target node. As the resolution of this technique is limited by the available anchor node locations, a large number of anchor nodes is required to determine location accurately. Another variant uses network connectivity to a regular grid of anchor nodes [14]. Assuming a nominal transmission range, the set of anchor nodes visible to a target node will be centered at that target's location. This grid technique requires a large number of anchor nodes to cover an area, and its location accuracy is directly related to the anchor node spacing.

Time-based location techniques measure signal propagation time to determine distances between target nodes and anchor nodes. The signals may be acoustic [17], ultrasound [67, 41], or radio [21, 22, 32, 42, 52, 57, 60, 85, 87, 89, 30].[1] The most widespread time-based location service is the NAVSTAR Global Positioning System (GPS) [39]. GPS is frequently used as the location service for mobile phones, ship navigation, car navigation systems, etc., and it achieves accuracy of approximately 4m [76]. The primary shortcoming of GPS is coverage. Although the GPS operates all over the world, it cannot service most indoor or "urban canyon" environments where receivers cannot communicate well with satellites. Alternative technologies are necessary to supplement GPS in these conditions to maintain location capability.

1.2 Time-based Location

Time-based location techniques are based on the following simple equation describing the distance D traveled by a signal in time t propagating through space at speed v.

$$D = v * t \tag{1.1}$$

[1]Descriptions of these systems are in Appendix B.

The speed v determines the distance accuracy for a given timing accuracy. For underwater acoustic signals and ultrasound signals in air, $v \approx 1500$ m/s and $v \approx 330$ m/s respectively. At these speeds, if the timing error is within 1μs for a single measurement, the distance accuracy will be on the order of millimeters. For radio transmissions traveling at the speed of light, $v \approx 3 * 10^8$ m/s, if the timing error is within 1 μs for a single measurement, we can only measure distance with an accuracy of 300 m. If the timing error is within 1 ns, we can measure distance with an accuracy of 0.3 m.

The simplest time-based technique measures the round trip time for messages between two nodes using the clock at only one node. The first node simultaneously sends a message to a second node and records the time the message is sent, the send timestamp. The second node responds "immediately" after receiving this message, and the first node records the time the return message is received, the receive timestamp. The difference between these two timestamps is the propagation time for the round trip, which is twice the distance between the nodes. Note that it is unnecessary to know the absolute time; it is sufficient to know the time difference. More complex techniques remove the requirement that the second node respond immediately but require clocks at each node. When using multiple clocks, one at each node, to make timing measurements, they need to be synchronized.

1.2.1 Synchronization

We note that each node can only read its local clock, which runs independently of all other clocks. Even if timestamping were perfectly accurate with infinite precision, any calculation using multiple clocks must reconcile differences in clock start times and differences in each clock's frequency. Clock synchronization, examined in detail in Chapters 3 and 4, addresses the problem of multiple, independent clocks.

There are several distinct ways to perform synchronization. The most common way is to make physical adjustments to clock offsets and frequency to either agree with a master clock as in NTP [62] and IEEE 1588 [1, 33], or to agree with a system average [43, 55, 29]. The alternative to physical adjustment is to map local clock times to virtual clock times. The virtual clock mapping may depend on the pair of clocks being compared [34, 78, 85, 57], or there may be one mapping for each clock for use with all other clocks in the system.

Synchronization accuracy depends on both the available clock oscillators[2] and the network environment. NAVSTAR satellites are synchronized to the nanosecond level [69], a capability enabled by the precision and accuracy of atomic clocks. For the inexpensive quartz crystal oscillators used in most computer hardware, conventional time synchronization techniques such as NTP [62] provide time precision only on the order of milliseconds over the Internet. With the same quartz crystal oscillators over local networks, IEEE 1588 synchronization [1, 33] can achieve submicrosecond accuracy.

1.2.2 Distance Measurements and Location

There are two types of time-based measurements involving distances, time of arrival (TOA) and time difference of arrival (TDOA). TOA measures the distance between two nodes, while TDOA measures the difference of the distances from two nodes to a third node. TOA measurements require more control over nodes than TDOA; there are some conditions under which TDOA can be measured, but TOA cannot. Using multiple TOA or TDOA measurements, location can be determined.

The problem of converting TOA and TDOA measurements with anchor nodes to location has been well studied as two optimization problems, trilateration [59] and hyperbolic location [36, 77]. Trilateration and hyperbolic location are the bases for all modern time-based location systems, including mobile phone location techniques,

[2]For a discussion of existing clock oscillator types, see Appendix F.

LORAN, and GPS. Trilateration uses TOA measurements to solve for location, while hyperbolic location uses TDOA measurements. Hybrid approaches combining TOA and TDOA measurements to solve for location have been proposed [49, 70], but are not known to be implemented in production systems.

One important extension of the trilateration problem is the sensor network localization problem [12, 13, 35, 74, 90, 84]. In this problem, many TOA distance measurements are used to solve for multiple node locations simultaneously rather than the single node location provided by trilateration. There is no known analogous problem for hyperbolic location.

1.3 Contributions

The goal of this dissertation is to create an inexpensive, time-based location system with real-time accuracy of several meters that can provide coverage in areas where GPS cannot. This system is designed for nodes that can accurately timestamp broadcast network messages with their individual, free-running clocks. Potential node examples are laptops with wireless cards, wireless access points, mobile phones, and other embedded devices. Although experimental results are presented for IEEE 802.11 devices, the techniques apply to any other communication protocol and hardware in which precise and accurate timestamping functionality is available.

Figure 1.1 shows the overview of the time-based location process. The first step is collection of timestamp data from events, which in our case are the sending and receiving of network messages.[3] The second step is to combine the timestamps to produce either TOA or TDOA distance information. The final step is to solve the optimization problem defined by the distance information for location, which is passed

[3]The timestamping process by itself is a nontrivial task. Multipath effects may result in multiple, interfering phase-shifted copies of the signal, while for location purposes, the timestamp should be applied to the direct propagation path. Examples of techniques for multipath correction for location purposes are leading edge curve fitting and frequency, antenna, and spatial diversity [38, 60].

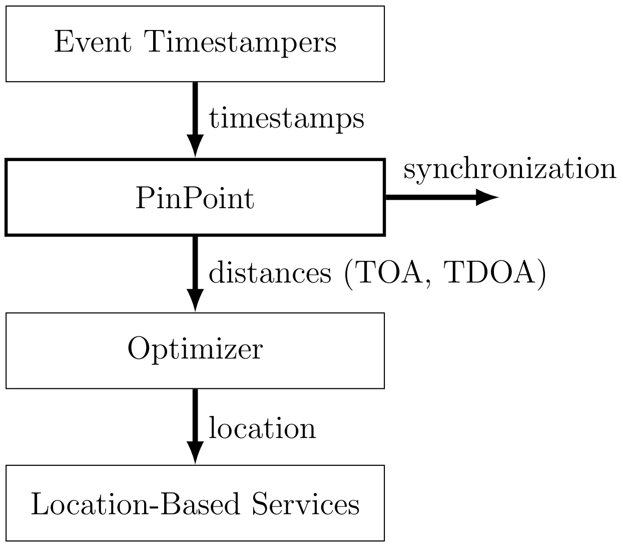


Figure 1.1: Time-based Location Process Overview

on for use by location-based services.

This dissertation contributes new knowledge primarily for the second step of converting timestamp information from inaccurate clocks to TOA and TDOA distance information. We assume that timestamp errors have consistent fixed bias and normally distributed error, which may not be true in multipath environments. Combining this distance information with existing optimization techniques and off-the-shelf 802.11 network cards and embedded devices, we demonstrate experimentally that PinPoint techniques succeed in providing location accuracy of a few meters.

We avoid many common assumptions of hardware capability in location and synchronization systems to reduce costs and make our techniques as general as possible. We do not assume that we can schedule packets with better than millisecond accuracy, in contrast with distance measurement designs based on known interpacket timings for the IEEE 802.11 protocol [42, 20, 21, 38]. We do not assume external synchronization such as IEEE 1588 is available [66, 54]. We also do not assume any capability to make hardware adjustments to clock frequency or offset, as is common among other synchronization designs [43, 55, 29]; the clocks are free-running.

Time-based location problems in 802.11 networks are difficult because of the coarse timing precision specified by the 802.11 standard and stability of clock oscillator drift. The 802.11 standard [2] specifies that timestamps must be available with a precision of 1 μs with clock drift accuracy of 100 microseconds per second (0.01%). With precision of 1 μs, a single measurement can only provide distance precision of $\sim$ 300 m, which is beyond the practical range of most 802.11 devices. To achieve the accuracy goal of the order of meters for a single measurement, timing precision and accuracy of a few nanoseconds is required. Using multiple measurements and statistical methods, this constraint can be relaxed, but better precision and accuracy allow better location accuracy with fewer measurements. With 100 ppm clock drift, a clock will lose less than 9 seconds per day. While this uncorrected clock drift is acceptable for many purposes, it is not for computing distance. Two uncorrected timing measurements taken 1 second apart can have timing error of 100 μs, which translates to 30 km of distance error for radio signals. Further, variation in clock drift is substantial enough to preclude modeling clocks with linear behavior.[4]

To model this nonlinear behavior in a practical setting, we present a consensus clock synchronization system based on dynamic measurements of clock offsets and clock drift. This approach allows virtual synchronization without physical adjustments. Each node uses a single mapping from its local time to a common consensus time scale, adjusting this mapping periodically in response to clock rate changes. Consensus clock synchronization has direct applications to TOA and TDOA distance measurements.

Consensus clock synchronization provides a foundation for comparing time differences from multiple nodes to compute TOA and TDOA. It also resolves the asymmetry in the original presentations of PinPoint TOA [85] and TDOA [57] distance measurements, where there were slight differences in distance results depending on which node in a pair was designated as node a or b.

[4]See Chapter 3.3.2 for empirical data.

Assuming that timestamping and multipath problems can be solved for complex environments using heuristics and other techniques, PinPoint provides the foundation for a complete wireless location system. PinPoint can locate both participating and non-participating nodes with accuracy better than 4 m using time of arrival (TOA) and time difference of arrival (TDOA) techniques.

This dissertation is structured as follows. First, we cover notation and definitions that are used throughout this work. Chapter 2 formally presents the distance measurements TOA and TDOA. From these measurements, we review existing the optimization techniques trilateration and hyperbolic location that compute location from TOA and TDOA measurements. Chapter 3 covers clock synchronization techniques for broadcast environments and their application to 802.11 devices for computing clock drift and offset. Chapter 4 introduces new techniques for consensus clock synchronization. Chapter 5 describes the PinPoint variants for converting timestamp measurements into TOA and TDOA distance measurements. We also present experimental evidence that PinPoint suffices for a location system with accuracy of a few meters. Chapter 6 covers our theoretical contributions for converting combined TOA and TDOA information into location. Chapter 7 gives examples of envisioned applications based on PinPoint variants.

1.4 Notation

1.4.1 Distance

$D(p, q)$ is the Euclidean distance between p and q.

$$D(p, q) = \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2 + (z_p - z_q)^2} \tag{1.2}$$

It is easier to describe distances in terms of light propagation time, so we adopt lowercase notation to reflect this:

$$d(p,q) = \frac{D(p,q)}{\text{speed of light}} \tag{1.3}$$

We assume a uniform speed of light of $2.998 * 10^8$ m/s and use time units for distance in this dissertation.

1.4.2 Time

We adopt the following convention for discussing time values. The value t is the common absolute time at all locations, ignoring any relativistic effects. We have no way of measuring t directly, since any time measurement is from a single clock's local time. For local times, we use the symbol τ. We denote a general local time at node a at absolute time t by $\tau_a(t)$. When a subscript is present, it will always designate the node whose local clock time is recorded.

We are interested in local times when messages are either sent or received. For local send times, we indicate a sender and message index. For local receive times, we add a receiver indicator. The presence of the receiver indicator therefore determines whether the local time is a send or receive time. Since we are concerned only with wireless broadcast messages, send times do not have receivers associated with them. The message index appears only when necessary for clarity.

$\tau^s(i)$	send time of ith message from sender s
$\tau^s_r(i)$	receive time of ith message from s at receiver r

The offset between two clocks can be measured in two distinct ways. Both are important for computing distance and time.

Event	Local Time	Absolute Time
Message sent by a	τ^a	t^a
Received at b	τ^a_b	t^a_b
Received at c	τ^a_c	t^a_c

Table 1.1: Local Time Examples

$\theta_{a \mapsto b}$ The difference between two local clock counters is measured at a specific local time. Assuming constant clock drift rates, this difference is a linear function. This is the more natural way of thinking about clock offset.

$\theta_{a \to b}$ The difference between two clocks is measured on a virtual consensus clock timescale. Assuming constant clock drift rates, this difference is constant. This is described in detail in Chapter 4.3.

Each clock ω is assumed to run at a rate $\beta_\omega = 1 + \delta_\omega$, where $|\delta_\omega| < 10^{-4}$. Either the β or δ notation may be used depending on the context.

The calculation of relative clock drifts has two distinct estimates.

$\widehat{\left[\frac{\beta_b}{\beta_a}\right]}$ The point estimate of slope is the primitive.

$\frac{\beta_b}{\beta_a}$ The cumulative estimate of slope may use multiple point estimates.

1.4.3 Wireless Nodes

We classify wireless nodes based upon whether they are participating in our location protocol and whether they have known location. All nodes are assumed to transmit messages with a common wireless network communication protocol.

Target nodes include any node that we wish to locate, which naturally have unknown location.

Anchor nodes both participate in the location protocol and have known location.

Mobile nodes participate in the location protocol but do not have known location.

	Location	
	Known	Unknown (Target)
Participant	Anchor	Mobile
Nonparticipant	Landmark	Stranger / Rogue

Table 1.2: Categorization of Wireless Nodes

Landmark nodes do not participate in the location protocol but have known location.

Stranger nodes neither participate in the location protocol nor have known location.

Rogue nodes are stranger nodes that are malicious or otherwise pose a threat.

This classification is summarized in Table 1.2.

We assume that location protocol participants can timestamp messages. Usually this ability includes both sent and received messages, but we will also examine the case where only receive timestamps are available. All timestamps recorded by participants are available for location computation.

Nodes with known location are not necessarily fixed. Their location information is simply known through outside sources and can be updated over time if the node is moving.

Chapter 2

Distance Measurement Primitives and the Location Problem

In this chapter, we present known techniques for converting distance information into location information. We refer to the general process of converting distance measurements to a location as the location problem.

All known time-based distance measurements produce either time of arrival (TOA) or time difference of arrival (TDOA) geometric results, which we refer to collectively as distance measurement primitives. These are the simplest units of geometric information available known to be computable from time information, and all known time-based location systems use one of these distance measurement primitives.

In this chapter, we assume that all clocks measure absolute time to simplify the presentation of these distance measurement primitives. In Chapter 3, we consider a more realistic clock model with relative clock drifts, offsets, and timestamp delay biases. In Chapter 5, we show how to compute TOA and TDOA under this clock model.

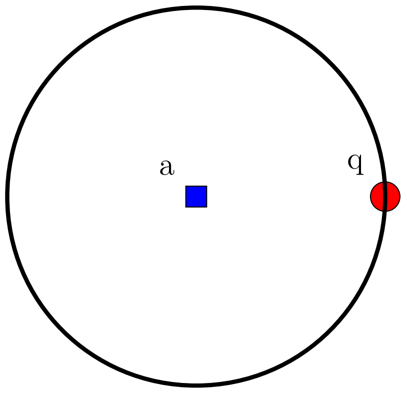


Figure 2.1: Circle satisfying TOA for a, q

2.1 Time of Arrival (TOA)

TOA measures the point-to-point distance between two participant nodes a and q. Nodes a and q measure the time messages are sent t^a from a and the time received t_q^a at q. The difference between these times is the propagation time.

$$\text{TOA}(a, q) = d(a, q) = t_q^a - t^a \tag{2.1}$$

If a is an anchor node, and q is a mobile node, the locus of possible positions of q satisfying the distance equation is a sphere centered at a.

2.2 Time Difference of Arrival (TDOA)

TDOA measures the difference of the distances between a single node q and two participant nodes a and b.

$$\text{TDOA}(a, b, q) = d(b, q) - d(a, q) \tag{2.2}$$

TDOA is a signed quantity; comparison of reception times shows which receiver (focus) is closer to the node q. The locus of possible positions of q satisfying the distance equation is a hyperboloid with foci at a and b. As a result of the triangle

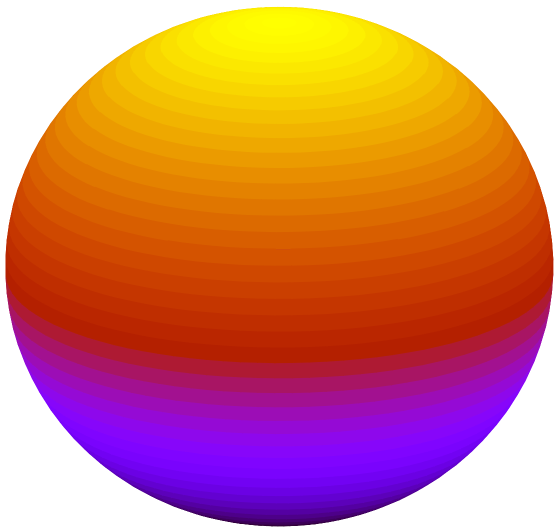


Figure 2.2: Sphere

inequality, the TDOA value is bounded by the distance between nodes.

$$d(a,b) \geq |d(b,q) - d(a,q)| \tag{2.3}$$

The two basic methods to compute TDOA are distinguished by where measurements are taken and computation occurs. The first method is network-based, where all timestamp measurements are made at participant nodes and a server computes location of the target node q. For this method, the target node q sends normal messages but need not participate explicitly in the location protocol. The second method is mobile-based; a mobile node computes its own location without measurements or computation from other nodes.

For the network-based method, a TDOA measurement requires two anchor nodes a and b acting as receivers.

1. q transmits a packet at time t^q.

2. a receives the packet at time $t_a^q = t^q + d(q,a)$

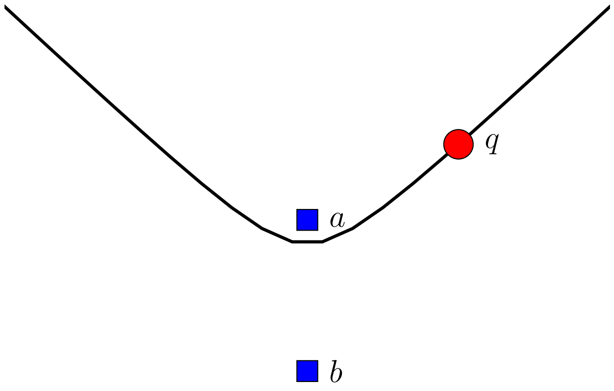


Figure 2.3: Hyperbola satisfying TDOA for receivers a, b

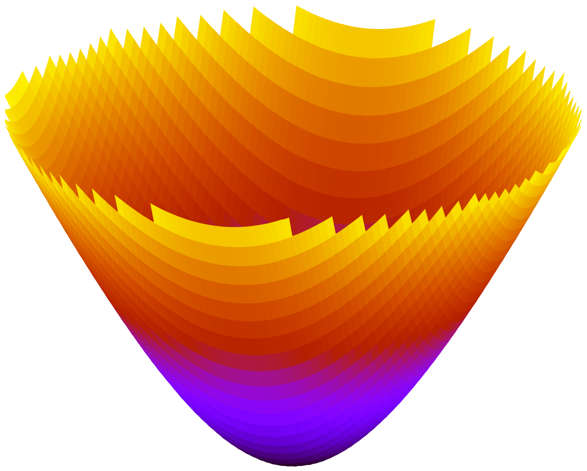


Figure 2.4: Hyperboloid Branch

3. b receives the packet at time $t_b^q = t^q + d(q, b)$

Since the same packet is received by both a and b,

$$\Delta t = t_b^q - t_a^q = d(b, q) - d(a, q) \tag{2.4}$$

Note that the value of t^q is not communicated to any anchor node. The value of t^q is not important — it is only important that packets sent by q be successfully received by both receivers. For this reason, this TDOA measurement may be made for target nodes that are either mobile or stranger nodes. If the target node q is transmitting messages

For the mobile-based method, a mobile node q computes its own location using anchor node transmissions. This TDOA measurement requires two anchor nodes acting as transmitters a and b with synchronized clocks.

1. a transmits a message at time t^a.

2. q receives this message at time $t_q^a = t^a + d(a, q)$.

3. b transmits a message at time $t^b = t^a + k$, where k is a known time delay. The value k may be conveyed in b's message or through out-of-band communication to q.

4. q receives this message at time $t_q^b = t^b + d(b, q)$.

Since the message from b is sent precisely k after the message from a,

$$\Delta t = t_q^b - t_q^a - k = d(b, q) - d(a, q) \tag{2.5}$$

The value t^a is not available to the mobile node q. If it were, the mobile node could compute TOA(a, q) and TOA(b, q). Communicating t^a to q is nontrivial when q is not synchronized to a and b.

The choice of TDOA method depends on resource constraints in the system. The network-based method has the advantage it works for any active target nodes, but the cost of locating many nodes simultaneously may be significant. The mobile-based method has the advantage of introducing only constant overhead at the anchor nodes; an unlimited number of mobile nodes may compute TDOA using the messages from the anchors. To use the mobile-based method, the mobile node must support the location protocol, which may require software changes.

2.3 Location Problem

In the generic location problem, a set of participant nodes locates a target node q. Using measured distances from either TOA or TDOA, participant nodes estimate a location solution q'. The error for the estimated solution is simply $D(q, q')$. The goal of the location problem is to minimize the error $D(q, q')$ given measured distance data.

We are primarily interested in two distinct variants of the location problem with anchor nodes, trilateration and hyperbolic location. Each uses only one information source input. Trilateration solves the location problem for TOA distance measurements, which are absolute distances. Hyperbolic location solves the location problem for TDOA distance measurements, which are the difference of distances.

To use multiple information sources, a hybrid approach is necessary. Our contribution to hybrid methodology in Chapter 6 describes conditions in which TOA information is strictly better than using both TOA and TDOA information. This result leaves open the question of how to best compute location under other conditions given both TOA and TDOA information.

In trilateration and hyperbolic location, each target node is located independently of other target nodes. Distances involving multiple target nodes are not used. The

$$\begin{tikzcd}
\text{TOA} \arrow[d] \arrow[dr] & & \text{TDOA} \arrow[dl] \arrow[d] \\
\text{Trilateration} & \text{Hybrid} & \text{Hyperbolic Location}
\end{tikzcd}$$

Figure 2.5: Information Sources and Associated Location Problems

problem considering the distance information involving multiple target nodes is the sensor network localization problem, a subject of significant ongoing research, described in Appendix C.

2.3.1 Trilateration

In the trilateration problem, a set of anchor nodes A locates a mobile node q. Using TOA, we estimate pairwise distances $\text{TOA}(a, q) = \hat{d}(a, q)$, where $a \in A$. $\text{TOA}(a, q)$ defines a sphere centered at an a. We find the location of q that best satisfies these distance measurements, which may either be error-free or be noisy and contain errors. In the error-free case, the trilateration problem solution is the intersection of spheres, which can be solved directly using algebra. In the noisy case, the trilateration problem is a nonlinear optimization problem.

The nonlinear optimization objective function to minimize is the squared error.

$$\text{SE} = \sum_{a \in A} \left(\hat{d}(a, q) - d(a, q) \right)^2 \tag{2.6}$$

To minimize the squared error, we use iterative gradient descent. This approach is the same as that described by Caffery and Stuber in [15] for hyperbolic location. This is not meant to be the fastest or most accurate solution to the optimization problem, but is used for ease of implementation to illustrate the efficacy of TOA methods for

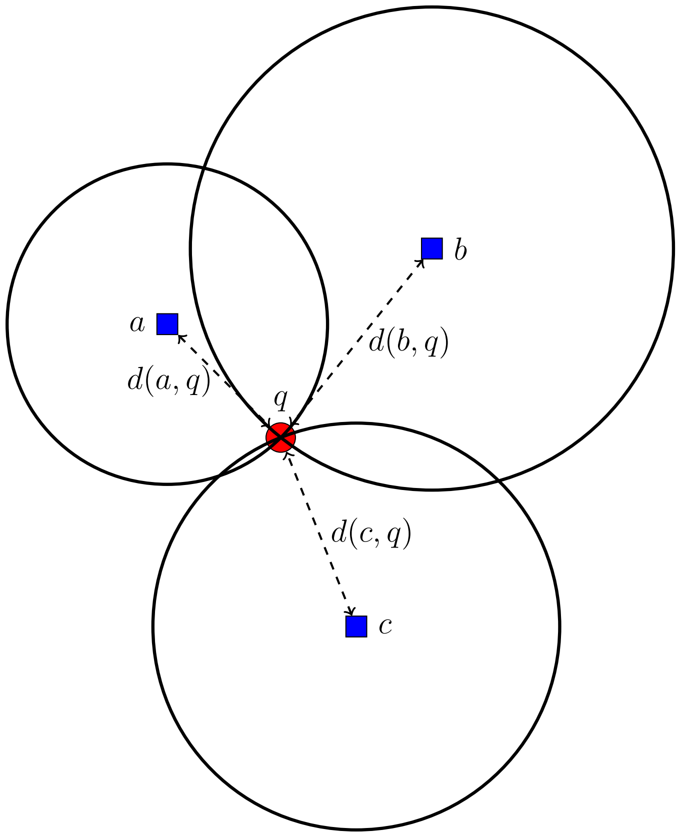


Figure 2.6: Trilateration

the location problem.

Gradient descent is an iterative optimization technique. It requires an initial guess q_0 for the location and is not guaranteed to converge to the correct solution. At each iteration, the normalized gradient $\frac{\nabla \mathrm{SE}(q)}{|\nabla \mathrm{SE}(q)|}$ is the direction of greatest increase, so $-\frac{\nabla \mathrm{SE}(q)}{|\nabla \mathrm{SE}(q)|}$ is the direction of greatest decrease for the squared error $\mathrm{SE}(q)$. We use a line search to compute a distance λ to move in this direction. If the gradient changes direction rapidly over short distances, λ will be small. The point q_{n+1}

$$q_{n+1} = q_n - \lambda \frac{\nabla \mathrm{SE}(q)}{|\nabla \mathrm{SE}(q)|} \tag{2.7}$$

is the next iterative solution.

The gradient for the trilateration squared error function is given by Equation 2.8.

$$\nabla \mathrm{SE}(q) = \begin{bmatrix} \sum_{a \in A} 2 \left(\hat{d}(a,q) - d(a,q) \right) \frac{x_p - x_q}{d(p,q)} \\ \sum_{a \in A} 2 \left(\hat{d}(a,q) - d(a,q) \right) \frac{y_p - y_q}{d(p,q)} \\ \sum_{a \in A} 2 \left(\hat{d}(a,q) - d(a,q) \right) \frac{z_p - z_q}{d(p,q)} \end{bmatrix} \tag{2.8}$$

The error for the iteration solutions determines when to stop the iterative optimization process. An absolute error threshold is not useful because there is no guarantee that any solution can satisfy it. A relative error stopping condition is therefore used. When the error between successive iterations is less than an iterative tolerance threshold, we terminate the optimization process. The last iteration solution is the best estimate for trilateration.

2.3.2 Hyperbolic Location

In the hyperbolic location problem, a set of anchor nodes A locates a target node q. Using TDOA, we estimate distances $\mathrm{TDOA}(a,b,q) = \hat{d}(b,q) - \hat{d}(a,q)$, where $a, b \in A$. The locations of a and b are the foci of a hyperboloid solution set. We find the

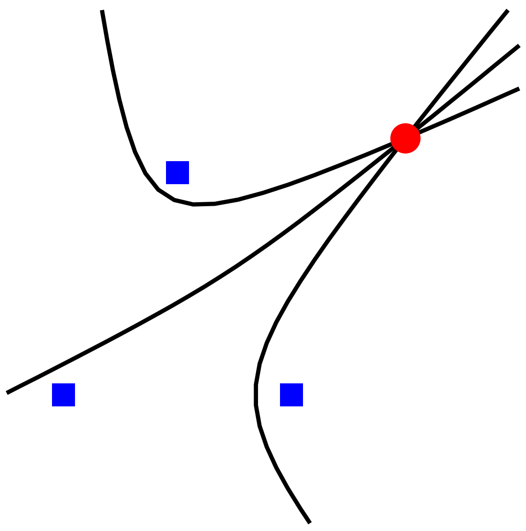


Figure 2.7: Hyperbolic Location

location of q that best satisfies all TDOA measurements. As with trilateration, the error-free case can be solved directly using basic algebra.

With measurement errors the hyperbolic location problem is a nonlinear optimization problem. A simple approach minimizes the squared error for all pairs of nodes a, b providing TDOA information.

$$\mathrm{SE} = \sum_{a,b} \Big([\hat{d}(b,q) - \hat{d}(a,q)] - [d(b,q) - d(a,q)] \Big)^2 \tag{2.9}$$

To minimize this function, we use the gradient descent method for hyperbolic location described by Caffery and Stuber in [15], which is the same approach described for trilateration above. We assume there may be significant error in the TDOA

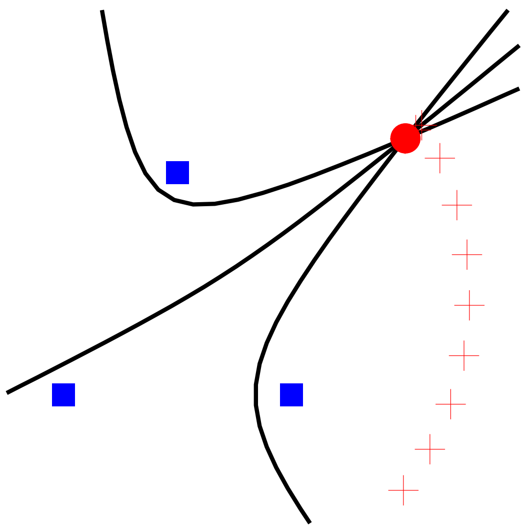


Figure 2.8: Optimization for multiple hyperbolas

measurements, precluding the use of exact techniques such as that by Fang [36].

$$\nabla \mathrm{SE} = \begin{bmatrix} \sum_{a,b\in A} 2\left[\left(\hat{d}(b,q)-\hat{d}(a,q)\right)-(d(b,q)-d(a,q))\right]\left[\frac{x_b-x_q}{d(b,q)}-\frac{x_a-x_q}{d(a,q)}\right] \\ \sum_{a,b\in A} 2\left[\left(\hat{d}(b,q)-\hat{d}(a,q)\right)-(d(b,q)-d(a,q))\right]\left[\frac{y_b-y_q}{d(b,q)}-\frac{y_a-y_q}{d(a,q)}\right] \\ \sum_{a,b\in A} 2\left[\left(\hat{d}(b,q)-\hat{d}(a,q)\right)-(d(b,q)-d(a,q))\right]\left[\frac{z_b-z_q}{d(b,q)}-\frac{z_a-z_q}{d(a,q)}\right] \end{bmatrix} \quad (2.10)$$

A sample optimization in two dimensions is shown in Figure 2.8. The anchor nodes are shown with squares, the optimization steps are shown with crosses, and the location of the q is shown with a circle.

2.3.3 Alternate Names

In the literature, the trilateration and hyperbolic location problems have many alternate, sometimes overlapping names. Triangulation, which is the location based upon angles, is frequently misused for both trilateration and hyperbolic location. Multilateration is also used to describe both, though for trilateration it usually applies to cases with four or more participants locating a mobile node. The hyperbolic location problem is also called hyperbolic multilateration, hyperbolic trilateration, hyperbolic triangulation, and hyperbolic positioning.

Regardless of the terminology, the important distinguishing feature is the information source. Location is based either on measurements of distances (TOA) or differences of distances (TDOA). The corresponding geometries are spheres for TOA and hyperboloids for TDOA.

2.3.4 Anchor Node Geometry

The relation between distance measurement accuracy and location accuracy is governed by anchor node geometry and is quantified using dilution of precision (DOP). DOP is a unitless quantity. Low values indicate favorable geometry, and high values indicate poor geometry. Dilution of precision can be subdivided into horizontal and vertical dilution of precision. Whenever possible, we prefer the anchor nodes to be arranged with favorable geometry to maximize location accuracy with a given level of distance measurement accuracy. In systems such as GPS this is not always possible because satellites are constantly moving and their orbits are predetermined to maximize planet coverage.

For hyperbolic location, Yang and Scheuing [82, 83] have shown the theoretical optimum anchor arrangement is a uniform angular array using the Cramer-Rao lower bound and an assumption of independent Gaussian noise. In 2D, the solution is the circle, with anchor nodes equally spaced around the perimeter. In 3D, the solutions are the five Platonic solids: tetrahedron, octahedron, cube, icosahedron, and dodecahedron. Anchor nodes are placed at the vertices of the solids.

The results for trilateration in the context of GPS are similar. The best configuration for a receiver on the surface of the earth with four satellites has three satellites equally spaced on the horizon at the minimum elevation angle and one satellite directly overhead [39].

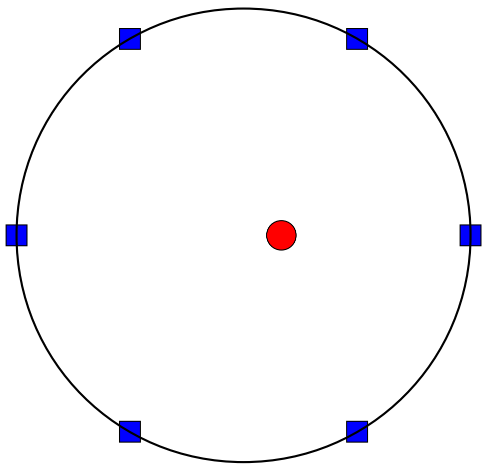
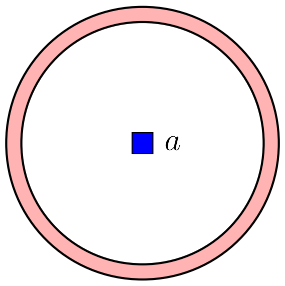


Figure 2.9: Ideal Geometry: Uniform Angular Array

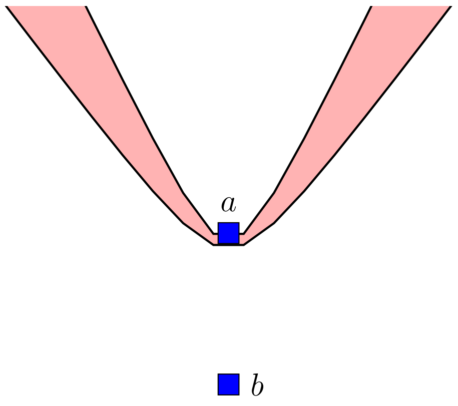
2.4 Comparison of TOA and TDOA

TOA measurements can be used to derive TDOA measurements, but there is no known way to convert from TDOA measurements to TOA. By subtracting two TOA measurements, we can obtain the same information as from a TDOA measurement $\text{TOA}(b, q) - \text{TOA}(a, q) = \text{TDOA}(a, b, q)$. In order to take TOA measurements, more control over nodes is required than for TDOA measurements; all node must timestamp messages, or nodes must respond to messages in a specific way, or messages from nodes must be sent synchronously at known times. In short, there are conditions under which only TDOA can be computed, but whenever TOA can be computed, so can TDOA.

A significant difference between TOA and TDOA for the location problem is the geometry of their solution sets. Each TOA sphere is contained within a bounded space, while the TDOA hyperboloid is unbounded and extends out into space. These are illustrated in Figure 2.10. For a given error tolerance, the intersection of spheres will be bounded, but this need not be the case for the intersection of hyperboloids.



(a) TOA: $k - \epsilon \leq d(a,q) \leq k + \epsilon$



(b) TDOA: $k - \epsilon \leq d(b,q) - d(a,q) \leq k + \epsilon$

Figure 2.10: Geometric solution sets with finite error tolerance

2.5 Summary and Other Location Problems

In this chapter, we presented the geometric solution sets for TOA and TDOA measurements and synchronized techniques to generate TOA and TDOA information from timestamped messages. We also presented gradient descent optimization techniques to solve the trilateration problem for TOA measurements and hyperbolic location for TDOA measurements. These techniques are sufficient to provide locations based on TOA or TDOA data, which are the bases of all known time-based location systems. Other optimization techniques are available to minimize the trilateration and hyperbolic location error functions.

Location problems that take into account more information than only TOA or only TDOA for one target may yield more accurate solutions. Hybrid approaches using both TOA and TDOA simultaneously are addressed in Chapter 6. The sensor network localization problem solves for multiple target locations simultaneously while using intertarget distances. Appendix C gives an overview of sensor network localization research. Further research in these areas may lead to more accurate location solutions than trilateration or hyperbolic location.

Chapter 3

Time Measurement

To estimate location in real time to within meters, we need precise and accurate timestamps as well as synchronization techniques to use timestamps from multiple nodes. We assume that all timestamps come from either the sending or receiving of a message. Each timestamp is an integer reading in units of clock ticks from a node's local clock. We assume that all nodes' clocks have the same nominal frequency, usually either 40 MHz or 44 MHz, though the frequencies will deviate from the nominal value.

As a practical matter, timestamp precision is a major issue for 802.11 networks. The 802.11 standard [2] specifies that timestamps must be available to one microsecond precision. Most commercial 802.11 hardware does not readily support timestamping to greater precision. With one microsecond timestamping precision, a single measurement can achieve no better than 300m precision. More precise timestamping enables more precise location estimates.

The most precise timestamping platform we have studied to date has been the Roving Networks RN-134 unit based on a G2 Microsystems chip. The RN-134 can timestamp both incoming and outgoing packets with a 44 MHz clock. Each tick of a 44 MHz clock is approximately 22.7 ns, which translates to 6.81m for speed of light communications.

In addition to precision, timestamping accuracy is necessary for location accuracy. There are two components of accuracy: bias and noise. The bias component is systematic, while the noise is random. We can correct bias for clock drift (Section 3.4.2) and distance measurements (Chapter 5). The effects of relatively small random noise can be overcome by using repeated measurements and statistics.

To reduce timestamping noise, hardware support for network MAC clock timestamping is required. The noise ($\sim \mu$s) for CPU-based timestamping in modern operating systems resulting from interrupt latency is too big for real-time location. Too many samples must be collected to find location in real time with reasonable accuracy. By timestamping with the MAC clock, interrupt latency is avoided.

In an idealized world, every clock measures time indistinguishably; we can directly compare times from different clocks and every clock runs at exactly the same frequency. In practice, however, no two clocks will have exactly the same frequency, and clock frequencies may change depending on oscillator stability and factors such as temperature. Significant frequency changes can occur over time periods as short as seconds. For a discussion of clock oscillators, see Appendix F. To handle these complexities of multiple independent clocks, we need synchronization.

3.1 Synchronization

In Cristian's definition for clock synchronization [25], two clocks a and b are synchronized within a fixed synchronization tolerance ϵ if

$$|\tau_a(t) - \tau_b(t)| < \epsilon \tag{3.1}$$

Two clocks can be synchronized by adjusting their times to match using methods such as Cristian's algorithm or the Simple Network Time Protocol (SNTP) [61]. This involves an offset adjustment to reset the clock tick counter. To maintain synchronization over time, this offset adjustment must be repeated periodically. To increase the period between adjustments, synchronization systems may also adjust clock frequency.

Synchronization can be performed relative to a master clock or to an average of clocks. NTP [62] and IEEE 1588 [1, 33] adjust offsets and clock frequency to a master clock. The proposed pulse-coupled oscillators [43, 55] and joint distributed synchronization [29] adjust to an average.

An alternative to strict synchronization with clock adjustments is to define virtual clock functions f_a, f_b, mapping physical clock times to a synchronized virtual clock:

$$|f_a(\tau_a(t)) - f_b(\tau_b(t))| < \epsilon \tag{3.2}$$

Reference broadcast synchronization, presented by Elson, Girod, and Estrin [34], finds a linear function f_a and uses the identity function for f_b.

For location purposes we are interested in synchronization in local networks, where the synchronization tolerance ϵ can be made on the order tens of nanoseconds. Multi-hop synchronization techniques such as NTP, which is designed to synchronize across the Internet, are not suited to our problem, where we measure distances.

Note that to measure distances, it is sufficient to measure the time elapsed for message propagation. It is not necessary to have know absolute time; signals will travel the same distance in one second at noon as they will at midnight.

3.1.1 Cristian's Algorithm

Cristian's Algorithm [24, 25] is a simple one round protocol for a node a to set its clock to that of node b. The messages and times are shown in Figure 3.1. a sends a request to b and records the time τ^a. Upon receipt of the message, b immediately sends a response with its current time $\tau^a_b = \tau^b$ to a. Node a receives this at time τ^b_a,

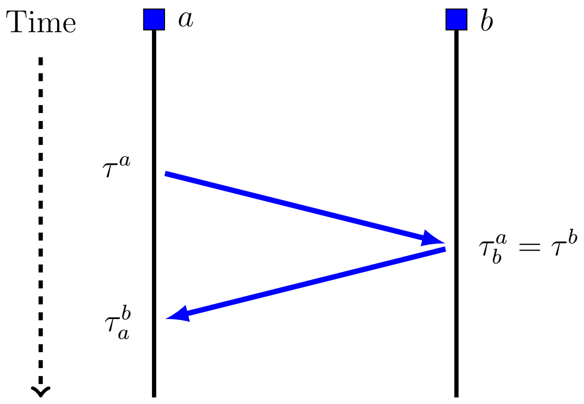


Figure 3.1: Cristian's Algorithm message exchange

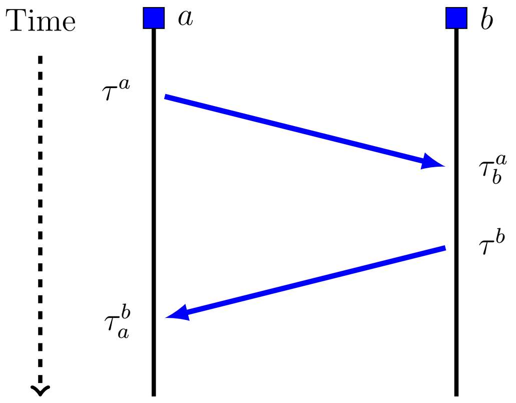


Figure 3.2: SNTP message exchange

and sets its clock to time $\tau^b + \frac{1}{2}(\tau^b_a - \tau^a)$.

If the response from b is delayed by processing or network congestion, the round trip time will be overestimated. Multiple rounds may be used, with the minimum round trip delay time used to synchronize a's clock. To eliminate the effect of these delays, other protocols such as SNTP use a second timestamp at node b.

3.1.2 Simple Network Time Protocol (SNTP)

SNTP [61] uses one round of messages to determine both round trip time δ and a clock offset $\theta_{a \mapsto b}$. SNTP is a stateless protocol that does not consider clock drift. Each message has a send and a receive timestamp, for a total of four timestamps. We need only the clock offset $\theta_{a \mapsto b}$ to correct $a's$ clock.

The equations for the round trip time δ and the clock offset $\theta_{a\mapsto b}$ are:

$$\delta = [(\tau_a^b - \tau^a) - (\tau^b - \tau_b^a)] \tag{3.3}$$

$$\theta_{a\mapsto b} = \frac{1}{2}(\tau^b - \tau_a^b + \tau_b^a - \tau^a) \tag{3.4}$$

To adjust its clock, a sets $\tau_a := \tau_a + \theta_{a\mapsto b}$.

In contrast to Cristian's algorithm, in SNTP the node b need not respond immediately to the request message. The node b must, however, timestamp both when receiving and sending messages.

The accuracy of SNTP depends upon network conditions and the frequency that messages are sent. Due to network congestion considerations, the maximum frequency guideline is one round of messages every 15 seconds. At this frequency, the theoretical synchronization accuracy limit for SNTP is on the order of milliseconds because for clock drift of 100 ppm, over 15 seconds two clocks may drift by 1500 μs, or 1.5 ms. SNTP can achieve accuracy close to this limit over local networks.

Even with greater message frequency, SNTP accuracy is limited by the clock drift. For messages sent every 0.02 seconds (50 message rounds per second) and 100 ppm clock drift, the accuracy is no better than 2 microseconds, which is insufficient for our location needs.

3.1.3 IEEE 1588

IEEE 1588 [1, 33] is a synchronization protocol designed to give submicrosecond accuracy for localized networks. Slave clocks synchronize to their master clock, which may in turn be synchronized to higher master clocks to form a hierarchy. The hierarchy is topped by the grandmaster clock. The grandmaster can be determined from member nodes by a master clock selection algorithm.

The IEEE 1588 message exchange provides the same information as the SNTP

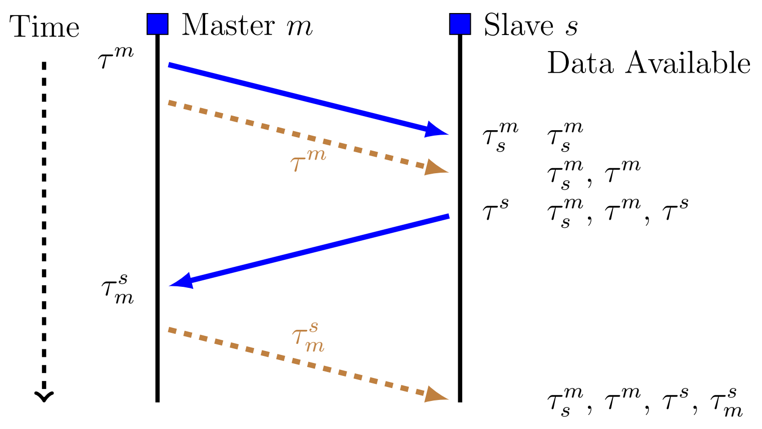


Figure 3.3: IEEE 1588 message exchange

exchange, with the difference that outgoing message send times are included in followup messages. As in SNTP, we can compute the clock offset and round trip time from the message exchange.

3.1.4 Reference Broadcast Synchronization

In a broadcast network environment, an alternative to computing offset adjustments is to synchronize two participant nodes by using a third node's transmissions. The transmission times for this node's messages need not be timestamped, in contrast to Cristian's method, SNTP, and IEEE 1588. The use of reference broadcasts to synchronize clocks of receivers has been shown effective for determining clock offset and drift in [34] due to the removal of sender timestamp variability. In the original presentation of this technique, propagation time is ignored. Propagation times are assumed to be on the order of μs, while the synchronization tolerance is in ms.

The node l transmits a packet at time $t^l(1)$. The packet is received at participants a and b. Assuming propagation delay is negligible, a receives the packet and timestamps it locally at time $\tau_a^l(1)$. b receives the packet at the same time as a, but timestamps it locally on its clock at $\tau_b^l(1)$. After collecting many timestamps, linear regression is used to fit slope m and intercept k to the sets $\{\tau_a^l(i)\}$ and $\{\tau_b^l(i)\}$ to determine the

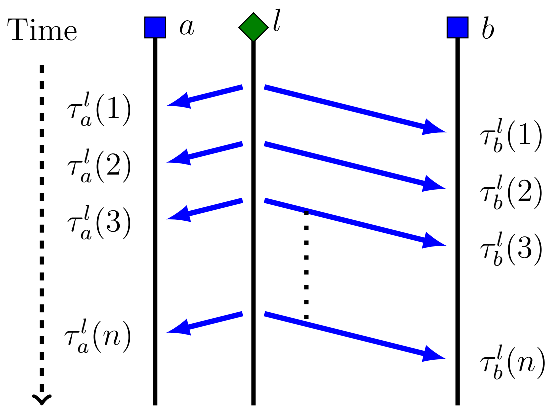


Figure 3.4: Reference broadcast synchronization

linear function

$$\tau_b = m\tau_a + k \tag{3.5}$$

Using this equation, times can easily be converted from a's local time to b's local time.

3.2 Basic Clock Model

To understand the performance of synchronization methods, we first need to establish a model for clock behavior for network timestamping. We model each node as having an independent, free-running clock that records time in its own frame of reference. The goal of our clock model is to accurately convert times for multiple clocks into a common frame of reference. We assume all clocks run with the same nominal frequency, for example 40 MHz or 44 MHz. The nominal clock tick lengths for these frequencies are 25 ns and 22.7 ns respectively.

$$\tau = \lfloor \beta(t + \alpha) \rfloor \tag{3.6}$$

Each clock starts independently, indicated by separate α values. Each clock runs at a slightly different rate, reflected by β. We model the discretization error combined with other measurement errors in e.

$$\tau = \beta(t + \alpha) + e \tag{3.7}$$

The quantity τ is an integer, with units of clock ticks whose duration is determined by the clock speed. According to Allan [6, 7], error terms for e follow power law spectra $S(f) \sim f^k$ where f is the Fourier frequency and k is an integer.

For message send and receive times, the application of timestamps to messages may be biased. Bias may be delays between receiving a message and applying its timestamp, or in the difference between when a message is expected to be sent and it is actually sent.

$$\tau^a = \beta_a(t^a + \alpha_a) + s_a + e^a \qquad \text{send} \tag{3.8}$$

$$\tau^a_b = \beta_b(t^a + \alpha_b + d(a,b)) + r_b + e^a_b \qquad \text{receive} \tag{3.9}$$

The term s_a is the send bias for the node a. The term r_b is the receive bias for the node b. If timestamping were unbiased, then $s_a = r_b = 0$. These terms are the differences between when a message is actually sent or received and when it is timestamped. These differences are in local clock units and not absolute time, so they are not multiplied by β. They can be negative, depending on the how timestamps are applied.

For location purposes, it is important to understand how our clock model behaves for existing network hardware.

3.3 Empirical Clock Behavior

Since a perfect clock does not exist, it is not possible to compute α or β for any one clock. We can only compute relative offsets or relative ratios for two clocks. If each node behaves linearly, the relationship between two nodes' timestamps should be a linear function. We expect each clock to behave linearly, but observe empirically that 802.11 clocks have significant nonlinear behavior.

To see this nonlinear behavior, we apply reference broadcast synchronization (RBS) [34], where a third node l broadcasts a message that is received by both a and b. RBS eliminates sender variability, but requires the third node l. In the original RBS paper, propagation distances were ignored. We examine how RBS works with respect to our clock model, which includes propagation distances.

For event i occurring at time $t^l(i)$, participant nodes a and b measure the time separately at the following times.

$$\tau_a^l(i) = \beta_a(t^l(i) + \alpha_a + d(a,l)) + r_a + e_a^l \tag{3.10}$$

$$\tau_b^l(i) = \beta_b(t^l(i) + \alpha_b + d(b,l)) + r_b + e_b^l \tag{3.11}$$

Refer to Figure 3.4. We can solve for $\tau_b^l(i)$ in terms of $\tau_a^l(i)$.

$$\tau_b^l(i) = \frac{\beta_b}{\beta_a}\left(\tau_a^l(i) - r_a - e_a^l\right) + \beta_b[(\alpha_b - \alpha_a) + (d(b,l) - d(a,l))] + r_b + e_b^l \tag{3.12}$$

The relationship between $\tau_b^l(i)$ and $\tau_a^l(i)$ is linear with slope $\frac{\beta_b}{\beta_a}$. With this in mind, we examine empirical data for τ_b^l and τ_a^l.

3.3.1 Experiment Setup

The hardware for this experiment consisted of three laptop computers, each with an Atheros-based Netgear wireless PC card using the madwifi-ng driver for GNU/Linux. Node l operated in AP mode, but functioned as a nonparticipant because none of its timestamps were used. The other two nodes, a and b, operated in both AP and monitor mode. All nodes were placed together to make $d(a,l) = d(b,l) = 0$. Node l

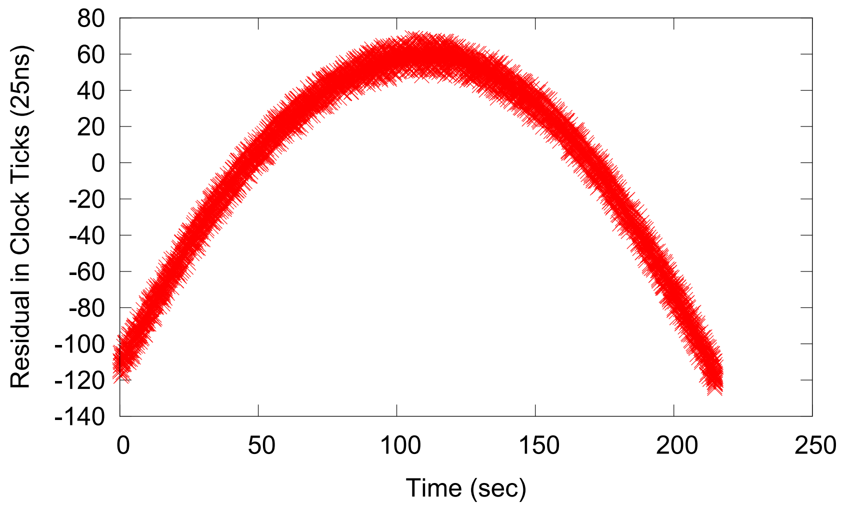


Figure 3.5: Clock deviation from RBS line

sent 200 beacons per second. Nodes a and b were run with reference clock speeds of 40 MHz and sent 40 beacons per second. For a 40 MHz clock, each clock tick is 25 ns. Over the experiment duration of $\sim$ 220 seconds, a and b recorded approximately 42000 beacons from c and $\sim$ 8400 beacons from each other.

Operating nodes a and b in AP mode is not necessary for RBS. The additional information supplied by operating these nodes as APs will be used to compare clock offsets to RBS in Section 3.4.1.

3.3.2 Nonlinear Results

RBS linear regression determined the clock relation to be:

$$\tau_b^l = (1 - 1.99 * 10^{-6})\tau_a^l + 889936589.95 \tag{3.13}$$

Figure 3.5 illustrates the error residuals, $\tau_b^l(i) - (m\tau_a^l(i) + k)$, for the message timestamps from this line. The deviation from the linear model ranges systematically between -127 and +70 clock ticks. This deviation is significant for distance measurements. If a timestamp measurement for a 40 MHz clock is off by 100 clock ticks, the computed distance can be off by 750m.

This nonlinear behavior is typical for the inexpensive clocks found on almost all network devices. Over the course of the experiment, the deviation is less than 0.02 ppm, well within the 802.11 standard tolerance of 100 ppm. Without replacing the clock oscillators with more expensive, stable technology (see Appendix F), we cannot expect to eliminate the nonlinear behavior.

In Figure 3.5, the first derivative of the residual function is linear, resulting in quadratic residuals. This corresponds to constant change in relative clock drift. In general, the residuals may follow higher order polynomials, particularly over longer time periods. The drift rate may fluctuate, but the drift rate will be bounded by the 802.11 hardware value of 100 ppm.

We believe that the main factor contributing to nonlinear behavior is temperature. In standard 802.11 operating environments, variable temperature is expected from changes in ambient temperature and electronics waste heat, thus the basic clock model is insufficient to completely characterize clock behavior.

3.4 Piecewise Linear Clock Model

Instead of fitting an explicit linear clock model to our timestamp measurements, we track the offset between two clocks implicitly to define the offset function $\theta_{a\mapsto b}(\tau)$. $\theta_{a\mapsto b}(\tau)$ enables translation between the times of two different clocks. Equation 3.14 shows how to translate a time τ_a from a's timescale to a value on b's timescale.

$$\tau_b = \tau_a + \theta_{a\mapsto b}(\tau_a) \tag{3.14}$$

Each point in the offset function is the result of one round of messages. Messages are sent with period P. Between the messages, we assume the offset function is locally linear and use an estimation of the relative clock drift to compute intermediate values.

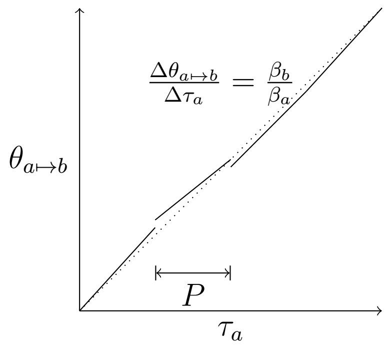


Figure 3.6: Piecewise Linear Clock Offset

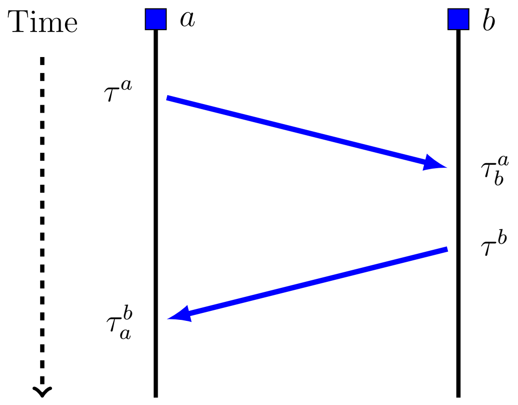


Figure 3.7: Message round for computing offset $\theta_{a\mapsto b}$

This results in a piecewise linear function. A notional example of such a piecewise linear function is shown in Figure 3.6.

3.4.1 Estimation of Offset

We apply SNTP [61] techniques for measuring clock offsets with network messages. SNTP collects send and receive timestamps from one round of messages to estimate the clock offset $\theta_{a\mapsto b}$ and roundtrip delay time δ for two network nodes. Initially, we are interested only in the offset $\theta_{a\mapsto b}$. The messages and timestamps necessary are shown in Figure 3.7. In this diagram, a message sent at time t^a arrives at time $t^a_b = t^a + \frac{1}{2}\delta$, and a message sent at time t^b arrives at time $t^b_a = t^b + \frac{1}{2}\delta$. The SNTP

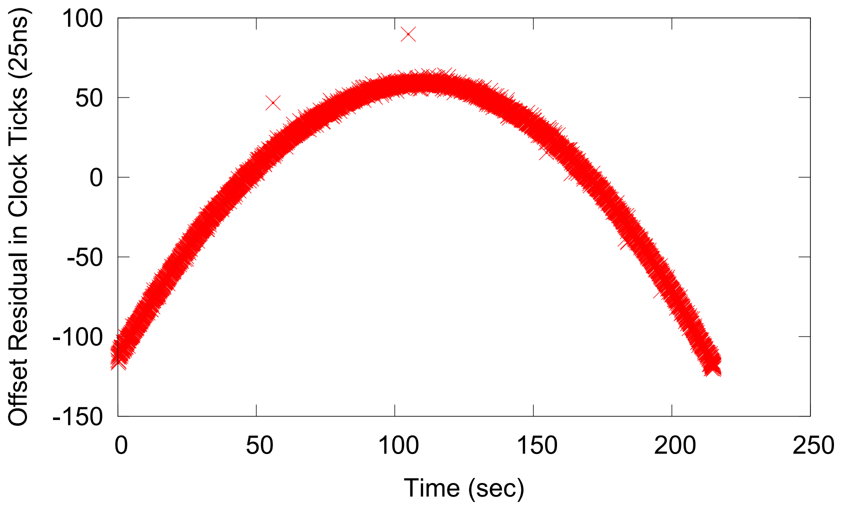


Figure 3.8: $\theta_{a\mapsto b}$ residuals

equation, which ignores clock drift and timestamping biases, for the offset $\theta_{a\mapsto b}$ is:

$$\theta_{a\mapsto b} = \frac{1}{2}[(\tau^b - \tau^b_a) + (\tau^a_b - \tau^a)] \tag{3.15}$$

We associate the $\theta_{a\mapsto b}$ measurement with the average $\bar{\tau}_a = \frac{1}{2}(\tau^a + \tau^b_a)$ for times at a.

The offset values $\theta_{a\mapsto b}(\tau_a)$ capture the nonlinear clock behavior displayed by the RBS residuals. We compare the linear residuals for the offset values with those for RBS. Figure 3.8 shows the residuals for the discrete estimates of $\theta_{a\mapsto b}(\tau_a)$, with the line of least squares $\theta_{a\mapsto b}(\tau_a) = m_\theta \tau_a + k_\theta$ as found by linear regression removed. Figure 3.9 shows the difference between the offset residuals without drift correction. Since the mean function value over time is zero, $\theta_{a\mapsto b}(\tau_a)$ reproduces the observed nonlinear clock behavior. $\theta_{a\mapsto b}(\tau_a) - (m_\theta \tau_a + k_\theta)$ and those for RBS.

To find $\theta_{a\mapsto b}(\tau_a)$ for an arbitrary time $\tau_a = \beta_a(t + \alpha_a)$, we find the closest messages to τ_a and then add a correction term for clock drift. The clock drift term depends on the time elapsed between τ_a and the time of the $\theta_{a\mapsto b}$ measurement, $\bar{\tau}_a = \frac{1}{2}(\tau^a + \tau^b_a)$. We make the assumption that the error terms e for send and receive times are independent with standard deviation σ. The error from the SNTP equation terms

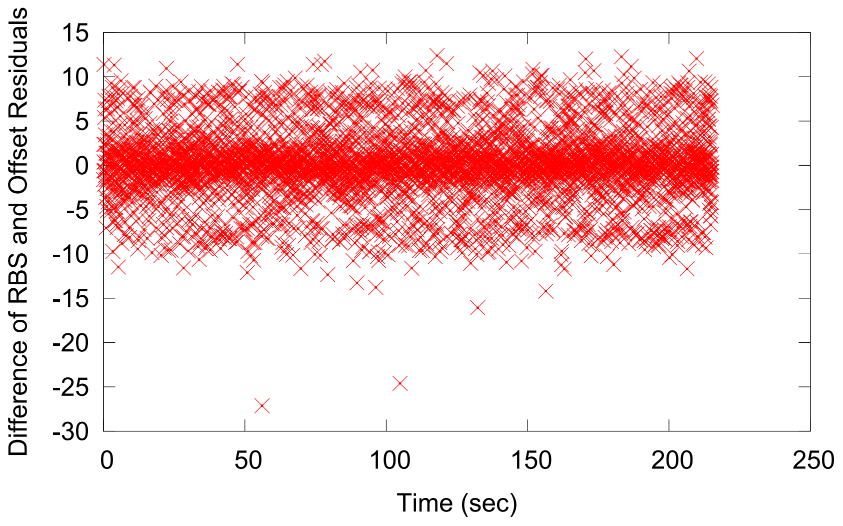


Figure 3.9: Difference of RBS and $\theta_{a\mapsto b}$ residuals

dominates because $\left|\frac{\beta_b}{\beta_a} - 1\right| < 10^{-4} \ll 1$.

$$\theta_{a\mapsto b}(\tau_a) = \frac{1}{2}[(\tau^b - \tau_a^b) + (\tau_b^a - \tau^a)] + \left(\frac{\beta_b}{\beta_a} - 1\right)\left(\tau_a - \frac{1}{2}(\tau_a^b + \tau^a)\right) \tag{3.16}$$

$$= \beta_b\alpha_b - \beta_a\alpha_a + (\beta_b - \beta_a)t + e$$

$$= \beta_b\alpha_b - \beta_a\alpha_a + \left(\frac{\beta_b}{\beta_a} - 1\right)(\tau_a - \beta_a\alpha_a) + e \tag{3.17}$$

Equation 3.16 describes one line segment of the piecewise linear function, for the nearest round of messages. Since any two clocks will run at at least slightly different rates, that is $\beta_b \neq \beta_a$, $\theta_{a\mapsto b}(\tau_a)$ will be a nonconstant. The piecewise linear offset function $\theta_{a\mapsto b}(\tau_a)$ will be linear if and only if $\frac{\beta_b}{\beta_a}$ is constant.

3.4.2 Estimation of Clock Drift $\frac{\beta_b}{\beta_a}$

To compute the drift corrected offset in equation 3.16 and make accurate distance measurements, we must estimate relative clock drift, $\frac{\beta_b}{\beta_a}$. We estimate clock drift using the set of messages sent from sender a and arriving at receiver b as shown in Figure 3.10. For index i corresponding to time, point clock drift can be estimated

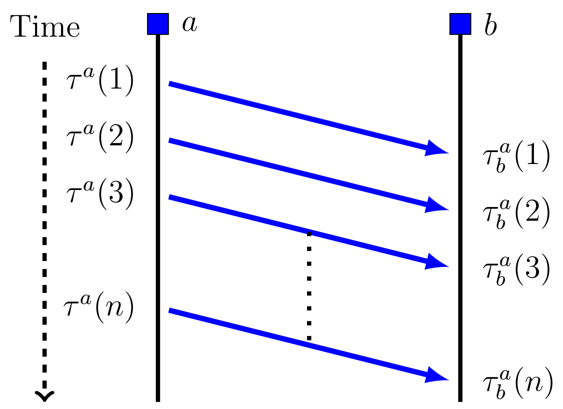


Figure 3.10: Messages for clock drift

using the simple slope equation

$$\widehat{\left[\frac{\beta_b}{\beta_a}\right]}(i) = \frac{\tau_b^a(i+k) - \tau_b^a(i)}{\tau^a(i+k) - \tau^a(i)} \tag{3.18}$$

Point Clock Drift Error Estimation

We estimate the point clock drift error due to clock measurement errors in our basic clock model. We consider the estimate for any two messages.

$$\widehat{\left[\frac{\beta_b}{\beta_a}\right]} = \frac{\tau_b^a(2) - \tau_b^a(1)}{\tau^a(2) - \tau^a(1)} \tag{3.19}$$

We assume the following:

1. Clock measurement errors e are independent and identically distributed with standard deviation σ.

2. e is relatively small when compared to the time between messages, $\sigma \ll \tau^a(2) - \tau^a(1)$.

3. Clock drifts are bounded: $\beta = 1 + \delta$, where $|\delta| < 10^{-4}$.

4. The distance at time t_2, $d_2 = d(a,b)(t_2)$ is nearly the same as that at time t_1,

$d_1 = d(a,b)(t_1)$.

$$|d(a,b)(t_2) - d(a,b)(t_1)| < \epsilon \tag{3.20}$$

The point estimate of clock drift with accompanying error terms is given in Equation 3.21. For the full derivation, see Appendix A.1.

$$\begin{aligned} \widehat{\left[\frac{\beta_b}{\beta_a}\right]} &= \frac{\tau_b^a(2) - \tau_b^a(1)}{\tau^a(2) - \tau^a(1)} \\ &\approx \frac{\beta_b}{\beta_a} + \frac{d_2 - d_1}{t_2 - t_1} + \frac{2\sigma}{t_2 - t_1} \end{aligned} \tag{3.21}$$

The clock drift error due to measurement error is $\frac{2\sigma}{t_2 - t_1}$. To reduce the single measurement error, the time between messages, $t_2 - t_1$, can be increased. This error must be balanced with consideration for the rate of measurements; taking more frequent measurements means that the accuracy of each measurement is decreased.

The clock drift error due to movement is $\frac{d_2 - d_1}{t_2 - t_1} = \frac{d(a,b)(t_2) - d(a,b)(t_1)}{t_2 - t_1}$. A node moving at human walking speeds (< 2 m/s) will negligibly affect the clock drift calculation; with 25 ns clock ticks, clock drift error due to walking is $< \frac{.25}{40 * 10^6} = 6.25 * 10^{-9}$ or less than 0.01 ppm.

Empirical Clock Drift Results

For the data from the experiment described in Section 3.3.1, we estimate the clock drift between nodes a and b using two different values of k for Equation 3.18 and apply an exponential filter to minimize error. For this data set, all nodes are stationary; there is no clock drift error resulting from node movement.

Computing clock drift from consecutive messages ($k = 1$) is very noisy. Figure 3.11 shows the clock drift computed using $k = 1$. The clock drift values vary from approximately $-11 * 10^{-6}$ to $8 * 10^{-6}$, with values clustered about $-2 * 10^{-6}$. This suggests that the true clock drift value is not changing as quickly as the graph points, and the clock measurement error is the noise source.

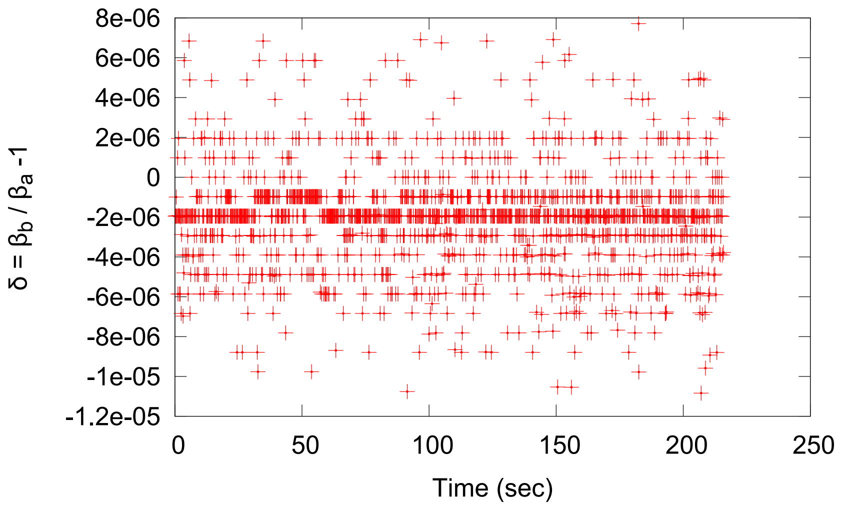


Figure 3.11: $\delta = \widehat{\left[\frac{\beta_b}{\beta_a}\right]} - 1$ with $k = 1$

Individual clock drift measurement error is reduced by decreasing the ratio of timestamping errors to the time between messages as predicted by Equation 3.21. We increase the time between messages by increasing the value of k. Figure 3.12 shows the clock drift computed with $k = 20$. The range of clock drift values is between $-2.6 * 10^{-6}$ and $-1.4 * 10^{-6}$. This decrease in range from the $k = 1$ case is consistent with a clock drift value of approximately $-2 * 10^{-6}$ and the twenty-fold increase in the time between messages.

To further decrease variability in the clock drift estimate, we apply an exponential filter to successive measurements with $w = 0.995$. The parameter w is chosen to minimize variability with the assumption that clock drift values will change relatively slowly and continuously. The ratio $\frac{\beta_b}{\beta_a}(i)$ is initialized with $\frac{\beta_b}{\beta_a}(0) = 1$ and is estimated for the time corresponding to index i:

$$\frac{\beta_b}{\beta_a}(i) = w * \frac{\beta_b}{\beta_a}(i-1) + (1-w)\widehat{\left[\frac{\beta_b}{\beta_a}\right]}(i) \tag{3.22}$$

The results of applying the exponential filter to the point clock drift estimates with $k = 20$ appear in Figure 3.13. As expected from the nonlinear offset plot Figure

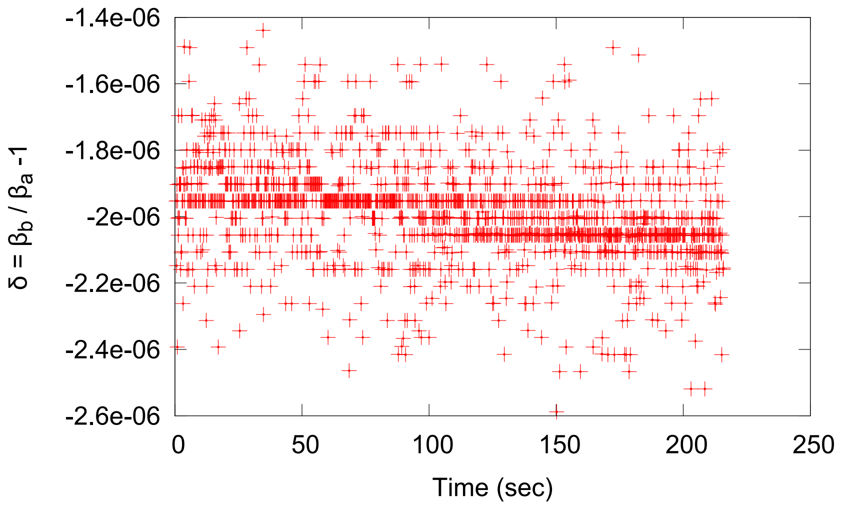


Figure 3.12: $\delta = \widehat{\left[\frac{\beta_b}{\beta_a}\right]} - 1$ with $k = 20$

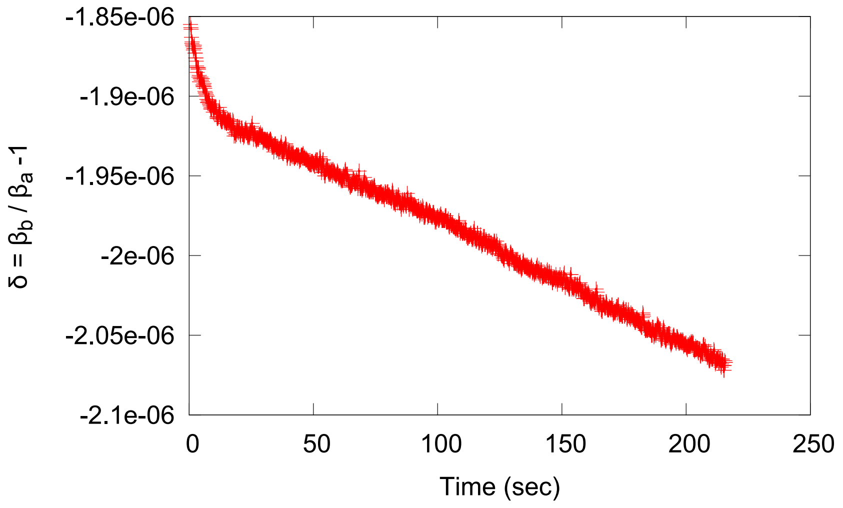


Figure 3.13: Exponential filter to estimate $\delta = \frac{\beta_b}{\beta_a} - 1$

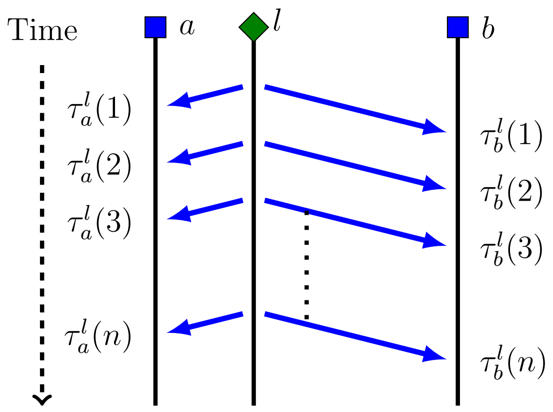


Figure 3.14: Reference Broadcast Synchronization

3.8, the ratio $\frac{\beta_b}{\beta_a}$ is nonconstant. From the RBS linear regression calculation for this data, $\delta = \frac{\beta_b}{\beta_a} - 1 = -1.99 * 10^{-6}$. For the exponential filter, the value varies between $-1.91 * 10^{-6}$ and $-2.07 * 10^{-6}$. Unlike the point clock drift estimates, the exponential filter range results from a changing clock drift rate rather than random error. When we require estimates of clock drift $\frac{\beta_b}{\beta_a}$, we will use the exponential filter version.

Clock Drift From Fixed Node

An alternative method of computing clock drift uses a reference node l with fixed location, as in reference broadcast synchronization [34]. A broadcast environment is required for a single signal from l to be received by both a and b. The point clock drift estimate is

$$\widehat{\left[\frac{\beta_b}{\beta_a}\right]} = \frac{\tau_b^l(i+k) - \tau_b^l(i)}{\tau_a^l(i+k) - \tau_a^l(i)} \tag{3.23}$$

The capability to timestamp received messages is sufficient for this technique; no timestamping of sent messages is required.

The error analysis is almost same as that for messages from one participant to another, with the exception that all timestamps are now from receive events. If send timestamp variability is higher than receive timestamp variability, this technique is more accurate.

3.5 Summary

In this chapter, we saw the significance of nonlinear behavior for a pair of Atheros-based wireless cards with inexpensive clocks, which have behavior representative of wireless network cards. We showed how to estimate two important quantities for clock pairs: offset and clock drift. These measures enable us to describe nonlinear clock behavior between two nodes empirically using a piecewise linear function. This function's line segments have slopes that are clock drift values, and the line segment intercepts are the offset values. Offsets are measured using rounds of messages between the two nodes, and clock drift is measured using an exponential filter applied to nonconsecutive messages from one node to the other. In Chapter 4, we build upon these two measures to develop a consensus synchronization system for any number of nodes.

Chapter 4

Consensus Clock Synchronization

Consensus clock synchronization is a distributed technique by which a set of clocks Ω can map their local times to a consensus timescale. This technique is flat rather than hierarchical like NTP [62] or IEEE 1588 [1]; all nodes are treated equally so there is no root node. It is suited to situations where the clocks are of similar quality, rather than situations where inaccurate clocks like quartz oscillators can be synchronized to very accurate atomic clocks. Each node communicates only with its neighbors, so no routing is required. It is distributed because each node computes using only local information.

Consensus clock synchronization is an extension of the Cyclone Network Synchronization system by Trinh [75]. Trinh demonstrated that local estimates of pairwise clock drift lead to a single virtual global clock. We extend this virtual global clock to a method that requires no precise adjustment of message send times in contrast to both Cyclone and the biologically inspired pulse-coupled oscillators presented by Hong and Scaglione [43] and extensions by Lucarelli and Wang [55]. Our method is similar to that presented by Denis, Pierrot, Abou-Rjeily for UWB networks [29] but without adjustments to either clock frequency or clock offsets. The method suffers no degradation over multiple hops in contrast to the reference broadcast synchronization technique presented by Elson, Girod, and Estrin [34].

The techniques for consensus clock synchronization are closely related to consensus agreement, first formulated by DeGroot [28], which describes how distributed nodes can compute a common distribution or point value. A good overview of consensus agreement results is given by Olshevsky and Tsitsiklis in [65]. One application of consensus agreement is to an averaging algorithm which computes the average of the values of a variable stored at each node in a distributed fashion.

Consensus clock synchronization computes a consensus linear clock model. The slope of this linear model is the consensus clock drift rate. Note that the consensus may change over time due to variations in the node clock frequencies, requiring each node to adjust its parameters.

Consensus clock synchronization requires estimation of pairwise clock drift and offset for network neighbors. To compute clock drift and offset, the ability to time-stamp outgoing and incoming messages is required.

Each node uses a piecewise linear function f to map its local time to the consensus time scale. This is similar to the offset function Equation 3.16, but in this case, the functions f will bring all nodes into agreement. All time comparisons are made after converting local times to this consensus time scale. Node a translates its local clock time τ_a to the corresponding consensus time τ_* using

$$\tau_* = f_a(\tau_a) = \frac{\beta_*}{\beta_a}\tau_a + \theta_{a\to*} \tag{4.1}$$

We formally define β_* in Section 4.2 and $\theta_{a\to*}$ in Section 4.3.

The clocks are synchronized in the sense that for any time t and small ϵ,

$$|f_a(\tau_a(t)) - f_b(\tau_b(t))| < \epsilon \tag{4.2}$$

The synchronization is virtual because the clocks do not synchronize to any particular event. Events such as message sending are not adjusted, and clock values are not adjusted. The clocks do not display the same time values and no clock need display the virtual global time.

We model the network as a weighted, undirected graph $G(\Omega, E)$, where Ω is the set of nodes and E is the set of edges connecting the nodes. Network nodes that are connected can communicate in either direction. An edge's weight is the propagation delay over a link. We assume that all nodes are self-connected, $\forall \omega \in \Omega, (\omega, \omega) \in E$. Let $\Omega_a = \{\omega | \omega \in \Omega, (a, \omega) \in E\}$ be the set of neighbors of a.

In this chapter, we prove the convergence of consensus clock synchronization assuming linear clock behavior and that receive and send timestamp bias can be corrected. Consensus clock synchronization techniques are expected to work unmodified for nonlinear clock behavior observed in Chapter 3, but it is unknown how to show this experimentally without an implementation including receive and send bias corrections. These bias corrections are necessary to compute a single function mapping all of a node's times to consensus times because send times must be corrected for send bias separately from receive times, which must be corrected for receive bias. Without these bias corrections, general synchronization is not known to be possible, but differences of times corresponding to distances can be computed, which is shown in Chapter 5.

4.1 Mathematical Background

The Perron-Frobenius Theorem and its extensions are important results relating to computation of eigenvectors for particular classes of matrices. They describe sufficient conditions for the power method to generate the eigenvector associated with the principle eigenvalue of a matrix. In particular, we would like to know whether the principle eigenvalue is real, simple and strictly greater in magnitude than all other eigenvalues. In this case, we can apply the power method for computing the eigenvector corresponding to the principle eigenvalue.

For details and proofs of theorems in this background section, see *Special Matrices and their Applications in Numerical Mathematics* by Fiedler [37].

Definition 4.1. *A simple eigenvalue has algebraic multiplicity one; the eigenvalue is a root of order one of its matrix's characteristic polynomial.*

We use the following notation for the n dimensional column vector containing only 0 or 1 entries. The dimension will be apparent from the context.

$$\mathbf{0} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \qquad \mathbf{1} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$$

4.1.1 Matrix Digraphs

The digraph (directed graph) of a matrix is a graph with edges between the nodes with corresponding nonzero matrix entries. There is a strong relationship between the structure of a matrix and its digraph.

Definition 4.2. *Let A be an $n \times n$ square matrix. The digraph of A is a graph $G(\Omega, E)$ with $|\Omega| = n$ nodes such that $(\omega_1, \omega_2) \in E \iff a_{\omega_1,\omega_2} > 0$*

Theorem 4.1. *A square matrix is irreducible if and only if its digraph is strongly connected.*

Proof. See [37]. □

In other words, if there is a path between every two nodes in the network digraph, the square matrix is irreducible. For consensus clock synchronization, we are only concerned with this case.

4.1.2 Perron-Frobenius and Extensions

Definition 4.3. *The spectral radius of a matrix A is $\rho(A) = \max\{|\lambda| : \lambda \in \mathcal{E}(A)\}$, where $\mathcal{E}(A)$, the spectrum of A, is the set of all eigenvalues of A.*

Definition 4.4. *A nonnegative matrix A consists of only nonnegative entries $a_{ij} \geq 0$. Likewise, a nonnegative vector x consists of only nonnegative entries $x_i \geq 0$.*

Theorem 4.2 (Perron-Frobenius Theorem)**.** *Let A be a nonnegative, irreducible square matrix of order n, $n > 1$. Then $\rho(A)$ is a simple positive eigenvalue of A and there is a positive eigenvector belonging to the eigenvalue $\rho(A)$. No nonnegative eigenvector belongs to any other eigenvalue of A.*

Proof. See [37]. □

Theorem 4.3. *Let A be an irreducible nonnegative square matrix of order n. Let h be a positive integer. The following properties of A and h are equivalent:*

1. *There exist exactly h distinct eigenvalues of A whose moduli are equal to $\rho(A)$.*
2. *The greatest common divisor of the lengths of all the cycles in the digraph of the matrix A is h.*

Proof. See [37]. □

For aperiodic ($h = 1$), strongly connected graphs, the principle eigenvalue of A will be strictly greater than all other eigenvalues. This eigenvalue is simple, and the eigenvector associated with this eigenvalue is unique. To compute this eigenvector, we can use the power method.

4.1.3 Power Method

The power method is a relatively simple method to compute the eigenvector associated with the principle eigenvalue.

Theorem 4.4. *Let A be a square matrix having unique eigenvalue λ satisfying $|\lambda| = \rho(A)$; suppose that λ is simple. Let v be an eigenvector of A^T belonging to λ. If z is an arbitrary vector such that $v^T z \neq 0$, then*

$$\lim_{k \to \inf} (\lambda^{-1} A)^k z = y \neq 0$$

and y is an eigenvector of A belonging to the eigenvalue λ.

Proof. See [37]. □

For any aperiodic, irreducible nonnegative square matrix, there exists a unique $\lambda = \rho(A)$ by the Perron-Frobenius Theorem. We can apply the power method to such a matrix.

For stochastic matrices, there is a stronger result independent of the initial starting vector z.

Theorem 4.5. *If A is an irreducible primitive (aperiodic) stochastic matrix, then the powers A^k converge (as $k \to \inf$) to the matrix $\mathbf{1}v^T$ of rank one, where*

$$v = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$$

is the eigenvector of A^T belonging to the eigenvalue 1, normalized in such a way that $v^T \mathbf{1} = 1$.

Proof. See [37]. □

All rows of $\mathbf{1}v^T$ are equal, so for any initial vector z, we get the following, for some scalar constant c.

$$\lim_{k \to \inf} A^k z = c\mathbf{1} \tag{4.3}$$

4.2 Clock Drift

The first of the two elements mapping local time to the consensus time is clock drift. First, we define the consensus clock rate.

Definition 4.5. *The consensus clock rate β_* is the average drift rate of the nodes in Ω.*

$$\beta_* = \frac{1}{|\Omega|} \sum_{\omega \in \Omega} \beta_\omega \tag{4.4}$$

Each node ω can compute time differences in the consensus time scale by multiplying local time differences by $\frac{\beta_*}{\beta_\omega}$.

By measuring the pairwise clock drift ratios of nodes using the technique in Chapter 3.4.2, we can compute $\frac{\beta_*}{\beta_\omega}$, the ratio of the average clock drift of all clocks in Ω to the clock drift from each node ω.

4.2.1 Clock Drift in Fully Connected Graph

The following matrix can be determined by estimating the clock drift ratios pairwise from Ω, the set of clocks.

$$B = \frac{1}{|\Omega|} \begin{bmatrix} 1 & \frac{\beta_b}{\beta_a} & \frac{\beta_c}{\beta_a} & \cdots \\ \frac{\beta_a}{\beta_b} & 1 & \frac{\beta_c}{\beta_b} & \cdots \\ \frac{\beta_a}{\beta_c} & \frac{\beta_b}{\beta_c} & 1 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{4.5}$$

It can be verified directly that $\lambda = 1$ is an eigenvalue of this matrix and the associated eigenvector is:

$$x = \begin{bmatrix} \frac{\beta_*}{\beta_a} \\ \frac{\beta_*}{\beta_b} \\ \frac{\beta_*}{\beta_c} \\ \vdots \end{bmatrix} \tag{4.6}$$

This eigenvector is strictly positive, so by the Perron-Frobenius Theorem, we know $\lambda = 1$ is the largest eigenvalue of B.

Each node can directly compute its corresponding eigenvector entry by averaging its locally available clock drift ratios. For node a:

$$\frac{\beta_*}{\beta_a} = \frac{1}{|\Omega|} \sum_{\omega \in \Omega} \frac{\beta_\omega}{\beta_a} \tag{4.7}$$

The resulting eigenvector entry is the ratio of the consensus clock drift β_* with one node's clock drift value. This enables each node to convert its local time scale to the same consensus time scale, which runs at rate β_*.

4.2.2 Clock Drift General Graph

Matrix B in Equation 4.5 is for the case when all nodes are directly connected. We now modify the entries of B based on the existing graph edges for when all nodes are not directly connected. Define k_{ω_1,ω_2} by whether there is an edge between ω_1 and ω_2.

$$k_{\omega_1,\omega_2} = \begin{cases} 0 & \text{if } (\omega_1, \omega_2) \notin E \\ 1 & \text{if } (\omega_1, \omega_2) \in E \end{cases} \tag{4.8}$$

The new matrix is

$$B = \begin{bmatrix} \frac{1}{|\Omega_a|}1 & \frac{k_{a,b}}{|\Omega_a|}\frac{\beta_b}{\beta_a} & \frac{k_{a,c}}{|\Omega_a|}\frac{\beta_c}{\beta_a} & \cdots \\ \frac{k_{b,a}}{|\Omega_b|}\frac{\beta_a}{\beta_b} & \frac{1}{|\Omega_b|}1 & \frac{k_{b,c}}{|\Omega_b|}\frac{\beta_c}{\beta_b} & \cdots \\ \frac{k_{c,a}}{|\Omega_c|}\frac{\beta_a}{\beta_c} & \frac{k_{c,b}}{|\Omega_c|}\frac{\beta_b}{\beta_c} & \frac{1}{|\Omega_c|}1 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{4.9}$$

Each row corresponds to relative clock drifts involving one node ω. The number of nonzero entries for the row corresponding to ω is $|\Omega_\omega|$.

We can verify directly that x,

$$x = \begin{bmatrix} \frac{\beta_*}{\beta_a} \\ \frac{\beta_*}{\beta_b} \\ \frac{\beta_*}{\beta_c} \\ \vdots \end{bmatrix} \tag{4.10}$$

which contains the scalars for the consensus time scale, is an eigenvector of B corresponding to $\lambda = 1$. This eigenvector is positive, so by the Perron-Frobenius Theorem, we know $\lambda = 1 = \rho(A)$.

The matrix B is nonnegative, and it is irreducible because we assume the network graph is strongly connected. Since all network graph edges are bidirectional and there are self-loops, B is aperiodic. Therefore we can apply the power method to find the eigenvector x corresponding to the eigenvalue $\lambda = \rho(A) = 1$.

$$x = \lim_{n\to\infty} B^n x_0 \tag{4.11}$$

Equivalently to the matrix multiplication, each node can iteratively compute its own eigenvector entry using only locally available clock drift ratios. Node a maintains the state $\frac{\beta_*}{\beta_\omega}(i), \forall \omega \in \Omega_a$. For the initial state, $\forall \omega \in \Omega$, $\frac{\beta_*}{\beta_\omega} = 1$. Node a computes

$$\frac{\beta_*}{\beta_a}(i+1) = \frac{1}{|\Omega_a|} \sum_{\omega \in \Omega_a} \left[(k_{a,\omega}) \frac{\beta_\omega}{\beta_a} * \frac{\beta_*}{\beta_\omega}(i) \right] \tag{4.12}$$

Node a then publishes its own $\frac{\beta_*}{\beta_a}$ value to its neighbors, which update their states accordingly. The process repeats iteratively and converges to the eigenvector entry. The iterative process may be independent of any additional measurements to update the matrix clock drift rate entries.

4.3 Offset

The second part of the mapping from local time to consensus time is offset. For the consensus timescale β_*, we first define the offset between clocks.

Definition 4.6. *$\theta_{a\to b}$ is the offset of a's clock from b's clock in the consensus timescale.*

$$\begin{aligned} \theta_{a\to b} &= \frac{1}{2}\left[\frac{\beta_*}{\beta_b}\left((\tau^b + \tau^a_b) - (s_b + r_b) \right) - \frac{\beta_*}{\beta_a}\left((\tau^a + \tau^b_a) - (s_a + r_a) \right) \right] \\ &= \beta_*(\alpha_b - \alpha_a) \end{aligned} \tag{4.13}$$

This computes the offset between two clocks in the consensus time scale from network message timestamps. For the remainder of this chapter, we will assume the card bias terms are known so they can be corrected and eliminated to leave Equation 4.14.

$$\theta_{a\to b} = \frac{1}{2}\left[\frac{\beta_*}{\beta_b}(\tau^b + \tau^a_b) - \frac{\beta_*}{\beta_a}(\tau^a + \tau^b_a) \right] \tag{4.14}$$

To use this equation, clock drifts must be computed first. For the full derivation including timestamp biases, see Appendix A.2.

We are most interested in the offset between each node's local times and the consensus time.

Definition 4.7. *$\theta_{a\to*}$ is the offset of a's clock from the consensus clock in the consensus timescale.*

$$\theta_{a\to*} = \frac{\beta_*}{|\Omega|}\sum_{\omega\in\Omega}(\alpha_\omega - \alpha_a) \tag{4.15}$$

$$= \frac{1}{|\Omega|}\sum_{\omega\in\Omega}(\theta_{a\to\omega}) \tag{4.16}$$

To see that the consensus clock offsets are consistent with the pairwise offsets, see Theorem A.1 in Appendix A.

4.3.1 Offset in Fully Connected Graph

First we define the matrix Θ, which contains all pairwise offsets from nodes in Ω. Each row of Θ contains the offsets for one node to each of its neighbors.

$$\Theta = \begin{bmatrix} \theta_{a\to a} & \theta_{a\to b} & \theta_{a\to c} & \cdots \\ \theta_{b\to a} & \theta_{b\to b} & \theta_{b\to c} & \cdots \\ \theta_{c\to a} & \theta_{c\to b} & \theta_{c\to c} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{4.17}$$

We wish to find the following vector

$$\Theta_* = \begin{bmatrix} \theta_{a\to*} \\ \theta_{b\to*} \\ \theta_{c\to*} \\ \vdots \end{bmatrix} \tag{4.18}$$

where each entry gives the offset between the local clock and the consensus clock. Node a computes Equation 4.16 directly by averaging its row of Θ to find $\theta_{a\to*}$. The other nodes compute their corresponding offsets similarly.

Any pair of Θ_* entries gives the corresponding pairwise offsets from the matrix Θ. For example, $\theta_{a\to*} - \theta_{b\to*} = \theta_{a\to b}$. See Appendix A.2.1 for details.

4.3.2 Offset in General Graph

In a general graph, nodes can measure their offsets with only a strict subset of nodes in Ω. Thus an alternative computation for $\theta_{a\to*}$ is necessary.

To compute $\theta_{a\to*}$, we use iterative techniques from consensus propagation. Each node's consensus offset is initialized to zero.

$$\forall\omega \in \Omega, \theta_{\omega\to*}(0) = 0 \tag{4.19}$$

Each node iteratively updates its consensus offset estimate using both its neighbors consensus offset estimates and current pairwise offsets from its neighbors.

$$\theta_{a\to*}(n+1) = \frac{1}{|\Omega_a|} \sum_{\omega\in\Omega_a} [\theta_{a\to\omega} + \theta_{\omega\to*}(n)] \tag{4.20}$$

After each iteration, each node ω publishes its updated $\theta_{\omega\to*}$ value to its neighbors.

4.3.3 Offset Matrix Analysis

Using matrix analysis, we prove that the offsets converge to the desired values.

$$\lim_{n\to\infty} \theta_{a\to*}(n) = \theta_{a\to*}$$

We define a stochastic matrix A based on the network graph edges to describe the offset information available to each node. This is similar to the normalized Laplacian matrix, but normalized differently using only one node's order in each row.

$$k_{\omega_1,\omega_2} = \begin{cases} 0 & \text{if } (\omega_1, \omega_2) \notin E \\ 1 & \text{if } (\omega_1, \omega_2) \in E \end{cases} \tag{4.21}$$

$$A = \begin{bmatrix} \frac{1}{|\Omega_a|} & \frac{k_{a,b}}{|\Omega_a|} & \frac{k_{a,c}}{|\Omega_a|} & \cdots \\ \frac{k_{b,a}}{|\Omega_b|} & \frac{1}{|\Omega_b|} & \frac{k_{b,c}}{|\Omega_b|} & \cdots \\ \frac{k_{c,a}}{|\Omega_c|} & \frac{k_{c,b}}{|\Omega_c|} & \frac{1}{|\Omega_c|} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{4.22}$$

In the row corresponding to node ω, there are $|\Omega_\omega|$ non-zero entries. Each row sums to 1 so the matrix is stochastic. Since A is stochastic, it will have principle eigenvalue $\lambda = 1$ and $(A - I)$ will be singular. Connections are bidirectional in our network graph, implying that $k_{\omega_1,\omega_2} = k_{\omega_2,\omega_1}$. As with the clock drift matrix, this matrix is aperiodic and irreducible because the corresponding digraph is strongly connected with self-loops.

The iterative computations for Θ_* can be rewritten in matrix form as follows.

$$\Theta_*(0) = \mathbf{0} \tag{4.23}$$

$$\Theta_*(n+1) = \text{diag}(A\Theta^T) + A\Theta_*(n) \tag{4.24}$$

We will prove

$$\lim_{n\to\infty} \Theta_*(n) = \Theta_* + c\mathbf{1} \tag{4.25}$$

For consensus synchronization, the $c\mathbf{1}$ term is irrelevant. When taking the difference between times at any two nodes, the c terms will cancel.

The following Lemma will be useful for simplifying a matrix geometric series:

Lemma 4.6.

$$(I - A)\left(\sum_{i=0}^{n-1} A^i\right) = (I - A^n) \tag{4.26}$$

Proof.

$$\begin{aligned}
\sum_{i=0}^{n-1} A^i &= I + A + A^2 + \ldots + A^{n-1} \\
(I - A)\left(\sum_{i=0}^{n-1} A^i\right) &= I + A + A^2 + \ldots + A^{n-1} \\
&\qquad - A - A^2 - \ldots - A^{n-1} - A^n \\
(I - A)\left(\sum_{i=0}^{n-1} A^i\right) &= I - A^n
\end{aligned}$$

□

Lemma 4.7. *Let A be the stochastic matrix in Equation 4.22, whose digraph is a finite, aperiodic, and strongly connected. Let Θ be the matrix of offsets stated above in Equation 4.17.*

$$\lim_{n\to\infty} A^n diag(A\Theta^T) = c\boldsymbol{1} \tag{4.27}$$

Where c is a real constant.

Proof. This is a straightforward result from Theorem 4.5 for the initial column vector diag$(A\Theta^T)$. All rows of $\lim_{n\to\infty} A^n$ are equal, so all elements of the resulting product vector are equal. □

Lemma 4.8. *Let A be the stochastic matrix based on a finite, strongly connected graph as defined in Equation 4.22. Let Θ be the matrix of offsets stated above in Equation 4.17.*

$$(I - A)\Theta_* = diag(A\Theta^T) \tag{4.28}$$

Proof. First we write $A\Theta_*$ and $\text{diag}(A\Theta^T)$ explicitly.

$$A\Theta_* = \begin{bmatrix} \sum_{\omega\in\Omega} \frac{k_{a,\omega}}{|\Omega_a|}\theta_{\omega\to *} \\ \sum_{\omega\in\Omega} \frac{k_{b,\omega}}{|\Omega_b|}\theta_{\omega\to *} \\ \vdots \end{bmatrix}$$

$$\text{diag}(A\Theta^T) = \begin{bmatrix} \sum_{\omega\in\Omega} \frac{k_{a,\omega}}{|\Omega_a|}\theta_{a\to \omega} \\ \sum_{\omega\in\Omega} \frac{k_{b,\omega}}{|\Omega_b|}\theta_{b\to \omega} \\ \vdots \end{bmatrix}$$

Now we examine the sum.

$$\begin{aligned} A\Theta_* + \text{diag}(A\Theta^T) &= \begin{bmatrix} \sum_{\omega\in\Omega} \frac{k_{a,\omega}}{|\Omega_a|}(\theta_{a\to\omega} + \theta_{\omega\to *}) \\ \sum_{\omega\in\Omega} \frac{k_{b,\omega}}{|\Omega_b|}(\theta_{b\to\omega} + \theta_{\omega\to *}) \\ \vdots \end{bmatrix} \\ &= \begin{bmatrix} \sum_{\omega\in\Omega} \frac{k_{a,\omega}}{|\Omega_a|}\theta_{a\to *} \\ \sum_{\omega\in\Omega} \frac{k_{b,\omega}}{|\Omega_b|}\theta_{b\to *} \\ \vdots \end{bmatrix} \\ &= \begin{bmatrix} \theta_{a\to *} \\ \theta_{b\to *} \\ \vdots \end{bmatrix} \\ &= \Theta_* \end{aligned}$$

By combining the Θ_* terms, we have

$$(I - A)\Theta_* = \text{diag}(A\Theta^T)$$

□

Theorem 4.9. *The sequence of offset computations in Equation 4.24 converges to* *the desired value plus a constant vector $c\boldsymbol{1}$, which has all equal entries.*

$$\lim_{n\to\infty} \Theta_*(n) = \Theta_* + c\boldsymbol{1} \tag{4.29}$$

Proof. The non-recursive formula for $\Theta_*(n)$ is

$$\begin{aligned}
\Theta_*(n) &= \left(\sum_{i=0}^{n-1} A^i \mathrm{diag}(A\Theta^T)\right) + A^n\Theta_*(0) \\
\Theta_*(n) &= \left(\sum_{i=0}^{n-1} A^i\right) \mathrm{diag}(A\Theta^T) \\
(I-A)\Theta_*(n) &= (I-A)\left(\sum_{i=0}^{n-1} A^i\right) \mathrm{diag}(A\Theta^T) \\
&= (I-A^n)\,\mathrm{diag}(A\Theta^T) \\
&= \mathrm{diag}(A\Theta^T) - A^n\mathrm{diag}(A\Theta^T)
\end{aligned}$$

Applying Lemma 4.8, we have

$$(I-A)\Theta_*(n) = (I-A)\Theta_* - A^n\mathrm{diag}(A\Theta^T)$$

Taking the limit and applying Lemma 4.7

$$\lim_{n\to\infty}(I-A)\Theta_*(n) = (I-A)\Theta_* + c\mathbf{1}$$

We multiply by $(I-A)$ to get an equality relating $\Theta_*(n)$ and Θ_*.

$$\begin{aligned}
&\lim_{n\to\infty}(I-A)^2(\Theta_*(n) - \Theta_*) = (I-A)c\mathbf{1} \\
&\lim_{n\to\infty}(I-A)^2(\Theta_*(n) - \Theta_*) = 0
\end{aligned}$$

We can now analyze using the kernel of the matrix multiplication transformation, which is the null space of $(I - A)^2$.

$$\lim_{n\to\infty} \Theta_*(n) \equiv \Theta_* \mod (I - A)^2$$

By the Perron-Frobenius Theorem, $\lambda = 1$ is a simple (multiplicity 1) eigenvalue of A. The eigenspace of A corresponding to eigenvalue $\lambda = 1$ with eigenvector $\mathbf{1}$, which is the null space of $(I - A)^2$, is spanned by $\mathbf{1}$.

$$\lim_{n\to\infty} \Theta_*(n) = \Theta_* + c\mathbf{1}$$

□

4.4 Verification by Simulation

We verify consensus clock synchronization by simulation using nodes with fixed clock drifts and perfect timestamping precision. This corresponds to one line segment of the piecewise linear model. Verification using real hardware requires timestamping with known or measurable receive and send biases.

For a set of nodes Ω, for each node ω, we select clock drift $\beta_\omega = 1 + \delta_\omega$ and initial starting values α_ω. Each node broadcasts two messages at specified times, which is sufficient to characterize the offset and clock drift values. Table 4.1 shows the set of inputs.

Receive times for these messages are computed for neighbors according to the connectivity and distances in the graph of Figure 4.1. For example, the second message sent by node b is at local time $\tau^b(2) = 1100$. This corresponds to global time $t^b(2) = \frac{\tau^b(2)}{\beta_b} - \alpha_b = \frac{1100}{0.99} - 100 = 1011.111$. Node a will receive this message at $\tau^b_a(2) = \beta_a(t^b(2) + \alpha_a + d(a, b)) = 1.01 * (1011.111 + 0 + 1) = 1023.232$.

Using these simulated network message times, we apply 100 iterations of the clock

Node ID (ω)	α	δ	$\tau^{\omega}(1)$	$\tau^{\omega}(2)$
a	0	0.01	0	1000
b	100	-0.01	100	1100
c	200	0.05	350	1350
d	300	0.03	200	1200
e	400	0.00	500	1500
f	410	-0.02	510	1600
g	411	-0.07	550	1700
h	600	0.04	800	1900

Table 4.1: Node property inputs for simulation

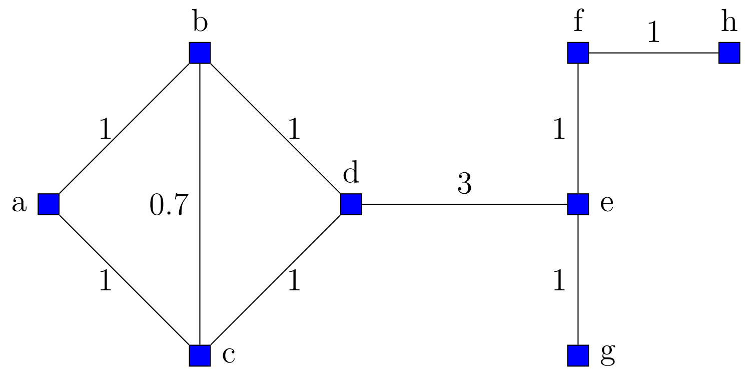


Figure 4.1: Network graph with distances for consensus synchronization simulation

Node ID (ω)	$\frac{\beta_*}{\beta_\omega} - 1$	$\theta_{\omega\to*}$
a	-0.002665	280.950
b	0.017483	180.221
c	-0.040659	79.490
d	-0.022031	-21.236
e	0.007307	-121.955
f	0.027864	-132.020
g	0.083126	-133.033
h	-0.031436	-323.405

Table 4.2: Consensus parameters $\frac{\beta_*}{\beta_\omega}, \theta_{\omega\to*}$

$$\begin{tikzcd}
 & t \arrow[dl] \arrow[dr] & \\
\tau_a = \beta_a(t + \alpha_a) \arrow[d] & & \tau_b = \beta_b(t + \alpha_b) \arrow[d] \\
t_a = \frac{\beta_*}{\beta_a}\tau_a + \theta_{a\to*} \arrow[dr] & & t_b = \frac{\beta_*}{\beta_b}\tau_b + \theta_{b\to*} \arrow[dl] \\
 & |t_a - t_b| < \epsilon \approx 0 &
\end{tikzcd}$$

Figure 4.2: Consensus clock synchronization verification scheme

drift and offset calculations to compute $\frac{\beta_*}{\beta_\omega}$ and $\theta_{\omega\to*}$ values. These results are shown in Table 4.2.

For time $t = 2000$, we compute each node's local clock time τ. Using these local times, we compute the consensus time for each node ω using $\frac{\beta_*}{\beta_\omega}$ and $\theta_{\omega\to*}$. The computed local and consensus times are shown in Table 4.3. The difference in the consensus times for any pair of nodes is small, < 0.025. Note that the computed consensus time is not equal to the input time because of the $c\mathbf{1}$ term in Equation 4.29.

These simulation results verify that a set of connected nodes can achieve synchronization by passing timestamped network messages amongst themselves in a distributed manner. For a practical implementation, clock precision and timestamping

Node ID (ω)	$\tau_\omega(t)$	t_ω
a	2020	2295.567
b	2079	2295.569
c	2310	2295.569
d	2369	2295.573
e	2400	2295.583
f	2361.8	2295.589
g	2242.23	2295.584
h	2704	2295.592

Table 4.3: Simulation verification results for $t = 2000$

bias and errors must be considered. The accuracy of consensus synchronization depends on these as well as the network graph diameter and the number of iterations to compute $\frac{\beta_*}{\beta_\omega}$ and $\theta_{\omega\to*}$.

4.5 Summary and Applications

Consensus clock synchronization provides distributed synchronization for free running clocks to a timescale running at clock rate β_*. As an average of the member node clock rates, the stability of this rate is better than any single node's rate. The computation is simple; each node ω sends timestamped messages containing its current values of $\frac{\beta_*}{\beta_\omega}$ and $\theta_{\omega\to*}$. By timestamping and using the contents of its neighbors packets, ω has sufficient information to update $\frac{\beta_*}{\beta_\omega}$ and $\theta_{\omega\to*}$. These values enable all nodes to use a common time scale with known offsets from their local clocks without frequency or offset adjustments to the physical clocks.

Future work is needed to evaluate practical accuracy for consensus clock synchronization for a system with measurable timestamping biases. Without knowing the separate send and receive timestamping biases, it is not meaningful to compare receive times at two separate nodes because the receive timestamp biases may be different. The best bias information known to be computable from the available timestamps, presented in Chapter 5.5, does not separate the send and receive bias values for any node.

It is not necessary, however, to know the separate send and receive timestamp biases to apply the consensus clock drift to measure distances. This is covered in Chapter 5.

Chapter 5

PinPoint

PinPoint is a distributed, linear complexity algorithm requiring no external clock synchronization to determine spatial geometry first presented by Youssef et al. [85]. The PinPoint system converts timestamp information into the distance primitives TOA and TDOA, which are inputs to optimization location problems. In this chapter, we extend the original PinPoint system, which uses time of arrival to compute TOA distance between PinPoint nodes.

We add three TDOA variants to the original PinPoint TOA, bringing the total number of variants to four.

TOA Pairwise distances can be computed between participants using send and receive times.

TDOA Active Target Participants measure send and receive times to compute differences of distances for actively transmitting targets. This was first described in [57] without consensus clock drift.

TDOA with No Send Times Participants measure only receive times to compute differences of distances for actively transmitting targets.

TDOA Low Traffic Mobile A participating mobile node uses its receive times for

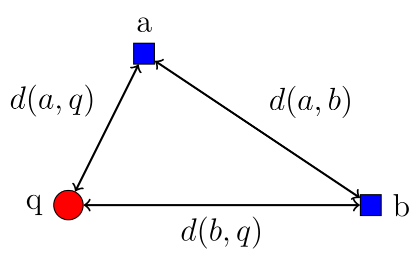


Figure 5.1: Distances for TOA, TDOA

messages between other participating nodes to locate itself.

To locate the target node q, the PinPoint TOA computes $d(b, q)$ and $d(a, q)$. The PinPoint TDOA variants compute $d(b, q) - d(a, q)$. These distances are shown in Figure 5.1.

Algorithmic changes to improve the accuracy of the primitive distance measurements include the following items.

send and receive timestamping biases Significant timestamp biases are corrected to improve distance accuracy.

consensus time scale By using the average clock drift for a set of nodes, we reduce the distance error from clock drift estimates.

robust clock drift estimates The estimation of β ratios is improved by replacing the simple slope calculation of [85] with an exponential filter.

PinPoint is designed for broadcast communication environments, where network messages can be heard by nodes in addition to the source and destination. A main feature of the TOA variant is $O(n)$ message overhead, where n is the number of participating nodes. With $O(n)$ messages, PinPoint can measure all $\frac{n(n+1)}{2}$ pairwise distances. This contrasts to the $O(n^2)$ messages required for point-to-point communication to measure the same distance information. The TDOA variants require single messages to be received at multiple participants, which is only possible in broadcast environments.

5.1 Implementation

To implement the PinPoint location system, the following capabilities are relevant:

1. timestamp using MAC clock to eliminate interrupt latency
2. timestamp at better than 1 μs precision
3. timestamp sent messages (necessary for all PinPoint variants except one TDOA variant)
4. timestamp received messages
5. maintain own free-running clock
6. timestamp management packets or packets with other destinations

The ability to timestamp management packets or packets with other destinations is required for PinPoint TDOA. Stranger nodes cannot be expected to multicast messages to anchor nodes, which must timestamp the same packets to compute TDOA. To timestamp the same packets, anchor nodes must be able to process packets from a stranger node, which could include beacons, data packets, or acknowledgments. This may be achieved using a level of promiscuous mode, in which packets that are usually ignored by the wireless driver are passed to the application for processing.

To measure the clock offsets and drifts as shown in Chapter 3, we need driver support for timestamping incoming messages, and it is highly desirable to timestamp outgoing messages also. To measure distance in real-time, we need hardware that supports accurate timestamping with precision better than the standard 802.11 clock precision of $1\mu s$.

Acquiring hardware capable of timestamping with better precision is a major practical problem. The two known off-the-shelf 802.11 hardware platforms supporting better precision are produced by Atheros and G2 Microsystems, which has been

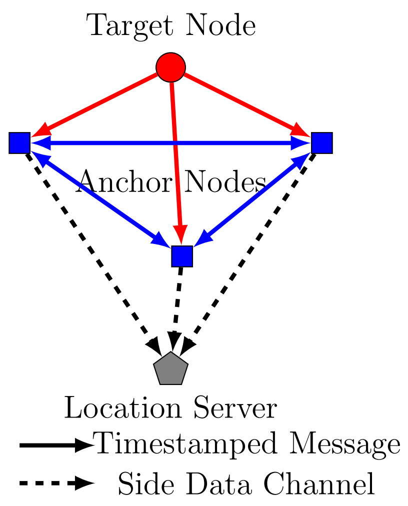


Figure 5.2: PinPoint data movement

acquired by Roving Networks. These two platforms also satisfy the other desired PinPoint capabilities.

PinPoint implementation has focused on collecting the necessary timestamp information at each node and communicating that information to a central location server for all processing. PinPoint nodes are designed to be as simple as possible, and all complex computations take place at the location server. The location server computes TOA and TDOA values for the collected timestamps from the PinPoint nodes and solves the appropriate optimization problem for location. The movement of timestamp data is shown in Figure 5.2. Communication with the location server may be over a side data channel, or it may be part of the wireless network used for the timestamped messages.

Based on the specific platform feature set available, the design of the PinPoint implementations to achieve the exchange of timestamped messages are different. For example, for the Atheros platform, the timestamped messages are management beacons from APs, while for the RN-134, the timestamped messages are data UDP packets. For a detailed discussion, see Appendix D.

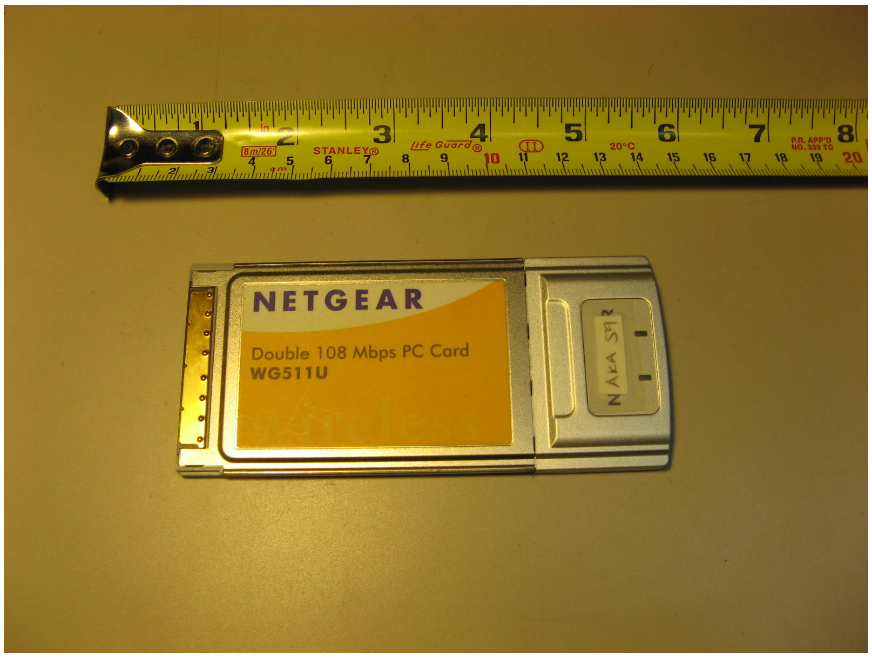


Figure 5.3: Netgear Atheros-based network card

5.1.1 Atheros Network Cards

Atheros is a chipset supplier to numerous commercial 802.11 card companies, including Netgear and Ubiquiti. In Figure 5.3 an example Atheros-based network card is shown that plugs into a standard laptop PC card slot, though these are now becoming obsolete. Under the GNU/Linux operating system with a modified version of the madwifi driver, the Atheros chipset can be programmed to timestamp with a 40 MHz clock, though this disrupts communication with normal 802.11 devices.

Using Atheros cards for PinPoint, each participant has a wireless card operating in AP and monitor mode. In AP mode, each card sends beacons at regular intervals. Each beacon contains the AP's send time in the beacon payload. In monitor mode, each card records receive times for beacons from all other anchor nodes as well as any stranger nodes. Each participant stores the timestamps for each transmitting node it can hear in a separate circular buffer.

When operating in AP and monitor mode, the Atheros card cannot send data packets. A separate network interface is required to communicate timestamps to the

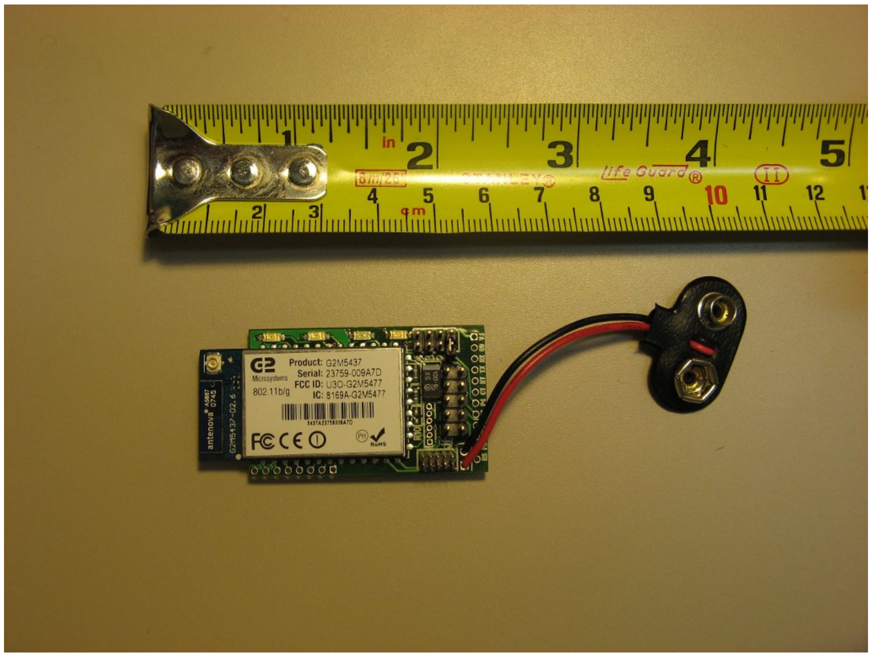


Figure 5.4: Roving Networks RN-134

location server.

5.1.2 G2-based Devices

G2 Microsystems, acquired by Roving Networks in 2010, develops system-on-a-chip products with 32-bit processor, 802.11b/g networking, and support for other add-on sensors. These standalone programmable products are designed for mobile, low-power applications. An example, the RN-134, is shown in Figure 5.4.

The RN-134 does not support AP mode, but it can record timestamps for any outgoing packet. The timestamp cannot be included in the outgoing packet as they can in beacon payloads.

Each participant node sends UDP packets to the location server on a regular basis to both exchange messages with other participants and convey those timestamps for those messages to the location. Each UDP packet contains a sequence number; the send time of the last UDP packet from this node; the MAC addresses the node has been instructed to listen for; and for each MAC address, the receive times and sequence numbers for the last five UDP packets from this address. Each participant node listens for UDP packets, which are addressed to an AP, from other participant nodes by listening in promiscuous mode. When receiving one of these UDP packets from another sender, a participant node updates its circular buffer for that sender by replacing the oldest sequence number and receive time pair with the values for the new packet.

The server pairs the send and receive timestamps for each message using the MAC addresses and sequence numbers.

5.2 Piecewise, Locally Linear Model

We use a piecewise, locally linear time model for all PinPoint variants. There is a basic message set that defines a single TOA or TDOA measurement. Within this message set, we treat all clock drifts as constant, which results in a locally linear model. Across message sets, the clock drifts may vary, which results in an overall piecewise linear model of time.

Each basic message set provides a single distance measurement sample. Assuming the errors for each basic message set are independent, we can average distance samples to reduce distance error without reducing timestamp errors or improving timestamp precision. If the standard deviation for a single distance measurement is σ_d, we can use n measurements to reduce the standard deviation of the average to $\frac{\sigma_d}{\sqrt{n}}$. To estimate location in real-time to a specified accuracy, we need to collect the corresponding number of samples in a few seconds. Both more accurate timestamps and better timestamp precision reduce the number of samples required to accurately measure distance and then estimate location.

We now present the PinPoint TOA and TDOA variants with their basic message sets, followed by the computations for card bias. The derivations are presented in

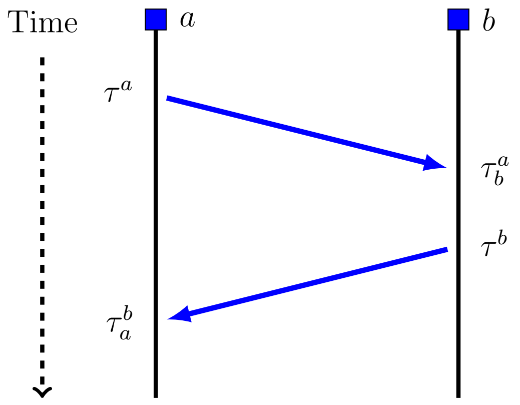


Figure 5.5: PinPoint TOA Basic Message Set

Appendix A.

5.3 PinPoint TOA

PinPoint TOA adds consensus clock drift correction to the basic roundtrip time delay technique of SNTP. Figure 5.5 shows the basic message set for PinPoint TOA, which consists of a round of messages between two participant nodes. We write the equations for the send and receive times using the clock model from Chapter 3.

$$\tau^a = \beta_a(t^a + \alpha_a) + s_a + e^a \tag{5.1}$$

$$\tau_b^a = \beta_b(t^a + \alpha_b + d(a,b)) + r_b + e_b^a \tag{5.2}$$

$$\tau^b = \beta_b(t^b + \alpha_b) + s_b + e^b \tag{5.3}$$

$$\tau_a^b = \beta_a(t^b + \alpha_a + d(a,b)) + r_a + e_a^b \tag{5.4}$$

Solving for $d(a,b)$, the PinPoint TOA distance equation is given by:

$$\beta_* d(a,b) = \frac{1}{2}\left(\frac{\beta_*}{\beta_a}\left[\left(\tau_a^b - \tau^a\right) - (r_a - s_a)\right] - \frac{\beta_*}{\beta_b}\left[\left(\tau^b - \tau_b^a\right) - (s_b - r_b)\right]\right) \tag{5.5}$$

Since β_* will be very close to one, $\beta_* d(a,b) \approx d(a,b)$. The complete derivation is presented in Appendix A.

Variant	Target is Participant	Send Timestamps	Participant Type
No Send Times	Y/N	N	anchor
Active Target	N	Y	anchor/mobile
Low Traffic Mobile	Y	Y	anchor/mobile

Table 5.1: TDOA Variants

TOA information can be used with anchor nodes for trilateration, or with mobile nodes to solve for the relative location of all nodes. The latter situation is known as the sensor network localization problem, and is described in Appendix C.

5.4 PinPoint TDOA

The three TDOA variants are all used to generate the same distance information for hyperbolic location The TDOA active target variant is used for nonparticipant target nodes. If the target node were a participant, TOA is the most accurate location method, as we show in Chapter 6. The TDOA no-send-times variant is used if participants do not have a send timestamp capability, which may be true of some hardware platforms. This variant may be used for both participant and nonparticipant target nodes. The TDOA low-traffic variant can be used to reduce the volume of network messages for a participant target. The three TDOA variants are summarized in Table 5.1.

5.4.1 TDOA Active Target

The active target variant locates an actively transmitting stranger node q. In order to locate a stranger node, we assume it can be detected and its MAC address discovered. Techniques for discovering rogue nodes are described in Chapter 7.4.1. With the stranger node MAC address identified, we show how to locate it using TDOA.

Within the active target basic message set in Figure 5.6, there are three messages sent at times t^a, t^b, t^q. The participant a sends a message at time t^a. The participant

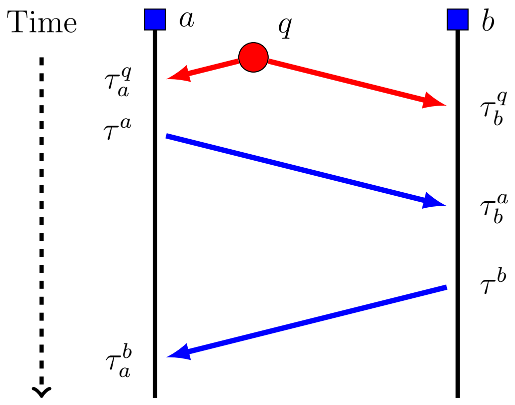


Figure 5.6: TDOA Active Target Basic Message Set

b sends a message at time t^b. The target q sends a message at time t^q. The messages between participants provide the information to compute the offset $\theta_{a\to b}$, which is used to find the time difference between the receive times $\tau_b^q - \tau_a^q$.

Using the clock model, the equations for the timestamps with biases in local time are as follows:

$$\tau^a = \beta_a(t^a + \alpha_a) + s_a + e^a \tag{5.6}$$

$$\tau_b^a = \beta_b(t^a + \alpha_b + d(a,b)) + r_b + e_b^a \tag{5.7}$$

$$\tau^b = \beta_b(t^b + \alpha_b) + s_b + e^b \tag{5.8}$$

$$\tau_a^b = \beta_a(t^b + \alpha_a + d(a,b)) + r_a + e_a^b \tag{5.9}$$

$$\tau_a^q = \beta_a(t^q + \alpha_a + d(a,q)) + r_a + e_a^q \tag{5.10}$$

$$\tau_b^q = \beta_b(t^q + \alpha_b + d(b,q)) + r_b + e_b^q \tag{5.11}$$

We solve the reference equations for $d(b,q) - d(a,q)$:

$$\begin{aligned}\beta_*\left[d(b,q) - d(a,q)\right] = \frac{\beta_*}{\beta_b}&\left(\tau_b^q - \frac{1}{2}(\tau^b + \tau_b^a) - \frac{1}{2}(r_b - s_b)\right) \\ - \frac{\beta_*}{\beta_a}&\left(\tau_a^q - \frac{1}{2}(\tau^a + \tau_a^b) - \frac{1}{2}(r_a - s_a)\right)\end{aligned} \tag{5.12}$$

As with TOA, $\beta_* \approx 1$, so $\beta_*\left[d(b,q) - d(a,q)\right] \approx d(b,q) - d(a,q)$.

The TDOA result of Equation 5.12 is identical to the solution that would be found using consensus clock synchronization for nodes a and b and subtracting their receive timestamps τ_a^q and τ_b^q. The key step is the application of Theorem A.1. Further substitution for $\theta_{a\to b}$ using Equation 4.13 yields Equation 5.12.

$$\begin{aligned}\tau_b^q - \tau_a^q &= \left(\frac{\beta_*}{\beta_b}\tau_b^q + \theta_{b\to *}\right) - \left(\frac{\beta_*}{\beta_a}\tau_a^q + \theta_{a\to *}\right)\\ &= \left(\frac{\beta_*}{\beta_b}\tau_b^q\right) - \left(\frac{\beta_*}{\beta_a}\tau_a^q\right) - \theta_{a\to b}\end{aligned}$$

5.4.2 No Send Times

Support for timestamping of outgoing messages is not common among 802.11 drivers. For this reason, we introduce another TDOA variant for participant nodes without outgoing timestamp support. This variant requires prior knowledge of the propagation delays between nodes. These propagation delays must be known regardless of potential changes in the environment or multipath effects. An advantage of this approach is all timestamps are subject to the same receive bias, which will cancel out.

For this TDOA variant, there are four nodes involved in the basic message set. Nodes a and b are anchor nodes. Node l can be either a landmark node or an anchor node. Node q is a stranger node or mobile node. The approach is to compute the clock offset $\theta_{a\to b}$ using messages from l and the known locations of a, b, and l; then use this offset to compute TDOA for messages from q.

$$\tau_a^l = \beta_a(t^l + \alpha_a + d(a,l)) + r_a \tag{5.13}$$

$$\tau_b^l = \beta_b(t^l + \alpha_b + d(b,l)) + r_b \tag{5.14}$$

$$\tau_a^q = \beta_a(t^q + \alpha_a + d(a,q)) + r_a \tag{5.15}$$

$$\tau_b^q = \beta_b(t^q + \alpha_b + d(b,q)) + r_b \tag{5.16}$$

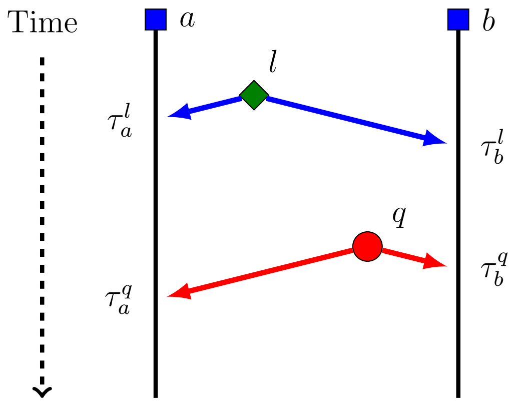


Figure 5.7: TDOA without Send Times Basic Message Set

The equation for TDOA without send times is given by Equation 5.17. The card bias values cancel in this solution.

$$\beta_* \left[d(b,q) - d(a,q)\right] = \beta_* \left[d(b,l) - d(a,l)\right] + \frac{\beta_*}{\beta_b}\left(\tau_b^q - \tau_b^l\right) - \frac{\beta_*}{\beta_a}\left(\tau_a^q - \tau_a^l\right) \qquad (5.17)$$

Again, $\beta_* \approx 1$, and thus $\beta_* \left[d(b,q) - d(a,q)\right] \approx d(b,q) - d(a,q)$, and the input $\beta_* \left[d(b,l) - d(a,l)\right] \approx d(b,l) - d(a,l)$.

5.4.3 Low Traffic Mobile Node

For the TOA and TDOA active target variants, each node must send an equal number of packets to maintain statistical independence of distance measurements. In the low traffic variant of TDOA, the target mobile node can minimize the number of packets sent. No packets sent from the target node are necessary for the basic message set. The target node does, however, need to send packets to participate in estimating $\frac{\beta_*}{\beta_m}$.

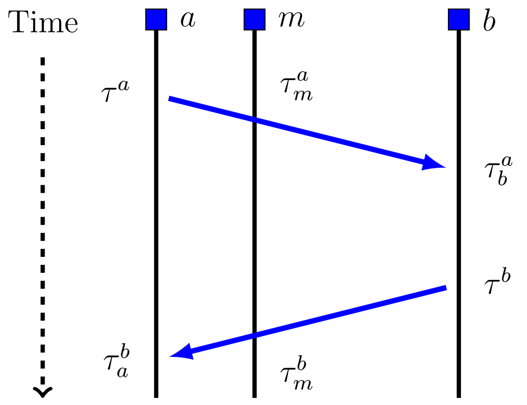


Figure 5.8: TDOA Low Traffic Mobile Node Basic Message Set

$$\tau^a = \beta_a(t^a + \alpha_a) + s_a \tag{5.18}$$

$$\tau_m^a = \beta_m(t^a + \alpha_m + d(a,m)) + r_m \tag{5.19}$$

$$\tau_b^a = \beta_b(t^b + \alpha_b + d(a,b)) + r_b \tag{5.20}$$

$$\tau^b = \beta_b(t^b + \alpha_b) + s_b \tag{5.21}$$

$$\tau_m^b = \beta_m(t^b + \alpha_m + d(b,m)) + r_m \tag{5.22}$$

$$\tau_a^b = \beta_a(t^a + \alpha_a + d(a,b)) + r_a \tag{5.23}$$

The target mobile node listens to messages between the tracking nodes and combines these timestamps with those between the tracking nodes. The target node compares the time lapsed on its own clock to that elapsed between the two tracking nodes.

$$\begin{aligned}\beta_* \left[d(b,m) - d(a,m)\right] =& \frac{\beta_*}{\beta_m}\left(\tau_m^b - \tau_m^a\right) \\ &- \frac{1}{2}\frac{\beta_*}{\beta_b}\left(\left(\tau^b - \tau_b^a\right) - (s_b - r_b)\right) \\ &+ \frac{1}{2}\frac{\beta_*}{\beta_a}\left(\left(\tau^a - \tau_a^b\right) - (s_a - r_a)\right)\end{aligned} \tag{5.24}$$

Again, $\beta_* \approx 1$, and thus $\beta_* \left[d(b,m) - d(a,m)\right] \approx d(b,m) - d(a,m)$.

Beacon Interval (k ticks)	Clock Speed (MHz)	Sample Size	$r-s$ clock ticks 5213A		2414
100	20	48000	22422.65	22422.58	-54377.87
500	20	11000	22423.20	22423.03	-361577.38
1000	20	6000	22423.02	22422.67	-745577.91
1000	40	8000	45230.69	45230.41	-722771.39

Table 5.2: Atheros card bias for 5213A and 2414 models

5.5 Card Biases

The card bias values are necessary corrections for proper computation of distance in PinPoint variants. Without the inclusion of card bias values, the distances reported by PinPoint TOA can be negative by tens of thousands of clock ticks when two nodes are adjacent, an obviously incorrect result.

In the distance computations for all PinPoint variants, each individual card's send and receive biases always appear together, making the difference $(r-s)$ sufficient to characterize the bias values. The correction is not a simple offset because it is dependent on clock drift values, which have been observed to change over time.

Solving for the card bias values requires at least three nodes. We can solve for $r-s$ by adding and subtracting the right sides of equations 5.5 and 5.12 when a and b are colocated $(d(b,q)-d(a,q)=d(a,b)=0)$. The complete derivation is found in Appendix A.5.

$$r_a - s_a = (\tau_a^q - \tau^a) + \frac{\beta_a}{\beta_b}(\tau_b^a - \tau_b^q) \tag{5.25}$$

$$r_b - s_b = (\tau_b^q - \tau^b) + \frac{\beta_b}{\beta_a}(\tau_a^b - \tau_a^q) \tag{5.26}$$

Atheros card bias varies with card model, the particular card, beacon interval and clock speed. Error from neglecting $r-s$ is $(\frac{\beta_b}{\beta_a}-1)*(r-s)$. For $(\frac{\beta_b}{\beta_a}-1)=10*10^{-6}$ and $r-s=45000$, this is 0.45 clock ticks, or roughly 4 m at 40 MHz clock speed.

5.6 Error Estimation

In this section, we examine the effect of timestamp measurement error of τ on TOA and TDOA. In performing this analysis, we make the following assumptions:

1. Packets are sent between the tracking nodes at intervals no larger than P.

2. The receive and send time errors e are independent with mean 0 and standard deviation $\sigma < 10$ clock ticks $\ll P$.

3. Clock drift is stable over time periods of tens of seconds about some value $\frac{\beta'_*}{\beta'_b}$.

$$\left| \frac{\beta_*(t)}{\beta_b(t)} - \frac{\beta'_*}{\beta'_b} \right| \leq \epsilon \tag{5.27}$$

   The relative drift rate may fluctuate within this range in any manner, but the fluctuations are bounded.

4. The clock drift estimate $\frac{\beta_b}{\beta_a}$ using methods from Chapter 3.4.2 is accurate.

$$e_\beta = \left| \frac{\beta'_b}{\beta'_a} - \frac{\beta_b}{\beta_a} \right| < \epsilon \tag{5.28}$$

We start by analyzing the error from timestamps for one node for the active target TDOA variant. This analysis generalizes easily to both timestamps for the other node as well as to other PinPoint variants.

$$\frac{\beta_*}{\beta_b}\left(\tau^q_b - \frac{1}{2}(\tau^b + \tau^a_b) - \frac{1}{2}(r_b - s_b)\right)$$

We can simply factor out the random variable terms e associated with the timestamps. The clock drift term $\frac{\beta_*}{\beta_b}$ is negligible for the timestamps errors because $\frac{\beta_*}{\beta_b} = 1 + \delta$ with $\delta < 10^{-4}$ and $\sigma < 10$, so $\delta\sigma < 10^{-3}$ clock ticks, which is negligible for 25 ns clock ticks and expected meter level accuracy.

$$\frac{\beta_*}{\beta_b}\left(e_b^q + \frac{1}{2}(e^b + e_b^a)\right) \approx e_b^q + \frac{1}{2}(e^b + e_b^a)$$

By assumption these terms are independent and the variance of the error is the sum of the variances of the timestamp errors, so the standard deviation is

$$\left(\sqrt{1^2 + \left(\frac{1}{2}\right)^2 + \left(\frac{1}{2}\right)^2}\right)\sigma = \left(\sqrt{\frac{3}{2}}\right)\sigma$$

Based on our assumptions, we can bound the difference between our estimated clock drift $\frac{\beta_*}{\beta_b}$ and the actual clock drift value $\frac{\beta_*(t)}{\beta_b(t)}$ for any basic set.

$$\left|\frac{\beta_*}{\beta_b} - \frac{\beta_*(t)}{\beta_b(t)}\right| \leq \left|\frac{\beta_b'}{\beta_a'} - \frac{\beta_b}{\beta_a}\right| + \left|\frac{\beta_b}{\beta_a} - \frac{\beta_b(t)}{\beta_a(t)}\right| \leq 2\epsilon$$

Assuming packets are sent at a regular interval P from the anchor nodes, the average of times τ_a^b and τ^a will be within one interval of τ_a^q.

$$\left|\tau_a^q - \frac{1}{2}(\tau_a^b + \tau^a)\right| < P$$

The absolute error due to clock drift error will therefore be

$$\begin{aligned}\left|\frac{\beta_*}{\beta_b} - \frac{\beta_*(t)}{\beta_b(t)}\right|\left(\tau_b^q - \frac{1}{2}(\tau^b + \tau_b^a) - \frac{1}{2}(r_b - s_b)\right) &\leq 2\epsilon(P + \frac{1}{2}|r_b - s_b|) \\ &\leq \epsilon(2P + |r_b - s_b|)\end{aligned}$$

To complete the analysis for the active target TDOA variant, we consider the symmetric terms for node a. The standard deviation will also be $\sqrt{\frac{3}{2}}\sigma$, with an additional error from clock drift bounded by $\epsilon(2P + |r_a - s_a|)$. The final error for the active target variant therefore has standard deviation $\sqrt{3}\sigma$, with additional error

Algorithm	Std Dev	Bias Bound
TOA	σ	$\epsilon[2P + \lvert r_a - s_a\rvert + \lvert r_b - s_b\rvert]$
TDOA Active Target	$\sqrt{3}\sigma$	$\epsilon[4P + \lvert r_a - s_a\rvert + \lvert r_b - s_b\rvert]$
TDOA No Send Times	2σ	$4\epsilon P$
TDOA Low Traffic	$\sqrt{3}\sigma$	$\epsilon[4P + \lvert r_a - s_a\rvert + \lvert r_b - s_b\rvert]$

Table 5.3: Expected Error for PinPoint TOA and TDOA Variants

bounded by $\epsilon(4P + |r_a - s_a| + |r_b - s_b|)$.

For the 802.11 based system, we now numerically estimate the PinPoint TDOA error. If we are able to measure clock drift within 0.1 ppm, $\epsilon \leq 1 \cdot 10^{-7}$. For the madwifi driver, the maximum beacon interval is $P = 10^6$ clock ticks. Assuming $\sigma = 1.6$ clock ticks and $r_a - s_a = 45000$, the total error has standard deviation 2.77, with bias less than 0.41 clock ticks. To reduce the bias, packets can be sent more frequently to lower the beacon interval.

The analysis for other PinPoint variant node timestamps is similar, using the above assumptions to estimate the standard deviation of distance error related to σ and a bounded error bias based on the clock drift error ϵ. This is summarized in Table 5.3.

The error analysis has important impacts on the the design of a PinPoint system. The message interval for tracking nodes affects the accuracy of a single distance sample. If fine scheduling control for messages is available, PinPoint messages should be scheduled into short intervals. This is the advantage of using query-response. Without fine control, which is the case for 802.11 networks, the message interval can be shortened by increasing the rate of PinPoint messages sent from each node. When changing the PinPoint message rate, real-time accuracy and responsiveness are also affected.

Considering the standard deviation of the distance measure, the PinPoint TOA variant is the most accurate, with the standard deviation of the distance measure equal to the standard deviation of the timestamp error. This is before considering

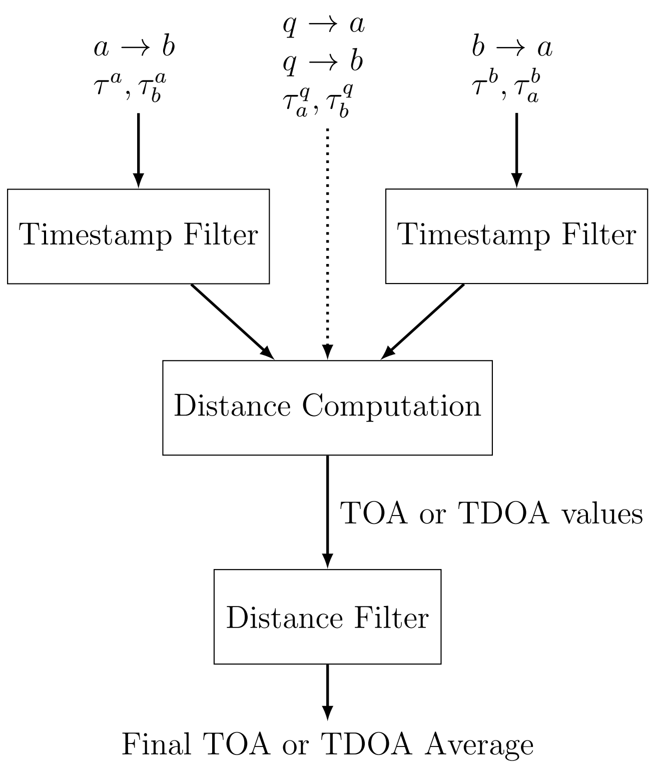


Figure 5.9: Timestamp and distance filtering

how TOA information is different from TDOA information, which we consider in Chapter 6. The active target and low traffic TDOA variants are equally accurate. The TDOA variant with no send times is the least accurate.

5.7 Outlier Filtering

In the empirical measurements we observed many readings which are well beyond normally expected behavior — outliers. Clearly, outliers must be eliminated to accurately estimate distances.

There are two filters working in series to eliminate outliers. The first filter operates on timestamps between participants. Only timestamps passing through this filter are used for clock drift and distance computations. The second filter operates on the derived TOA and TDOA distance values. This is summarized in Figure 5.9.

5.7.1 Timestamp Stream Filtering

By filtering, we eliminate timestamp outliers that are inconsistent with known physical behavior and expected error. Filtering is performed on a stream of incoming messages between the participant sender a and participant receiver b. Both nodes must be participants for the precision timestamps to be available. Each sender-receiver pair is handled separately. To perform filtering we assume the following timestamp behaviors within our system.

1. The rate of change of the ratio $\frac{\beta_b}{\beta_a}$ can be bounded by M, for example, 1 ppm per second.
2. The rate of change in distance between nodes is bounded by human moving speeds $< 10m/s$.
3. The vast majority of errors e will be less than a bound E, $|e| < E$ with high probability.

The criterion for accepting message timestamps is based on the point clock drift estimate $\widehat{\left[\frac{\beta_b}{\beta_a}\right]}(i)$ for these timestamps and the current clock drift exponential filter . If the difference between the clock drift estimates is consistent with timestamp errors and potential clock drift given the time $T = \tau^a(i+k) - \tau^a(i)$, we accept the new timestamps. Otherwise we discard these timestamps and proceed to the next ones.

$$\left| \frac{\beta_b}{\beta_a}(i-1) - \widehat{\left[\frac{\beta_b}{\beta_a}\right]}(i) \right| \leq \frac{E}{T} + MT \tag{5.29}$$

As we saw in Equation 3.21 in Chapter 3.4.2, the clock drift estimation error is roughly proportional to $\frac{\sigma}{T}$. For 44 MHz clock ticks (22.7 ns), the clock drift error due to human movement is small $\frac{\Delta d}{T} < 2.27 * 10^{-7}$ and will be negligible relative to $\frac{E}{T} \approx 2.27 * 10^{-5}$ for $E = 20$ and 50 packets every second. Over short time periods T, the $\frac{\sigma}{T}$ error may be high, and it will be difficult to filter effectively. Over very long

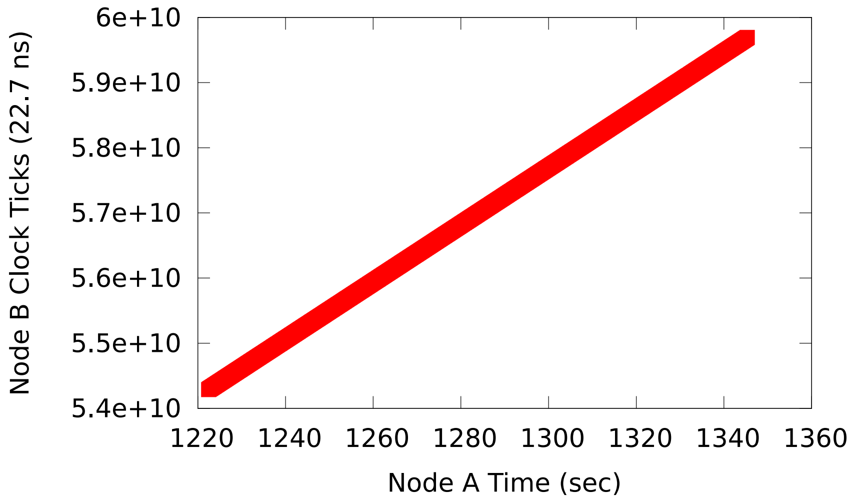


Figure 5.10: Raw Timestamps

time periods, the clock drift estimation error will be negligible, and changes in clock drift will be more significant than timestamp measurement error. In this case, the term MT causes the filter to accept timestamps.

Outliers may not be apparent from examining the raw timestamp values alone, which are shown in Figure 5.10. With the expectation of roughly linear behavior, we remove the linear regression line to see an outlier roughly 900,000 clock ticks greater than expected in Figure 5.11. By applying the timestamp stream filter with generous parameters $M = \frac{10^{-6}}{44*10^6 \text{ clock ticks}}$, $k = 10$, and $E = 20$ clock ticks, the outlier is removed, and the linear residuals for the remaining timestamps are shown in Figure 5.12.

If the first timestamps into the filter are themselves outliers, many good timestamps will be discarded because they are inconsistent with the initial outlier timestamps. Assuming relatively sparse outliers, we can reset the filter if too many timestamps are discarded in succession. To keep a system reset responding on reasonable time scales, the threshold for resets was set at 200 successive timestamps, or four seconds of data at 50 packets per second.

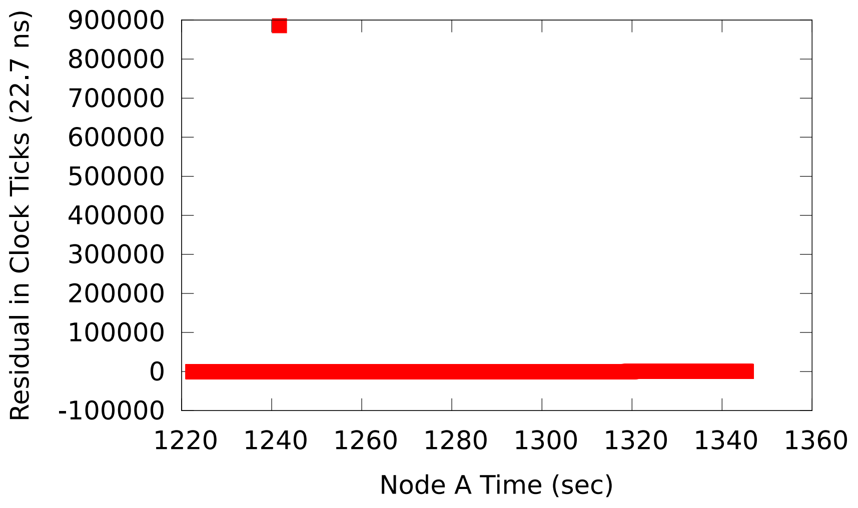


Figure 5.11: Unfiltered Residuals

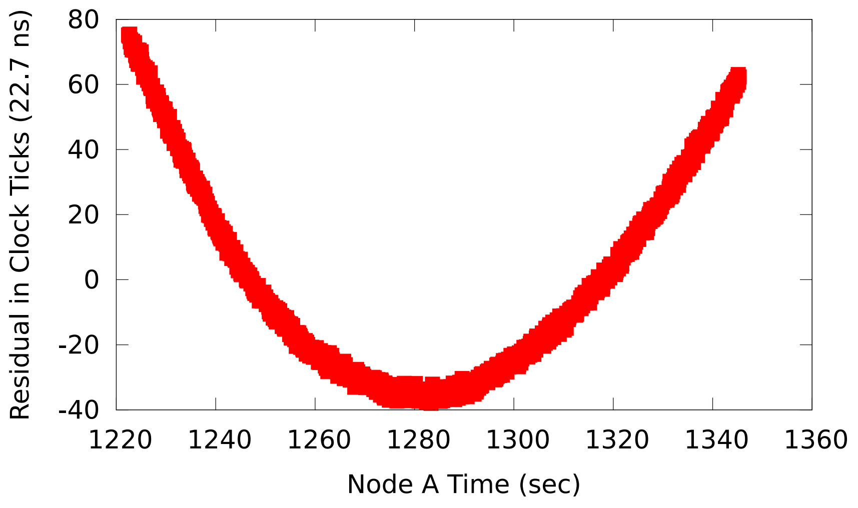


Figure 5.12: Filtered Residuals

5.7.2 Distance Filtering

TOA and TDOA distance results are filtered using a median-based approach. Averages are sensitive to outliers, The median can be computed using a partial sort, which runs in expected linear time. Any measurement outside a hard limit of 10 clock ticks from the median is considered an outlier. Following filtering, we use a simple average to compute the times for TOA and TDOA. Using the speed of light, this time average translates to a distance and determines a sphere for solving the trilateration problem or a hyperboloid for use in solving the hyperbolic location problem.

5.8 TDOA Distance Measurement Experimental Results

This experiment was designed solely to measure distances using TDOA in one dimension and does not require an optimization step to solve for location. The three nodes were arranged in line to make the measured TDOA value equivalent to the distance between two of the nodes. This is the simplest experimental verification of the PinPoint TDOA techniques because it involves only distance measurements.

5.8.1 Experiment Setup

Netgear WG511T and Netgear WG511U PC cards with Atheros chipsets were used for this experiment. The cards were used with laptops running GNU/Linux with the 2.6.22 kernel and the madwifi-ng driver [4]. The wireless card reference clocks were set to run at 40 MHz rather than the standard 1 MHz. The beacon interval values were left at the default of 100 nominal ms. Instead of 10 beacons per second, there were 400 beacons per second. These changes are summarized in Table 5.4.

Three laptops, each with a single wireless PC Card, were used to verify accurate

Clock speed (MHz)	1	40
Clock tick time	1 μs	25 ns
Beacon interval	100 ms	2.5 ms
Beacons / sec	10	400

Table 5.4: Clock characteristics with increased reference clock speed

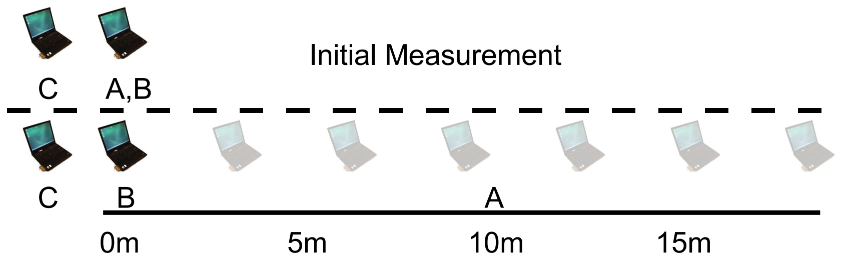


Figure 5.13: Distance Experiment Layout

time distance of arrival measurements. Laptop C operated as an AP, sending beacons only. Beacons sent by C were used to compute TDOA for A and B. Laptops A and B operated as both APs and monitors. The use of AP mode for A and B was for the the purpose of sending beacons. The madwifi-ng drivers allow operation of a single physical card in multiple modes using a mechanism called virtual access points (VAPs). All laptops transmitted and received on the same channel.

Laptops B and C were left stationary at separate locations. A was moved to seven separate locations, all collinear with B and C. In the linear configuration, the difference of distances is $d(B, C) - d(A, C) = -d(A, B)$. At each location of A, the timestamps for all received beacons were recorded by B for both A and C and by C for A and B. Measurements were performed for 60-100 seconds at each location.

5.8.2 Results

The initial measurement with A and B at distance zero was used to determine the card delay quantities. The card delay quantities are given in Table 5.5. Note that the card delay values are significant. With the measured value $\frac{\beta_b}{\beta_a} \approx 3.2{*}10^{-6}$; $\frac{\beta_*}{\beta_b} = -1.6{*}10^{-6}$,

Card	MAC	$r - s$
A	06:0f:b5:28:2c:65	45229.71
B	06:0f:b5:10:91:5c	45229.44

Table 5.5: Card Delay Values for Distance Measurement

Distance (m)	Δt Clock Ticks (25 ns)	Estimated (m)	Error (m)
0.00	0	0	N/A
3.05	0.16	1.19	-1.86
6.10	0.84	6.33	0.23
9.14	0.68	5.14	-4.01
12.19	1.45	10.84	-1.35
15.24	1.36	10.22	-5.02
18.29	2.56	19.18	0.89

Table 5.6: Distance Results

$\frac{\beta_*}{\beta_a} = 1.6*10^{-6}$, and the correction for the card delay is $\frac{1}{2}\left(\frac{\beta_*}{\beta_b}(r_b - s_b) - \frac{\beta_*}{\beta_a}(r_a - s_a)\right) \approx$ 0.21 clock ticks. At 25 ns per clock tick, this is roughly 1.57m.

The TDOA computation gives measurements within 5m. The cause of these measurement errors is suspected to be multipath effects.

The histogram of TDOA values for the 15.24m case is shown in Figure 5.14. The distribution is unimodal with a mean of 1.370 clock ticks and a median of 1.373 clock ticks. The standard deviation of the distribution is 2.83 clock ticks. The TDOA histogram does not provide an obvious explanation for the distance underestimation.

5.9 Active Transmitter TDOA Experimental Results

This experiment demonstrates that an 802.11 transmitter may be located reliably when in line-of-sight, with an accuracy of less than 3.5m. The end-to-end accuracy is determined by comparing the distance between actual locations of the target node to those estimated by the PinPoint TDOA system. PinPoint estimates location by

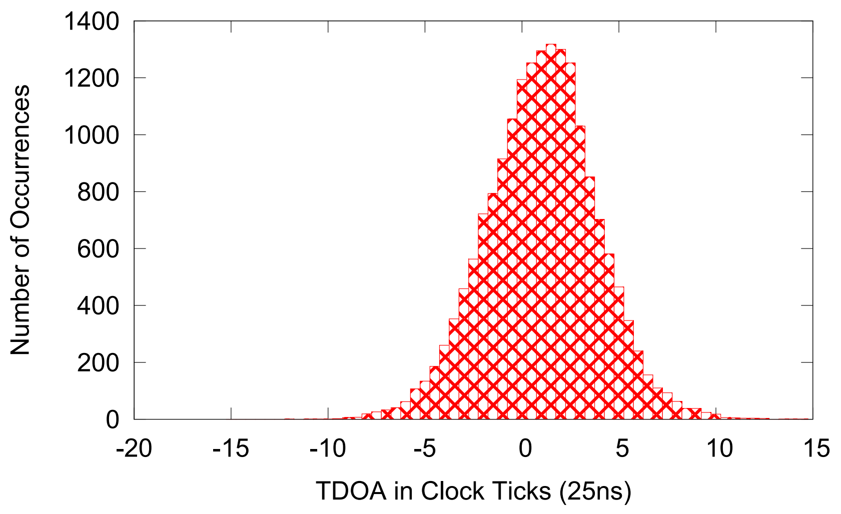


Figure 5.14: TDOA histogram for TDOA = 15.24m

measuring the three pairwise TDOA values for three anchor nodes and solving the hyperbolic location problem by gradient descent as described in Chapter 2.3.2.

5.9.1 Experiment Setup

The test equipment consisted of 2 anchor laptops with Ubiquiti Superrange Cardbus network cards, 1 anchor laptop with a Netgear WG511T network card, and 1 target laptop with a Netgear WG511U network card. The display laptop and each anchor laptop were connected to the router via Ethernet.

Each anchor card ran at 40 MHz reference clock speed with beacon interval of 1000, which is nominally in milliseconds, but at the increased clock speed translates to a real beacon interval of $\frac{1000\text{ms}}{40} = 25\text{ms}$, or 40 beacons per second. The 40 MHz value is the maximum clock speed available for the Atheros based cards.

After an initial measurement with all anchor nodes at the same location to determine the card delay quantities, all three anchor nodes were placed in a single corridor. The locations of the anchor nodes are shown in Figure 5.15. The target laptop was moved to seven additional locations, also shown in Figure 5.15. The target laptop was left at each location for approximately 3 minutes. The laptop was oriented so

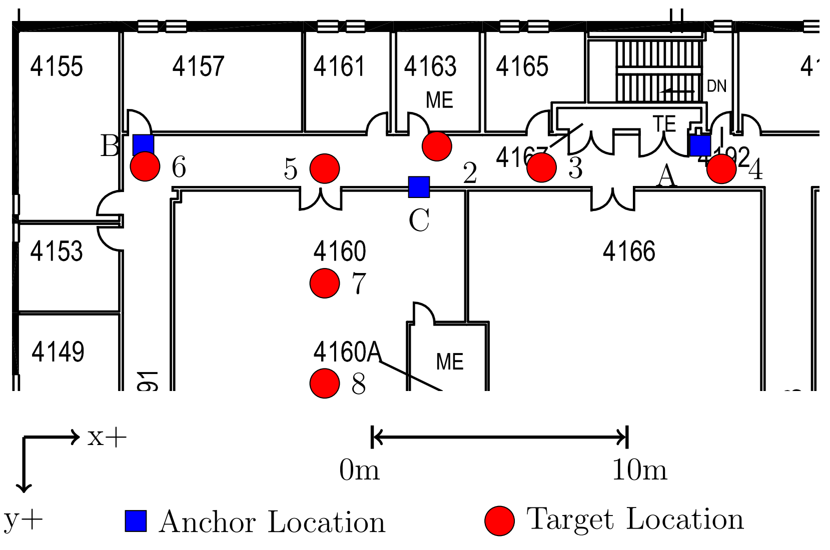


Figure 5.15: Anchor Node and Rogue AP Locations

Node	MAC	$r - s$	Chipset
A	06:15:6d:54:80:49	45230.35	5213A
B	06:15:6d:53:f9:61	45230.22	5213A
C	06:1e:2a:67:84:c9	-722771.34	2414

Table 5.7: Card Delay Values for 40 MHz, beacon interval 25 ms

that the wireless PC card had line of sight to all anchor nodes.

5.9.2 Results

The card delay quantities for this experiment's anchor nodes are given in Table 5.7. Nodes A and B used Ubiquiti network cards. Node C used the Netgear card.

Locations 2-8 in Table 5.8 show the actual and estimated target locations with location error. These locations are shown in Figure 5.16. Coordinates are in the Qt system, with the origin in the upper left corner. The x dimension increases moving to the right, and the y dimension increases moving down.

Figure 5.17 shows the hyperbolas and estimated location of the target for location 2. The anchor nodes are shown as squares. The estimated location is shown using concentric circles with inner radius 2.5m and outer radius 5m. In this example, the

	Actual		Estimated		Error
Location	x (m)	y (m)	x (m)	y (m)	(m)
1	Calibration				
2	24.76	9.10	26.92	10.96	2.85
3	28.50	9.86	29.29	11.39	1.72
4	34.93	9.91	37.63	9.42	2.74
5	20.75	9.90	19.07	11.30	2.19
6	14.33	9.82	11.02	9.42	3.33
7	20.75	14.00	19.07	11.30	3.18
8	20.75	17.58	21.06	10.98	6.61

Table 5.8: Rogue AP Location Error

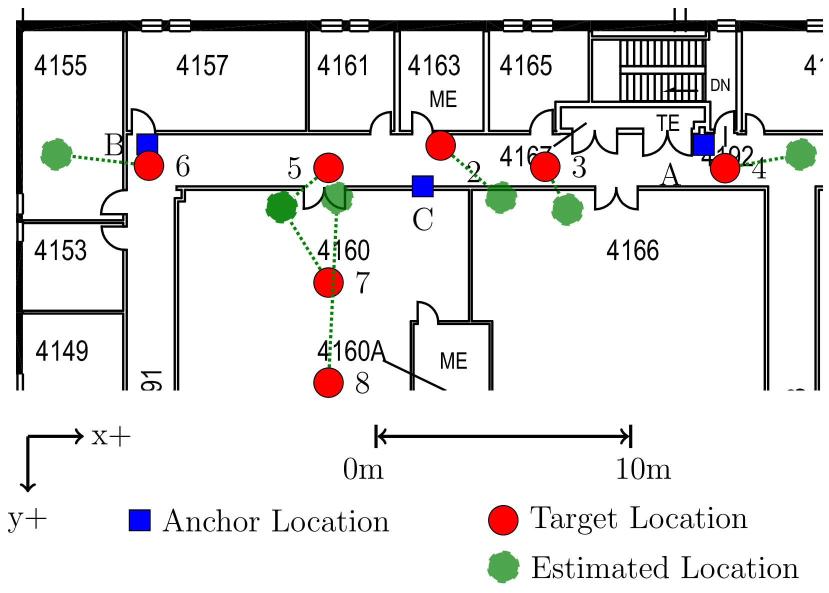


Figure 5.16: Estimated locations

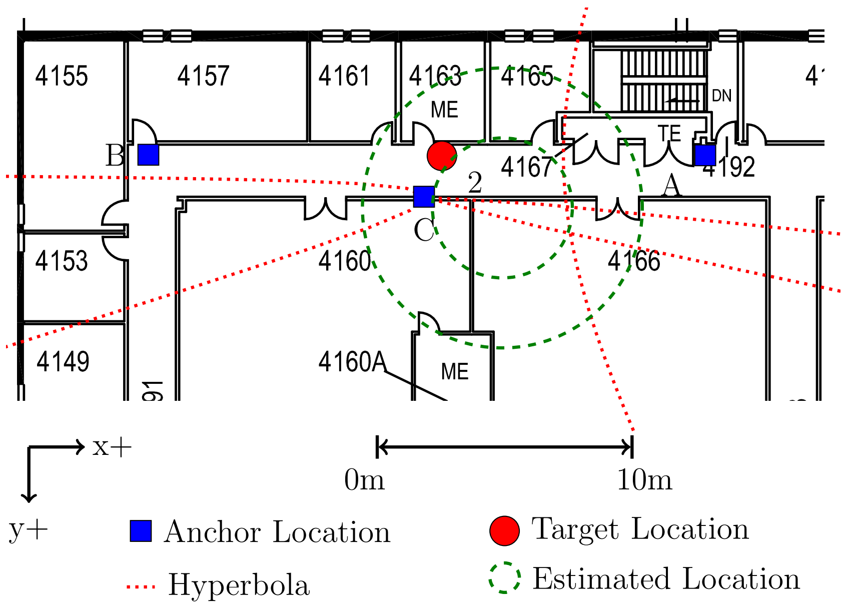


Figure 5.17: Optimization for Location 2

estimated location is 2.85m from the actual location.

The arrangement of the anchor nodes in the corridor has poor tracking geometry. This is a limitation of the indoor corridor environment, where nodes cannot be placed with equal angular distribution as described by Yang and Scheuing in [82]. In particular, a pair of anchor nodes cannot distinguish between two positions on the same hyperbola using TDOA. For the degenerate hyperbola case, the hyperbola is a line starting at one anchor that extends away from the other anchor. For locations 4 and 6, the initial guess strongly affects the estimated location. For these two locations, the center anchor node was used as the initial guess.

TDOA was effective in locating the target in line-of-sight situations, but not as effective for non-line-of-sight situations. Locations 2-6 are within line-of-sight of the anchor nodes. The error for line-of-sight arrangements was less than 3.5m. Locations 7 and 8 are non-line-of-sight. The error for one of the non-line-of-sight measurements was 3.18m, the other was 6.61m.

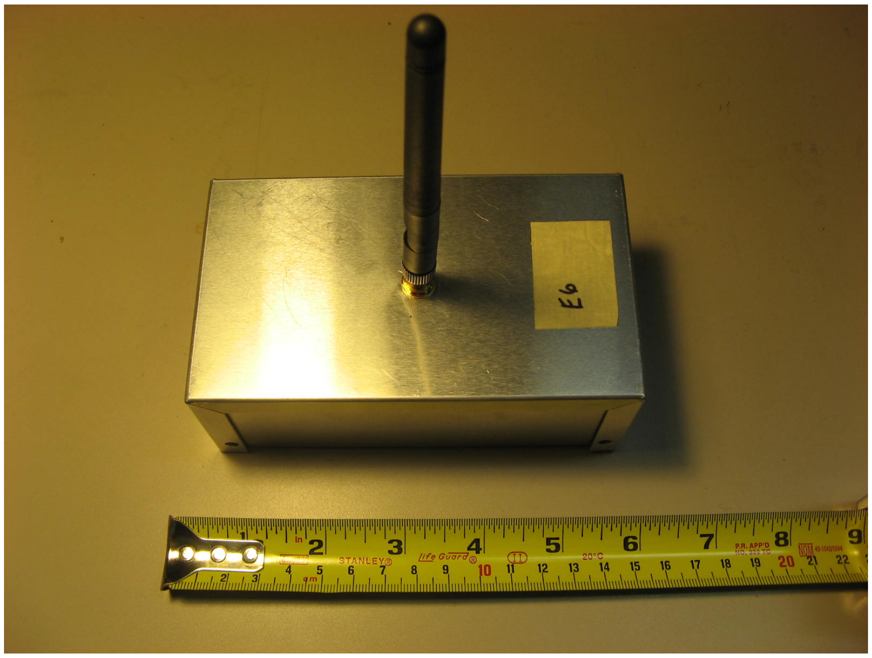


Figure 5.18: G2-based RN-134 in Aluminum Enclosure

5.10 PinPoint TOA with G2-based RN-134

The RN-134 is a small, mobile battery-powered device. This experiment measured the distance between two RN-134 nodes outdoors on a residential driveway. In this environment, multipath effects are expected to be minimized.

5.10.1 Experiment Setup

For this experiment the RN-134 tags were placed in aluminum boxes to shield them from RF interference. This strongly reduces the number of timestamps thrown out by the stream filter. Although this dissertation focuses on the converting timestamp information to distance information, RF design for the physical devices strongly impacts packet reception rates and timestamp accuracy and cannot be ignored.

The boxes were placed at locations 0m, 5m, 10m, 15m, and 20m apart. Each node sent 50 packets per second over two minutes at each location, collecting approximately 6000 distance samples. All packets were sent at the bit rate of 1 megabit per second

Actual Distance (m)	PinPoint Distance (clock ticks)	Measured Distance (m)	Standard Deviation (clock ticks)
0	73433.16	N/A	0.96
5	73433.76	4.06	0.97
10	73434.70	10.46	0.98
15	73435.19	13.82	0.98
20	73436.16	20.44	1.00

Table 5.9: PinPoint TOA results with RN-134

(Mbps) to maximize range.

Beyond 20m, the internode communication and timestamping accuracy degrades significantly, likely due to low signal strength values. The percentage of packet timestamps received and passing the stream filter can drop from $> 99\%$ to $< 10\%$. Communication between a node and the AP was more reliable, and at greater ranges. The node-to-node range is poor for 802.11; improvements to the signal processing capabilities, enclosure, and antenna attachments are expected to improve effective range.

The RN-134 uses a 44 MHz clock to timestamp; each clock tick is 22.7 ns. In one clock tick, radio signals will travel approximately 6.81m.

5.10.2 Results

Table 5.9 shows the PinPoint TOA distance results for this experiment. The distance results for all locations are all within 1.5 m of the actual distances.

With only two nodes, it is not known how to correct for timestamp bias with changing clock drift values because the techniques in Chapter 5.5 require three nodes. We can, however, assume constant clock drift and subtract the distance reading in clock ticks at 0m from all subsequent readings, then estimate the error due to changing clock drift. Assuming the clock drift is constant when recording at 0m, the reading will be $\frac{1}{2}\left[\frac{\beta_*}{\beta_a}(0)(r_a - s_a) + \frac{\beta_*}{\beta_b}(0)(r_b - s_b)\right]$. Any changes in clock drift values $\Delta\frac{\beta_*}{\beta_a}$ and

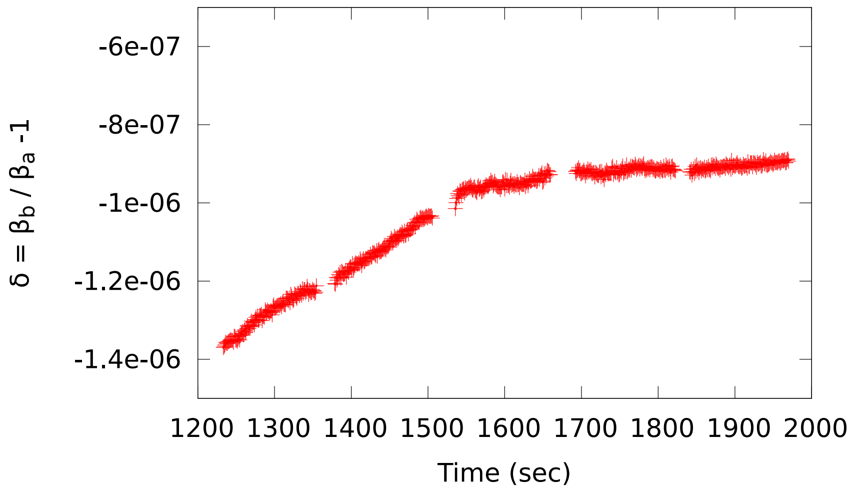


Figure 5.19: Clock drift over five locations

$\Delta\frac{\beta_*}{\beta_b}$ will result in error of

$$\frac{1}{2}\left[\Delta\frac{\beta_*}{\beta_a}(r_a - s_a) + \Delta\frac{\beta_*}{\beta_b}(r_b - s_b)\right]$$

For a numeric value, we examine Figure 5.19, which shows the clock drift over time between the two nodes for all five locations. The graph is discontinuous when the nodes were moved to new locations. Clock drift values are in the range of $[-1.4e^{-6}, 0.9e^{-6}]$, with a maximum difference in clock drift of $0.5e^{-6}$. The maximum expected difference in the bias term due to this change is approximately $0.5e^{-6} * 73433.16 \approx 0.037$ clock ticks, which makes the maximum expected distance error for failing to correct bias for the RN-134 approximately 0.25 m.

The histograms of distance measurements are very nearly normal. Figure 5.20 shows the unimodal histogram of distance measurements at 15m, with standard deviation of less than 1 clock tick. Figure 5.21 shows the corresponding qq-plot. The deviation of the curve from the straight line is a graphical representation of the deviation from the normal distribution. Considering the effects of discretization in generating the step levels, the distribution is nearly normal with short tails.

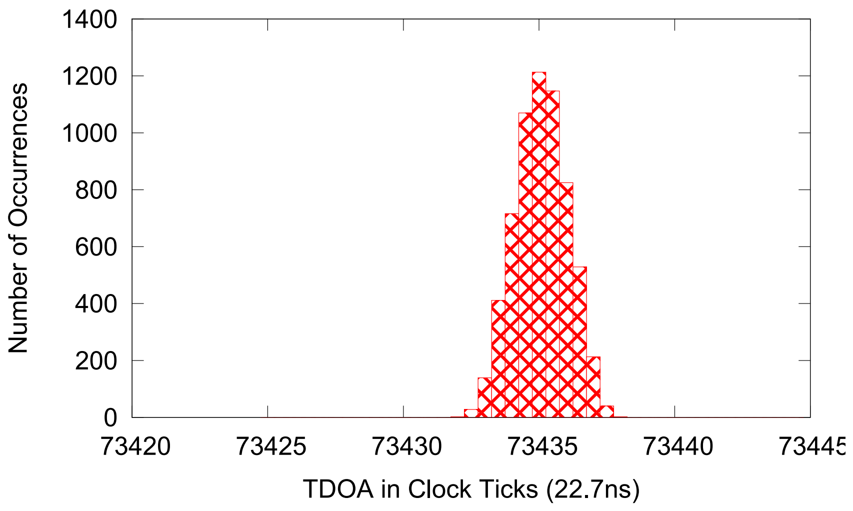


Figure 5.20: Distance histogram at 15m

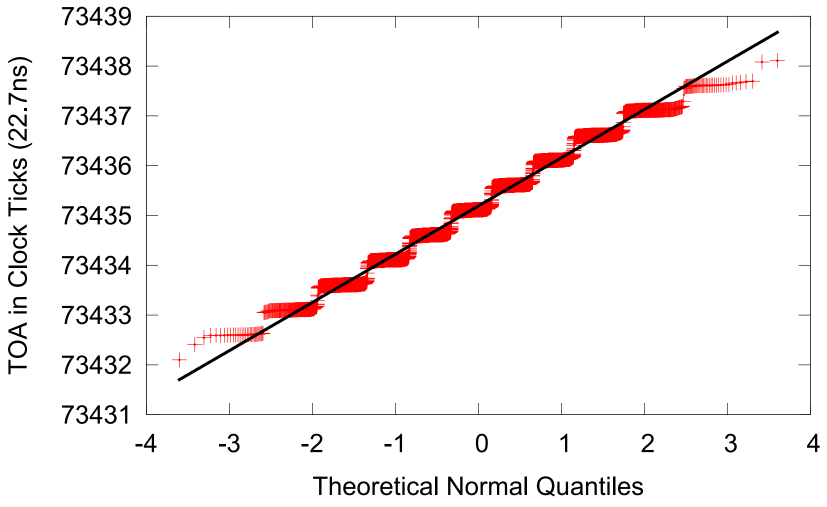


Figure 5.21: QQ plot for at 15m

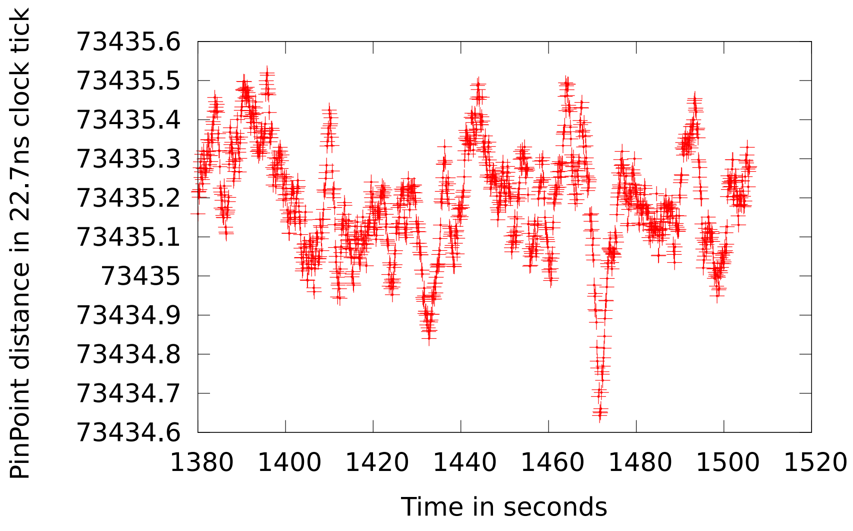


Figure 5.22: 100 sample averages for distance at 15m

The relatively small standard deviation of error allows real-time location. Figure 5.22 shows the distance measured over time using 100 sample windows, which for this experiment is two seconds of data. Figure 5.23 shows the histogram of the 100 sample averages. The average is within 1m of the correct 15m reading (73435.36 clock ticks) 43.4% of the time, within 2m 81.6% of the time, and within 3m 97.0% of the time. The size of the sample window is a tradeoff between accuracy and responsiveness. Larger sizes are more accurate, but less responsive. Packet rates can also be increased to improve responsiveness.

5.11 Summary

In this Chapter, we derived techniques for computing TOA and TDOA from timestamp measurements, correcting for consensus clock drift values. We demonstrated experimentally for multiple platforms that these techniques can compute distances with accuracy of a few meters. For TOA, we showed accuracy of better than 1.5m for simple distance measurements. For TDOA, we showed end-to-end location accuracy of 3.5m using simple hyperbolic location optimization based on gradient descent.

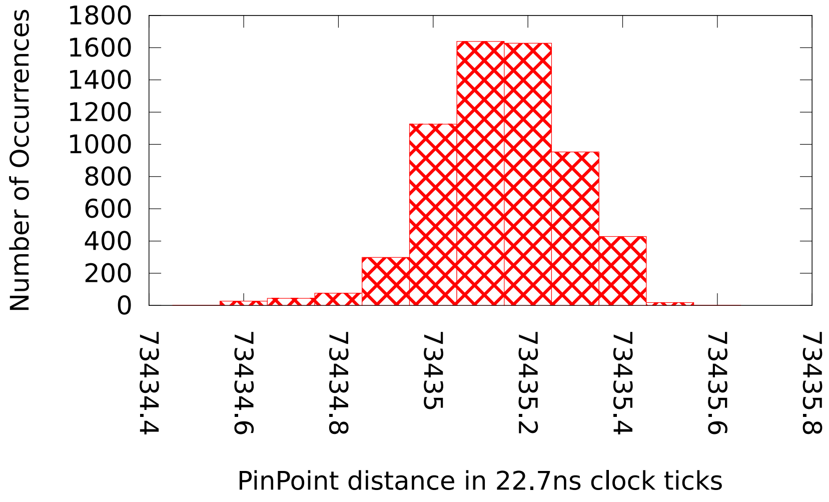


Figure 5.23: Histogram of 100 sample averages for distance at 15m

These results show that the conversion of timestamp information to distance information is sound. To realize a fully practical system, especially in environments with multipath effects, better control of RF considerations and signal processing is necessary. The timestamping capabilities provided by the Atheros and RN-134 hardware are not designed for location purposes. Better timestamping of the direct path communication was shown by Golden and Bateman and is planned for inclusion in 802.11v [38]. Known measures to improve general network performance in multipath environments such as additional bandwidth, spatial and antenna diversity including multiple-input, multiple-output (MIMO) designs [40], and frequency diversity should also improve PinPoint timestamping for location.

PinPoint accuracy may be increased by examining the assumption that timestamp errors are independent. Especially for more stable oscillators, the relation of clock drift values to timestamp discretization may be significant. In this case, correcting for the Vernier effect may increase PinPoint accuracy.

PinPoint is communication protocol agnostic. The data presented here are for 802.11 platforms, but the techniques apply to any broadcast communication protocol. PinPoint only requires timestamping capability for network messages.

Chapter 6

Hybrid TOA-TDOA Analysis

Given both TOA and TDOA distance information, we wish to find the optimal method of determining node locations. We examine the simplest version of the problem, where a set of anchor nodes locates a single node q. This case is a hybrid of the trilateration and hyperbolic location problems. Even this simple case provides insight on how to combine TOA and TDOA information more generally.

Prior work in this area has focused on the design of data fusion architecture for wireless location [49, 70]. Kleine-Ostmann, Bell, and Reza examined how to combine TOA and TDOA information, but assumed that these data sources are independent of one another. They evaluated fusion techniques with TOA and TDOA values assuming this independence by performing an analysis of variance for location estimates from separate sources. There is no known production system implementing a hybrid TOA and TDOA location system.

For PinPoint, TOA and TDOA measurements are available for any set of at least three broadcasting participant nodes. A hybrid TOA-TDOA system requires no more broadcast traffic than is used for either TOA or TDOA. The messages passed between nodes are used both for TOA as well as for TDOA. Since the TOA and TDOA measurement input timestamps overlap, the TOA and TDOA results are correlated.

As a first step in studying hybrid TDOA-TOA location, we examine how to optimally compute the difference $d(b,q) - d(a,q)$ for three nodes a, b, q. This is the distance primitive computed by TDOA. If $d(b,q) - d(a,q)$ is optimally computed without using TDOA, then TDOA provides no more information than TOA alone, and the optimal hybrid location system will simply be trilateration. If the optimal computation of $d(b,q) - d(a,q)$ includes TDOA, it is still unknown how best to combine TDOA and TOA information for computing location.

We can estimate the difference $d(b,q) - d(a,q)$ in distinct two ways, one using TDOA and one using TOA.

$$\mathrm{TDOA}(a,b,q) = d(b,q) - d(a,q) \tag{6.1}$$

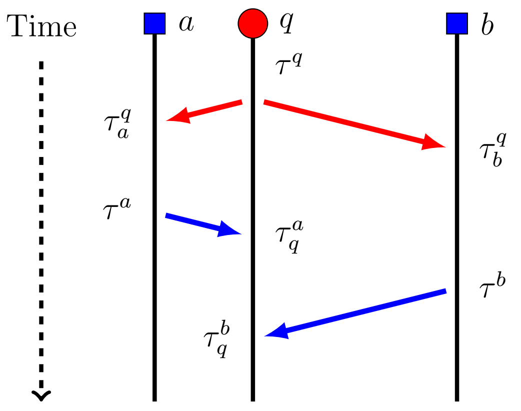
$$\mathrm{TOA}(b,q) - \mathrm{TOA}(a,q) = d(b,q) - d(a,q) \tag{6.2}$$

We can also use a weighted average of the two estimates for $k \in [0,1]$.

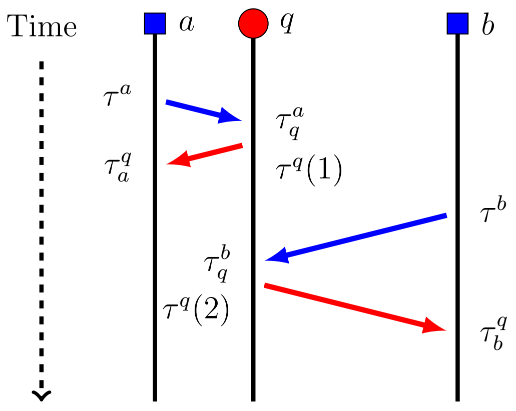
$$d(b,q) - d(a,q) = k * \mathrm{TDOA}(a,b,q) + (1-k) * (\mathrm{TOA}(b,q) - \mathrm{TOA}(a,q)) \tag{6.3}$$

We find the value of k that minimizes the standard deviation of the error for $d(b,q) - d(a,q)$.

We need to distinguish between two techniques for computing $d(b,q) - d(a,q)$ using TOA. TOA measures the quantities $d(a,q)$ and $d(b,q)$ separately. A single message τ^q from the target node q may be used to estimate both $d(a,q)$ and $d(b,q)$, or two timestamps $\tau^q(1)$ and $\tau^q(2)$ may be used to estimate the distances separately. We name these cases TOA with reuse and TOA without reuse respectively. The timestamps for the two techniques are shown in Figure 6.1. The method of computing $d(b,q) - d(a,q)$ using TOA affects the expected error.



(a) With reuse



(b) Without reuse

Figure 6.1: Two TOA techniques

6.1 PinPoint

Assuming send and receive errors are zero mean, identical and independently distributed, we quantify the standard deviation of the error for $d(b, q) - d(a, q)$ as measured by PinPoint. We first review the standard deviation of the error for TOA and TDOA individually. We then compute the standard deviation of the error for TOA with reuse and without reuse. Finally, we compute the optimal combination of TDOA with TOA with reuse or TOA without reuse.

The first-order error terms for TOA and TDOA are the timestamping errors. Using the standard deviation and bias bound analysis from Chapter 5.6, we are interested in the standard deviation term. For this analysis, we remove the bias bound by assuming we can estimate clock drift and the receive and send biases exactly. With this simplification, the major error terms are simply from the timestamping error. The simplified forms of TOA and TDOA are shown below.

$$\text{TOA}(a, b) = \frac{1}{2}\left(\left(\tau_a^b - \tau^a\right) - \left(\tau^b - \tau_b^a\right)\right) \tag{6.4}$$

$$\text{TDOA}(a, b, q) = \tau_b^q - \tau_a^q - \frac{1}{2}(\tau^b - \tau_a^b + \tau_b^a - \tau^a) \tag{6.5}$$

Table 6.1 shows how timestamping errors for single timestamps propagate to dis-

				TDOA	TOA w/ reuse	TOA w/o reuse
Error	$d(a,b)$	$d(a,q)$	$d(b,q)$	$d(b,q)-d(a,q)$	$d(b,q)-d(a,q)$	$d(b,q)-d(a,q)$
τ^a	-0.5	-0.5		0.5	0.5	0.5
τ^b	-0.5		-0.5	-0.5	-0.5	-0.5
τ^q		-0.5	-0.5			$\sqrt{0.5}$
τ^a_b	0.5			-0.5		
τ^a_q		0.5			-0.5	-0.5
τ^b_a	0.5			0.5		
τ^b_q			0.5		0.5	0.5
τ^q_a		0.5		-1	-0.5	-0.5
τ^q_b			0.5	1	0.5	0.5
Total	1	1	1	$\sqrt{3}$	$\sqrt{1.5}$	$\sqrt{2}$

Table 6.1: Error Analysis of $d(b,q)-d(a,q)$ for TDOA and TOA

tance computations. The standard deviation of all timestamp errors is assumed to be σ. All table entries are standard deviations in multiples of σ, with signs retained to determine when terms cancel. The key difference between TOA with and without reuse is the timestamps for τ^q. For TOA with reuse, the τ^q term from $d(b,q)$ cancels the identical τ^q term from $d(a,q)$; any error present in τ^q is the same in both places. For TOA without reuse, the $\tau^q(1)$ term has error independent of $\tau^q(2)$. The variance of these errors adds, giving standard deviation of $\sqrt{0.5^2+0.5^2}=\sqrt{0.5}$ for the sum of τ^q values.

We now find the optimal combinations of TOA with TDOA. Table 6.2 shows the computation of standard deviation of error in terms of the parameter k for the hybrid computation of $d(b,q)-d(a,q)$. These quantities are computed with the assumption that the timestamps used for TOA are the same used for TDOA.

To achieve the best estimate for each hybrid combination, we minimize the standard deviation of the error subject to the constraint $k \in [0,1]$. Table 6.3 shows the optimal k values and the associated error. For TOA with reuse, where we reuse τ^q, optimally $k=0$, meaning the value measured by TDOA is ignored. For TOA without reuse, where separate $\tau^q(1)$ and $\tau^q(2)$ values are used, optimally $k=0.25$. In this

	kTDOA + $(1-k)$TOA	
	TOA w/ reuse	TOA w/o reuse
τ^a	0.5	0.5
τ^b	-0.5	-0.5
τ^q	0	$\sqrt{0.5}(1-k)$
τ^a_b	$-0.5k$	$-0.5k$
τ^a_q	$-0.5(1-k)$	$-0.5(1-k)$
τ^b_a	$0.5k$	$0.5k$
τ^b_q	$0.5(1-k)$	$0.5(1-k)$
τ^q_a	$-0.5(k+1)$	$-0.5(k+1)$
τ^q_b	$0.5(k+1)$	$0.5(k+1)$
Total	$\sqrt{1.5+1.5k^2}$	$\sqrt{2-k+2k^2}$

Table 6.2: Error Analysis of $d(b,q)-d(a,q)$ for Hybrid TDOA/TOA

Method	k	Error
Hybrid using TOA w/ reuse	0	$\sqrt{1.5}$
Hybrid using TOA w/o reuse	0.25	$\sqrt{1.875}$

Table 6.3: Optimal k

case, the TDOA measurement supplies more information than TOA alone. The utility of TDOA information therefore depends on how TOA information is computed.

Both TOA methods of estimating $d(b,q)-d(a,q)$ have smaller standard deviation than TDOA. Since this is the most primitive information available from TDOA, and TOA computes it with smaller standard deviation, TOA information is better than TDOA information. For the TOA with reuse, TOA is strictly better than TDOA information, which does not improve the estimate. In the case of TOA without reuse, TDOA information improves the estimate, and a hybrid TOA-TDOA location system may improve location accuracy over a pure TOA system.

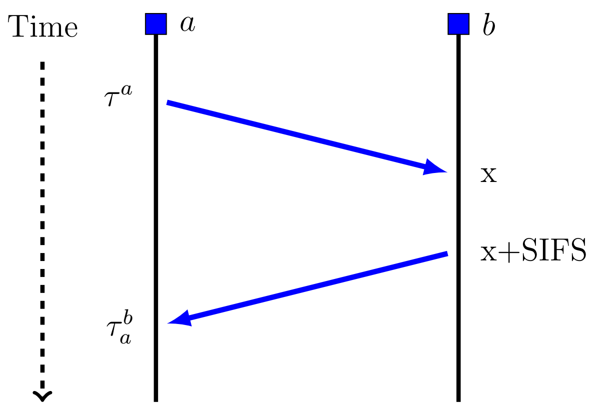


Figure 6.2: Goodtry SIFS-based distances

6.2 Goodtry

The Goodtry system [42] measures $d(a, b)$ by exploiting the fixed 802.11 short inter-frame spacing (SIFS) between data packets and their related management packets for two nodes a and b. Timestamps from a single node can be used to measure $d(a, b)$.

Originally designed for operation using 1 μs resolution timestamps, Goodtry assumes that packets adhere to the 802.11 spacing. Goodtry is therefore not protocol agnostic.

The distance between two points can be measured by one participant using a pair of packets. The pair, which must satisfy the SIFS, can be a RTS-CTS pair or data-ACK pair.

$$d(a, b) = \frac{1}{2}\left(\tau_b^a(CTS) - \tau^a(RTS) - t_{RTS}\right) \tag{6.6}$$

$$d(a, b) = \frac{1}{2}\left(\tau_b^a(ACK) - \tau^a(data) - t_{data}\right) \tag{6.7}$$

For four packet exchanges, observers can measure distance using either:

$$d(a, b) = \frac{1}{4}\left(\tau_c^b(ACK) - \tau_c^b(CTS) - t_{data} - t_{CTS} - 2t_{SIFS}\right) \tag{6.8}$$

$$d(a, b) = \frac{1}{4}\left(\tau_c^a(data) - \tau_c^b(RTS) - t_{CTS} - t_{RTS} - 2t_{SIFS}\right) \tag{6.9}$$

The terms $t_{data}, t_{CTS}, t_{RTS}, t_{ACK}$ are the times to transmit the packets of the corresponding types.

Each packet exchange can only be used to measure distance between one pair of nodes a and b. Packets from c will not meet the SIFS for the RTS-CTS-data-ACK chain between a and b. Node c cannot measure $d(a, c)$ or $d(b, c)$ without directly communicating with a or b. Measuring all internode distances is thus an $O(n^2)$ operation.

Assuming the error from receiving and responding over the SIFS is similar to that for general timestamping, Goodtry falls into the TOA without reuse category, in which the best estimate of $d(b, q) - d(a, q)$ is made using both TOA and TDOA measurements. It is hypothesized that combining these will improve system location accuracy.

6.3 Hybrid Information for Location

This chapter addressed how best to compute the TDOA distance primitive $d(b, q) - d(a, q)$ using both TOA and TDOA information based on the same timestamps. In the TOA with reuse case, TDOA provides no useful information beyond that from TOA, and the optimal location technique combining PinPoint TOA with TDOA for active targets therefore appears to be equivalent to that for a TOA-only system. In the TOA without reuse case, TDOA improves the estimate of $d(b, q) - d(a, q)$. The question of how to compute location using both TOA and TDOA information remains open.

An important difference between the sphere solution from TOA and the hyperboloid solution from TDOA is that the sphere is bounded while the hyperboloid is not (see Figure 2.10). This suggests that spheres should be used whenever available to compute location. It is unclear then, how exactly the estimate of TDOA from the hybrid information should be combined with the TOA information producing the spheres. A geometric optimization using both spheres and hyperboloids with weights according to the standard deviations of error could be used, but this ignores the greater effect of TOA sphere information on constraining the location solution. Additional study is needed to complete this analysis with experimental data.

Future work is also needed to explore the more complex sensor network localization problem, where we solve for multiple node locations simultaneously, when TDOA information is available. The sensor network localization problem for TOA information only is described in Appendix C.

Chapter 7

PinPoint Usage Cases

Location-based services are a very hot area of context-aware computing that have been enabled by the decreasing cost and wide availability of services such as GPS that provide location information. Location-based services assume a service providing location information exists, which may not actually currently be the case for their operational environments. For example, some winners of the Android Developer Challenge [5] assume location information in indoor environments where GPS is not available. Although there are systems with the capability to provide location information in these environments[1], a system has not yet emerged that combines accuracy with low labor, equipment, and maintenance costs. Location-based services designers are waiting for location technologies to catch up to their operational requirements. PinPoint, with its design based on inexpensive components and accuracy in the range of a few meters, can enable or improve these existing applications and provide the location information for future location-based services.

In this chapter, we describe potential usage cases for PinPoint system variants. These examples highlight features of a PinPoint-based system, including the capability to find location without a fixed infrastructure and locate nonparticipant devices where hardware or software modifications are not feasible.

[1]See Appendix B for examples.

To realize these example applications, further work to integrate existing timestamping and optimization techniques is necessary. Techniques to combat multipath effects such as frequency, antenna diversity [60] and signal processing techniques for identifying and timestamping the direct-path signal [38] are necessary to improve timestamp quality. Techniques to identify non-LOS conditions [81] or other heuristics may further reduce errors. Sensor network localization techniques are necessary to locate mobile nodes when anchor node coverage is not sufficient to provide a full set of three anchors for two dimensions or four anchors for three dimensions.

These example applications are not comprehensive. The independence from the communication protocol and the ability to locate nonparticipants provide great design flexibility for other potential applications. Communication equipment operating in different or multiple frequency bands can be used.

7.1 Firefighters

Location systems for firefighters [32] aim to provide situational awareness for command and control functions, exit guidance, and homing capabilities. First responder situations are often time critical where the small time differences may have great impact on outcomes.

PinPoint can locate firefighters in a building as well as civilians, while potentially operating on the same network used for data or voice communication. For this usage case, we assume each vehicle and firefighter carries a participating PinPoint device and vehicles are GPS-enabled. Firefighters are mobile nodes that need not stay within communication range of the vehicles; messages can be routed through other firefighters to maintain connectivity. Civilians carrying other mobile devices are stranger nodes that can also be located as long as they the firefighter devices can timestamp their signals. An example is a cell phone making a 911 call from within the building.

PinPoint is rapidly deployable, requiring minimal on-site setup or calibration. Firefighters may enter a building containing no wireless infrastructure, while vehicles remain outside the building and use GPS to serve as anchor nodes.

PinPoint TOA measures the distance between the two types of participants, firefighters and vehicles. The measured distances define an instance of the sensor network localization problem, which can be solved using techniques outlined in Appendix C. The location of each firefighter can be measured in the three dimensional space of the building using the available TOA information. If building floorplans are available, firefighters can be tracked to specific rooms.

7.2 Retail Store

Location-based services can improve customer experiences in retail stores while also allowing the store to improve its own bottom line.

A retail store has a fixed wireless infrastructure with PinPoint anchor nodes to support an active TDOA system. These nodes may double as APs to provide network services to customers.

Customers moving through the store can access services through this network. Context-based services can guide customers through the physical store, directing to particular purchases or restrooms based on location and user preferences. Coupons or advertisements can also be delivered based on customer behavior to help the store move time-sensitive merchandise.

For these services, customers can use their own personal devices such as smartphones or use store-provided equipment. Usage of network services requires active transmission by the customer device, which can be used to compute the customer location from TDOA using the fixed infrastructure. The location can passed back to the client device.

Location information can also be used for marketing research. The physical layout of the store and products may be improved by studying how customers interact with the store. This may include what areas of the store they visit and how long they spend in those areas, information that is very valuable for marketing.

7.3 Museums

Location is used to improve exhibit design and the museum visitor experience [8]. Manual techniques for observing visitor behavior are extremely resource intensive. A staff person must follow and time a visitor throughout the exhibit, and during this time, it is not possible to follow other visitors.

The technological basis for a museum application is identical to the retail store. There is a fixed infrastructure available, which provides anchor nodes and a wireless network to locate nonparticipant users.

Visitors access wireless services through devices such as personal smartphones or network-enabled audio tour handheld devices. These devices can be located using the PinPoint TDOA active target variant. This location information can be collected without altering how the visitor interacts with the museum by retaining familiar existing services. New context aware services can also be provided such as locating other members of a visiting group or recommending other exhibits [19].

7.4 Rogue AP Problem

Rogue APs are a serious network security problem. A single rogue AP can allow unauthorized access to network resources, bypassing traditional network security mechanisms. Rogue APs come in two main varieties. The first variety is an AP attached to the wired network without permission from the network administrator. This variety introduces security vulnerabilities to the network. The second variety is deployed in the same physical area as an existing wireless network to spoof legitimate APs. This variety can be used to conduct man-in-the-middle attacks.

7.4.1 Rogue AP Detection

Prior work has produced two basic techniques for rogue AP detection. The first technique is traffic analysis of a rogue AP's wired connection [11, 80, 58]. This technique's advantages are its independence from the wireless protocol and the AP signal range, but it is only effective for detecting rogue APs attached to a wired network. The second technique is to monitor radio transmissions to detect rogue APs spoofing legitimate APs [18, 9, 45].

Wired Traffic Analysis

This technique differentiates between regular end hosts and unauthorized wireless APs connected to wired connections. Traffic analysis techniques [11, 80, 58] are based on the observation that wireless clients will have channel contention and slower link speed than wired clients. This difference leads to greater inter-packet and round trip times for wireless clients when compared to wired clients.

Wei et al. [80] demonstrated how a single monitoring device at the network gateway can use inter-packet times to differentiate between proper hosts and APs. The monitoring device operates in a passive, online manner. Wei presented high accuracy performance results with training, and 60%-76% accuracy without training for a network containing thousands of hosts.

Mano et al. [58] showed round trip time averages are efficient connection type differentiators when used with uniform packet sizes. Uniform packet sizes are achieved by active packet slicing at monitoring points in the network. Monitoring points must consider the type of network traffic (e.g. ftp, ssh, http) when measuring round trip times. Mano's results are asserted to be more robust across operating systems and wireless speeds.

Wireless Monitoring

Wireless monitoring analyzes information distributed by APs such as BSSID, MAC addresses, beacon sequence numbers, signal strength, and send times. Monitoring may be done either with fixed infrastructure such as APs and desktop machines or manually with mobile units such as laptops or handheld devices.

Simple detection schemes have an authorized list of APs BSSIDs and MAC addresses [18]. Rogue APs with unauthorized BSSID or MAC address values can be easily detected by checking the authorized list. The problem with this schemes is that BSSIDs and MAC addresses are easily spoofed.

In DAIR [9], detection of rogue APs using spoofed values depends on whether the rogue AP is near the authorized AP being spoofed. If both the rogue AP and authorized AP are in transmission range, detection is performed by analysis of beacon sequence numbers, which should be monotonically increasing. If both are not in transmission range, detection is made from comparing historical signal strength values, which is easy in the case when an AP is suddenly detectable and should not be, but may otherwise be resource intensive and inaccurate.

An alternate method to detect spoofing based on clock drifts was implemented by Jana and Kasera in [45]. Each crystal oscillator clock has a slightly different frequency. A legitimate AP has a drift ratio β_l. A spoofer has drift ratio β_s. A client with drift ratio β_c can verify the AP by capturing beacons and checking the ratio $\frac{\beta_l}{\beta_c}$ against a stored list of known APs. Since most clocks have different β values, it is easy to detect most spoofing. This technique is similar to that used in the identification of nodes in the onion routing system TOR by Murdoch [63].

7.4.2 Existing Rogue AP Location

Solutions to the physical location problem are less sophisticated. For rogue APs attached to the wired network, physical location of the AP requires documentation mapping wired switch ports to physical locations. The location process identifies the rogue AP MAC address from the router serving the rogue AP, locates the switch port from the MAC address, and finally requires a lookup in the documentation to find the switch port's physical location. Location of the second rogue AP variety is limited to manual site surveys. The area within transmission range of the detector must be searched by personnel with signal strength monitors and directional antennas to manually locate the AP, which may be difficult because of RF multipath effects.

7.4.3 PinPoint for Rogue AP Location

Assuming a rogue AP can be detected using the techniques of Chapter 7.4.1, PinPoint's active target TDOA variant can locate the rogue AP. This process is automated based upon transmissions from the rogue AP, requiring none of the manual processes associated with existing techniques. For this application, simply threatening discovery and location may suppress traffic from the rogue AP, which destroys its utility as an active malicious threat. Passive listener threats, of course, are not possible to discover or locate using the techniques in this dissertation.

7.5 Summary

Location determination is a very active research and application development area. In this chapter, we have described a few potential applications based on PinPoints capabilities to support sensor network localization and location of nonparticipant devices. There are many other cases where this technology can be used, and it is expected that new location-based services will appear as location technology performance improves.

Chapter 8

Conclusion

This dissertation explored the relation between time and distance in systems using inexpensive, imprecise clocks. Using off-the-shelf 802.11 hardware, we demonstrated measurements of TOA and TDOA distance primitives and a full TDOA location system in line-of-sight environments. These systems can achieve accuracy of 3m with time delay of a few seconds. This was achieved without external synchronization, adjustments to the physical clock frequency or offsets, or precision packet scheduling. Instead, using clock offsets and clock drifts we developed a piecewise linear time model that automatically corrects for changing clock drift values that cause nonlinear clock behavior. From this model, we also derived a consensus synchronization technique for general synchronization purposes.

Consensus synchronization techniques enable distributed synchronization to a common time scale by all connected nodes within a network. If separate send and receive timestamping biases can be found, we proved that simple, distributed iterative computations yield a synchronized consensus. Without the separate bias values, consensus synchronization clock drift and offset still provide a framework for understanding how to compute distances. Further research is necessary to compare the performance of consensus synchronization against existing synchronization systems.

With both TOA and TDOA primitives available, we discovered that time-based location systems should favor TOA over TDOA for accuracy. Solutions to the location problem are better constrained when using TOA information to generate spheres than when using TDOA information to generate hyperboloids. The TDOA distance primitive can also be estimated more accurately when using TOA information, especially when timestamps can be reused in the manner described in Chapter 6. For these reasons, it is better for location system participants to use TOA than TDOA. TDOA should be used for nonparticipants or for nodes without reliable timestamping capability.

The performance of a PinPoint-based system is dependent upon the quality of the input timestamps, which requires both attention to mundane issues like proper shielding and grounding as well as advanced signal processing techniques to mitigate multipath effects. The Atheros and RN-134 demonstration platforms are remarkable in the access provided to higher precision timestamps, but they are still deficient in these areas.

To complete a location system for commercial purposes, integration work is needed. A combination of timestamping, synchronization, distance techniques, and optimization are required to generate a complete location system. This dissertation has established techniques for synchronization and distance, and we believe existing optimization techniques are adequate. The major hurdle to completing a practical system is the hardware platform supporting precision, accurate timestamping. Completion of the location system will create a new capability — location information will be available in any area covered by wireless networks with low setup and hardware costs.

Appendix A

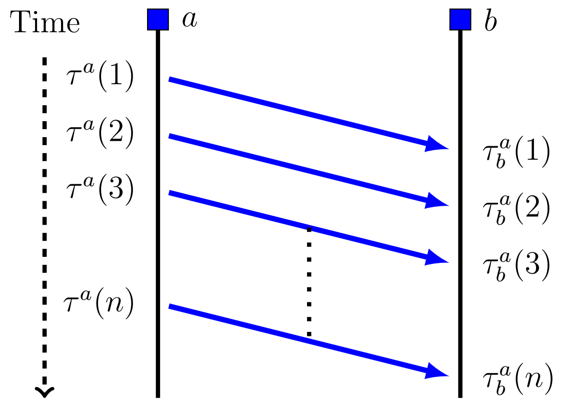
Derivations

The use of average clock drift to compare times eliminates the need to correct for clock drift between pairs of nodes and bases distance on the average of rate of clocks in use. The average clock drift for a set of clocks $\Omega = \{a, b, \ldots\}$ is given by

$$\beta_* = \frac{1}{|\Omega|} \sum_{\omega \in \Omega} \beta_\omega$$

The derivations make use of the locally linear clock model with card timestamping bias terms.

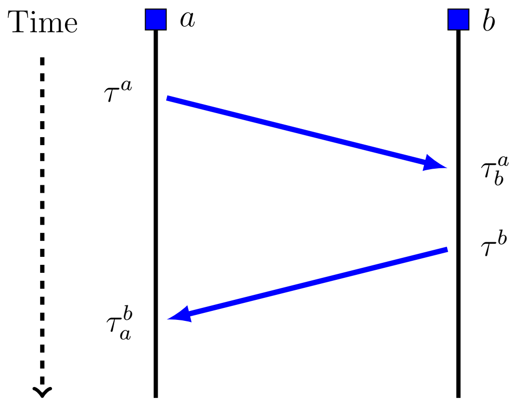
$$\tau^a = \beta_a \left(t^a + \alpha_a\right) + s_a$$
$$\tau_b^a = \beta_b \left(t^a + \alpha_b + d(a,b)\right) + r_a$$



A.1 Point Clock Drift Error Estimation

We estimate the impact of timestamp errors on the point estimate of clock drift.

$$
\begin{aligned}
\widehat{\left[\frac{\beta_b}{\beta_a}\right]} &= \frac{\tau^a_b(2)-\tau^a_b(1)}{\tau^a(2)-\tau^a(1)} \\
&= \frac{\beta_b(t_2+\alpha_b+d_2)+r_b+e^a_b(2)-\beta_b(t_1+\alpha_b+d_1)-r_b-e^a_b(1)}{\beta_a(t_2+\alpha_a)+s_a+e^a(2)-\beta_a(t_1+\alpha_a)-s_a-e^a(1)} \\
&= \frac{\beta_b(t_2-t_1+d_2-d_1)+e^a_b(2)-e^a_b(1)}{\beta_a(t_2-t_1)+e^a(2)-e^a(1)} \\
&= \frac{\beta_b+\beta_b\frac{d_2-d_1}{t_2-t_1}+\frac{e^a_b(2)-e^a_b(1)}{t_2-t_1}}{\beta_a+\frac{e^a(2)-e^a(1)}{t_2-t_1}} * \frac{\beta_a-\frac{e^a(2)-e^a(1)}{t_2-t_1}}{\beta_a-\frac{e^a(2)-e^a(1)}{t_2-t_1}} \\
&= \frac{\left(\beta_b+\beta_b\frac{d_2-d_1}{t_2-t_1}+\frac{e^a_b(2)-e^a_b(1)}{t_2-t_1}\right)*\left(\beta_a-\frac{e^a(2)-e^a(1)}{t_2-t_1}\right)}{\beta_a^2-\left(\frac{e^a(2)-e^a(1)}{t_2-t_1}\right)^2} \\
&= \frac{\beta_b\beta_a+\beta_b\beta_a\frac{d_2-d_1}{t_2-t_1}+\beta_a\frac{e^a_b(2)-e^a_b(1)}{t_2-t_1}-\beta_b\frac{e^a(2)-e^a(1)}{t_2-t_1}}{\beta_a^2-\left(\frac{e^a(2)-e^a(1)}{t_2-t_1}\right)^2} \\
&\quad -\frac{\beta_b\frac{(d_2-d_1)(e^a(2)-e^a(1))}{(t_2-t_1)^2}+\frac{(e^a_b(2)-e^a_b(1))(e^a(2)-e^a(1))}{(t_2-t_1)^2}}{\beta_a^2-\left(\frac{e^a(2)-e^a(1)}{t_2-t_1}\right)^2} \\
&\approx \frac{\beta_b\beta_a+\beta_b\beta_a\frac{d_2-d_1}{t_2-t_1}+\beta_a\frac{e^a_b(2)-e^a_b(1)}{t_2-t_1}-\beta_b\frac{e^a(2)-e^a(1)}{t_2-t_1}}{\beta_a^2} \\
&\approx \frac{\beta_b}{\beta_a}+\frac{d_2-d_1}{t_2-t_1}+\frac{\sqrt{4}\sigma}{t_2-t_1}
\end{aligned}
$$



The $\frac{d_2-d_1}{t_2-t_1}$ term indicates clock drift error will increase with physical movement. As discussed for Equation 3.21, this error is negligible for human speeds.

A.2 Clock Offset $\theta_{a\rightarrow b}$ in Consensus Time Scale

One round of messages provides the offset between two clocks in the consensus time scale. Using $\theta_{a\mapsto b}$ without clock drift correction as a starting point, we add global clock drift.

$$\begin{aligned}
&\frac{1}{2}\left[\frac{\beta_*}{\beta_b}(\tau^b+\tau_b^a)-\frac{\beta_*}{\beta_a}(\tau^a+\tau_a^b)\right]\\
&=\frac{1}{2}\left[\frac{\beta_*}{\beta_b}\beta_b(t^b+\alpha_b+t^a+\alpha_b+d(a,b))-\frac{\beta_*}{\beta_a}\beta_a(t^a+\alpha_a+t^b+\alpha_a+d(a,b))\right]\\
&\quad+\frac{1}{2}\frac{\beta_*}{\beta_b}(s_b+r_b)-\frac{1}{2}\frac{\beta_*}{\beta_a}(s_a+r_a)\\
&=\frac{1}{2}\left[\beta_*(t^b+\alpha_b+t^a+\alpha_b+d(a,b))-\beta_*(t^a+\alpha_a+t^b+\alpha_a+d(a,b))\right]\\
&\quad+\frac{1}{2}\frac{\beta_*}{\beta_b}(s_b+r_b)-\frac{1}{2}\frac{\beta_*}{\beta_a}(s_a+r_a)\\
&=\frac{1}{2}\left[\beta_*(2\alpha_b-2\alpha_a)\right]+\frac{1}{2}\frac{\beta_*}{\beta_b}(s_b+r_b)-\frac{1}{2}\frac{\beta_*}{\beta_a}(s_a+r_a)\\
&=\beta_*(\alpha_b-\alpha_a)+\frac{1}{2}\frac{\beta_*}{\beta_b}(s_b+r_b)-\frac{1}{2}\frac{\beta_*}{\beta_a}(s_a+r_a)
\end{aligned}$$

We solve for the difference of α terms. If the clock drift values are constant, $\theta_{a\to b}$ is constant as well.

$$\begin{aligned}\theta_{a\to b} &= \frac{1}{2}\left[\frac{\beta_*}{\beta_b}\big((\tau^b+\tau^a_b)-(s_b+r_b)\big)-\frac{\beta_*}{\beta_a}\big((\tau^a+\tau^b_a)-(s_a+r_a)\big)\right] \\ &= \beta_*(\alpha_b-\alpha_a)\end{aligned} \tag{A.1}$$

A.2.1 Consensus Clock Offset

The consensus clock offset $\theta_{a\to *}$ maps a single node's times to the consensus clock.

$$\theta_{a\to *} = \frac{\beta_*}{|\Omega|}\sum_{\omega\in\Omega}(\alpha_\omega-\alpha_a) \tag{A.2}$$

$$= \frac{1}{|\Omega|}\sum_{\omega\in\Omega}(\theta_{a\to\omega}) \tag{A.3}$$

Using $\theta_{a\to *}$ and $\theta_{b\to *}$ is equivalent to using $\theta_{a\to b}$. If nodes a and b are in the same connected network but cannot communicate directly, they can compute their pairwise offset.

Theorem A.1.

$$\theta_{a\to *}-\theta_{b\to *}=\theta_{a\to b} \tag{A.4}$$

Proof.

$$\begin{aligned}
\theta_{a\to*} - \theta_{b\to*} &= \frac{\beta_*}{|\Omega|}\sum_{\omega\in\Omega}(\alpha_\omega - \alpha_a) - \frac{\beta_*}{|\Omega|}\sum_{\omega\in\Omega}(\alpha_\omega - \alpha_b) \\
&= \frac{\beta_*}{|\Omega|}\sum_{\omega\in\Omega}\left[(\alpha_\omega - \alpha_a) - (\alpha_\omega - \alpha_b)\right] \\
&= \frac{\beta_*}{|\Omega|}\sum_{\omega\in\Omega}\left[\alpha_b - \alpha_a\right] \\
&= \frac{\beta_*}{|\Omega|}|\Omega|\left(\alpha_b - \alpha_a\right) \\
&= \beta_*\left(\alpha_b - \alpha_a\right) \\
&= \theta_{a\to b}
\end{aligned}$$

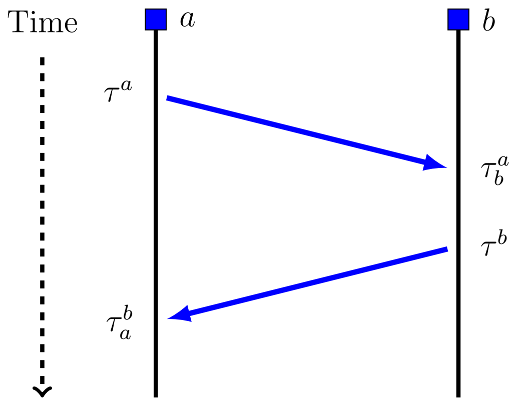
□

Corollary A.2.

$$\theta_{a\to*} = \theta_{a\to b} + \theta_{b\to*} \tag{A.5}$$

To translate τ_a, a time from a's timescale, to the consensus timescale

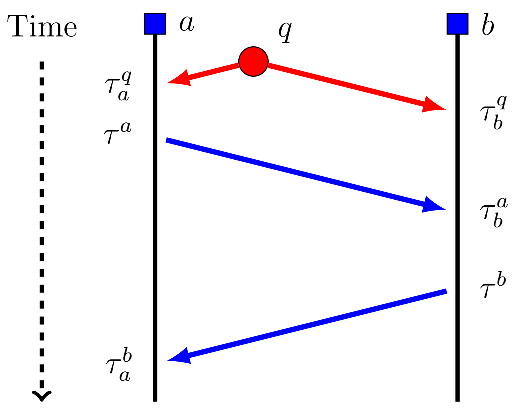
$$\tau_*(\tau_a) = \frac{\beta_*}{\beta_a}\tau_a + \theta_{a\to*} \tag{A.6}$$



A.3 TOA

$$
\begin{aligned}
&\frac{1}{2}\left[\frac{\beta_*}{\beta_a}\left(\tau_a^b-\tau^a\right)-\frac{\beta_*}{\beta_b}\left(\tau^b-\tau_b^a\right)\right]\\
&=\frac{1}{2}\Bigg[\frac{\beta_*}{\beta_a}\left(\beta_a\left(t^b+\alpha_a+d(a,b)-t^a-\alpha_a\right)+(r_a-s_a)\right)\\
&\qquad-\frac{\beta_*}{\beta_b}\left(\beta_b\left(t^b+\alpha_b-t^a-\alpha_b-d(a,b)\right)-(s_b-r_b)\right)\Bigg]\\
&=\frac{1}{2}\Bigg[\frac{\beta_*}{\beta_a}\beta_a\left(t^b+\alpha_a+d(a,b)-t^a-\alpha_a\right)\\
&\qquad-\frac{\beta_*}{\beta_b}\beta_b\left(t^b+\alpha_b-t^a-\alpha_b-d(a,b)\right)\Bigg]\\
&\qquad+\frac{1}{2}\left[\frac{\beta_*}{\beta_a}(r_a-s_a)+\frac{\beta_*}{\beta_b}(r_b-s_b)\right]\\
&=\frac{1}{2}\left[\beta_*\left(t^b+d(a,b)-t^a\right)-\beta_*\left(t^b+t^a-d(a,b)\right)\right]\\
&\qquad+\frac{1}{2}\left[\frac{\beta_*}{\beta_a}(r_a-s_a)+\frac{\beta_*}{\beta_b}(r_b-s_b)\right]\\
&=\frac{1}{2}\left[\beta_*\left(t^b-t^a\right)+\beta_*d(a,b)-\beta_*\left(t^b+t^a\right)+\beta_*d(a,b)\right]\\
&\qquad+\frac{1}{2}\left[\frac{\beta_*}{\beta_a}(r_a-s_a)+\frac{\beta_*}{\beta_b}(r_b-s_b)\right]\\
&=\beta_*d(a,b)+\frac{1}{2}\left[\frac{\beta_*}{\beta_a}(r_a-s_a)+\frac{\beta_*}{\beta_b}(r_b-s_b)\right]
\end{aligned}
$$

$$\beta_*d(a,b)=\frac{1}{2}\left[\frac{\beta_*}{\beta_a}\left[\left(\tau_a^b-\tau^a\right)-(r_a-s_a)\right]-\frac{\beta_*}{\beta_b}\left[\left(\tau^b-\tau_b^a\right)-(s_b-r_b)\right]\right] \tag{A.7}$$



A.4 TDOA Variants

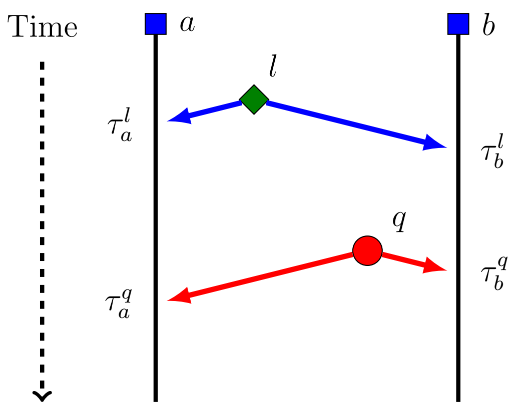
There are currently three distinct variants of TDOA.

A.4.1 TDOA with Active Target

$$\begin{aligned}
&\frac{\beta_*}{\beta_b}\left[\tau_b^q - \frac{1}{2}\left(\tau^b + \tau_b^a\right)\right] - \frac{\beta_*}{\beta_a}\left[\tau_a^q - \frac{1}{2}\left(\tau^a + \tau_a^b\right)\right] \\
&= \frac{\beta_*}{\beta_b}\beta_b\left[t^q + \alpha_b + d(b,q) - \frac{1}{2}\left(t^b + \alpha_b + t^a + \alpha_b + d(a,b)\right)\right] + \frac{\beta_*}{\beta_b}\left[r_b - \frac{1}{2}(s_b + r_b)\right] \\
&\qquad - \frac{\beta_*}{\beta_a}\beta_a\left[t^q + \alpha_a + d(a,q) - \frac{1}{2}\left(t^a + \alpha_a + t^b + \alpha_a + d(a,b)\right)\right] - \frac{\beta_*}{\beta_a}\left[r_a - \frac{1}{2}(s_a + r_a)\right] \\
&= \beta_*\left[t^q + d(b,q) - \frac{1}{2}\left(t^b + t^a + d(a,b)\right)\right] + \frac{1}{2}\frac{\beta_*}{\beta_b}\left(r_b - s_b\right) \\
&\qquad - \beta_*\left[t^q + d(a,q) - \frac{1}{2}\left(t^a + t^b + d(a,b)\right)\right] - \frac{1}{2}\frac{\beta_*}{\beta_a}\left(r_a - s_a\right) \\
&= \beta_*\left[d(b,q) - d(a,q)\right] + \frac{1}{2}\frac{\beta_*}{\beta_b}\left(r_b - s_b\right) - \frac{1}{2}\frac{\beta_*}{\beta_a}\left(r_a - s_a\right)
\end{aligned}$$

Solving for the difference of distances yields:

$$\begin{aligned}
\beta_*[d(b,q) - d(a,q)] = &\frac{\beta_*}{\beta_b}\left(\tau_b^q - \frac{1}{2}(\tau^b + \tau_b^a) - \frac{1}{2}(r_b - s_b)\right) \\
&- \frac{\beta_*}{\beta_a}\left(\tau_a^q - \frac{1}{2}(\tau^a + \tau_a^b) - \frac{1}{2}(r_a - s_a)\right)
\end{aligned} \tag{A.8}$$



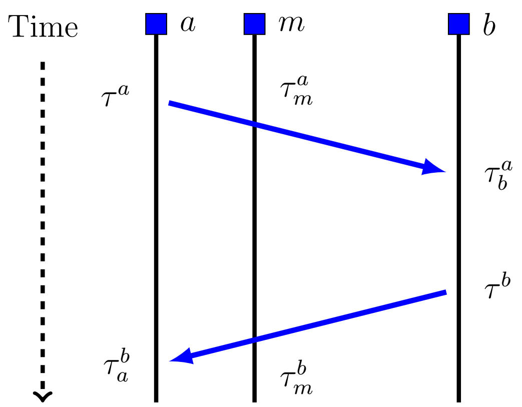
A.4.2 TDOA without Send Times

This variant requires no send times, but uses known geometry for a, b, l.

$$\begin{aligned}
&\frac{\beta_*}{\beta_b}\left(\tau_b^q - \tau_b^l\right) - \frac{\beta_*}{\beta_a}\left(\tau_a^q - \tau_a^l\right) \\
&= \frac{\beta_*}{\beta_b}\beta_b\left(t^q + \alpha_b + d(b,q) - t^l - \alpha_b - d(b,l)\right) + \frac{\beta_*}{\beta_b}(r_b - r_b) \\
&\qquad - \frac{\beta_*}{\beta_a}\beta_a\left(t^q + \alpha_a + d(a,q) - t^l - \alpha_a - d(a,l)\right) + \frac{\beta_*}{\beta_a}(r_a - r_a) \\
&= \beta_*\left(t^q + d(b,q) - t^l - d(b,l)\right) - \beta_*\left(t^q + d(a,q) - t^l - d(a,l)\right) \\
&= \beta_*\left(d(b,q) - d(b,l)\right) - \beta_*\left(d(a,q) - d(a,l)\right) \\
&= \beta_*\left[\left(d(b,q) - d(a,q)\right) - \left(d(b,l) - d(a,l)\right)\right]
\end{aligned}$$

Solving for the difference of distances for the unknown q yields

$$\beta_*\left[d(b,q) - d(a,q)\right] = \beta_*\left[d(b,l) - d(a,l)\right] + \frac{\beta_*}{\beta_b}\left(\tau_b^q - \tau_b^l\right) - \frac{\beta_*}{\beta_a}\left(\tau_a^q - \tau_a^l\right) \tag{A.9}$$



A.4.3 TDOA for Low Traffic Mobile Node

This variant does not require the mobile node being located to send messages for the basic message set.

$$
\begin{aligned}
&\frac{\beta_*}{\beta_m}\left[\tau^b_m - \tau^a_m\right] - \frac{1}{2}\frac{\beta_*}{\beta_b}\left(\tau^b - \tau^a_b\right) + \frac{1}{2}\frac{\beta_*}{\beta_a}\left(\tau^a - \tau^b_a\right) \\
&= \frac{\beta_*}{\beta_m}\beta_m\left(t^b + \alpha_m + d(b,m) - t^a - \alpha_m - d(a,m)\right) \\
&\quad - \frac{1}{2}\frac{\beta_*}{\beta_b}\beta_b\left(t^b + \alpha_b - t^a - \alpha_b - d(a,b)\right) \\
&\quad + \frac{1}{2}\frac{\beta_*}{\beta_a}\beta_a\left(t^a + \alpha_a - t^b - \alpha_a - d(a,b)\right) \\
&\quad + \frac{1}{2}\frac{\beta_*}{\beta_b}(r_b - s_b) - \frac{1}{2}\frac{\beta_*}{\beta_a}(r_a - s_a) \\
&= \beta_*\left(t^b + d(b,m) - t^a - d(a,m)\right) - \frac{1}{2}\beta_*\left(t^b - t^a - d(a,b)\right) + \frac{1}{2}\beta_*\left(t^a - t^b - d(a,b)\right) \\
&\quad + \frac{1}{2}\frac{\beta_*}{\beta_b}(r_b - s_b) - \frac{1}{2}\frac{\beta_*}{\beta_a}(r_a - s_a) \\
&= \beta_*\left[(d(b,m) - d(a,m)) + (t^b - t^a) - \frac{1}{2}(t^b - t^a) + \frac{1}{2}d(a,b) + \frac{1}{2}(t^a - t^b) - \frac{1}{2}d(a,b)\right] \\
&\quad + \frac{1}{2}\frac{\beta_*}{\beta_b}(r_b - s_b) - \frac{1}{2}\frac{\beta_*}{\beta_a}(r_a - s_a) \\
&= \beta_*\left(d(b,m) - d(a,m)\right) + \frac{1}{2}\frac{\beta_*}{\beta_b}(r_b - s_b) - \frac{1}{2}\frac{\beta_*}{\beta_a}(r_a - s_a)
\end{aligned}
$$

$$\begin{aligned}\beta_* \left[d(b,m) - d(a,m)\right] =& \frac{\beta_*}{\beta_m}\left(\tau^b_m - \tau^a_m\right) \\ &- \frac{1}{2}\frac{\beta_*}{\beta_b}\left(\left(\tau^b - \tau^a_b\right) - (s_b - r_b)\right) \\ &+ \frac{1}{2}\frac{\beta_*}{\beta_a}\left(\left(\tau^a - \tau^b_a\right) - (s_a - r_a)\right)\end{aligned} \tag{A.10}$$

A.5 Card Bias

With a, b colocated such that $d(a,q) = d(b,q)$ and $d(a,b) = 0$, we can solve for the card bias values $r_a - s_a$ and $r_b - s_b$. Bias values are measured in local clock times.

$$\begin{aligned}(\tau^q_b - \tau^b) + \frac{\beta_b}{\beta_a}(\tau^b_a - \tau^q_a) &= \beta_b(t^q + \alpha_b + d(b,q) - t^b - \alpha_b) + r_b - s_b \\ &\quad + \frac{\beta_b}{\beta_a}[\beta_a(t^b + \alpha_a + d(a,b) - t^q - \alpha_a - d(a,q)) + r_a - r_a] \\ &= \beta_b(t^q + d(b,q) - t^b) + r_b - s_b \\ &\quad + \frac{\beta_b}{\beta_a}[\beta_a(t^b + d(a,b) - t^q - d(a,q))] \\ &= \beta_b(t^q + d(b,q) - t^b) + r_b - s_b \\ &\quad + \beta_b(t^b + d(a,b) - t^q - d(a,q)) \\ &= r_b - s_b + \beta_b(t^q + d(b,q) - t^b + t^b + d(a,b) - t^q - d(a,q)) \\ &= r_b - s_b + \beta_b(d(b,q) + d(a,b) - d(a,q)) \\ &= r_b - s_b\end{aligned}$$

The situations for a and b are identical, thus

$$r_a - s_a = (\tau^q_a - \tau^a) + \frac{\beta_a}{\beta_b}(\tau^a_b - \tau^q_b) \tag{A.11}$$

$$r_b - s_b = (\tau^q_b - \tau^b) + \frac{\beta_b}{\beta_a}(\tau^b_a - \tau^q_a) \tag{A.12}$$

A.6 Alternate Clock Drift Notation

The alternate form of clock notation is

$$\beta = 1 + \delta \qquad\qquad |\delta| < 10^{-4} \tag{A.13}$$

The $|\delta|$ bound comes from the manufacturing standard for quartz oscillators.

Lemma A.3. *If $\delta \ll 1$,*

$$\frac{1}{1+\delta} \approx 1 - \delta \tag{A.14}$$

Proof.

$$\begin{aligned}
\frac{1}{1+\delta} &= \frac{1}{1+\delta}\frac{(1-\delta)}{(1-\delta)} \\
&= \frac{(1-\delta)}{(1-\delta^2)} \\
&= \frac{(1-\delta)}{(1-\delta^2)}\frac{(1+\delta^2)}{(1+\delta^2)} \\
&= \frac{(1-\delta+\delta^2-\delta^3)}{(1-\delta^4)} \\
&= \frac{(1-\delta+\delta^2-\delta^3)}{(1-\delta^4)}\frac{(1+\delta^4)}{(1+\delta^4)} \\
&\vdots \\
&= \lim_{n\to\infty}\frac{1}{1-\delta^n}\sum_{i=0}^{n}(-1)^i\delta^i \\
&= \lim_{n\to\infty}\sum_{i=0}^{n}(-1)^i\delta^i
\end{aligned}$$

Since $\delta \ll 1$, the higher powers of δ are negligible.

$$\frac{1}{1+\delta} = \lim_{n\to\infty}\sum_{i=0}(-1)^i\delta^i \approx 1 - \delta$$

$\square$

Appendix B

Existing Time-based Location Systems

The location problem has been studied extensively and many different time-based location systems currently exist. In this appendix we review current operational and research systems.

Time-based locations systems are based on either TOA or TDOA measurements, which result in spherical or hyperboloid solution sets respectively. Another important characteristic is whether the target node q must transmit actively or can listen passively. Active transmission requires more network resources and generally means the system can support fewer target nodes.

	Mobile Unit Mode	
	Active	Passive
TOA	PinPoint TOA	E-OTD
	Goodtry	GPS
	Pseudolite	D-GPS
	Active Bat	A-GPS
		Cricket
TDOA	GSM UL-TOA	E-OTD
	PinPoint TDOA	LORAN

Table B.1: Location systems classification

System Name	Approx Range	Accuracy	Source
GPS	global	4 m	[76]
DGPS	300 km	1 m	
LORAN, DECCA	1000 km	460m	[30]
Goodtry	100 m	~4 m	[42]
PinPoint	100 m	3 m	
Pseudolite (Code-Phase)	500 m	2 m	[52]
Pseudolite (Carrier-Phase)	500 m	0.01 m	[52]
Mobile Phone Location	40 km	50 m	
Cricket	5 m	30 cm	[73]
Active Bat	5 m	10 cm	[41]

Table B.2: Existing location systems: scope and accuracy

Table B.2 summarizes the approximate range and accuracy of existing time-based location systems. The ranges are the approximate distances between anchor and target nodes. No attempt has been made to standardize the accuracy values for either statistical measures or responsiveness.

There are two types of phase estimation that can be used to determine distance, code-phase and carrier-phase. Code-phase estimation measures the receive time for known portions of the radio signal with basic timestamping. This is available in all location systems we describe here. Carrier-phase estimation determines the phase shift of the carrier wave modulo the wavelength. This can allow precise distance measurement, but is also highly ambiguous unless a second source of information supplies coarse measurement distances. Carrier-phase estimation is available for GPS and pseudolite systems.

B.1 Global Positioning System (GPS)

NAVSTAR GPS is a satellite-based trilateration location system. It is currently the only global navigational satellite system (GNSS) operational in the world. The European Galileo and Russian GLONASS systems are very similar in design and function and are planned for political reasons, but neither is yet fully operational.

GPS is used for aviation, naval, and ground navigation, both for military as well as civilian applications.

GPS is composed of three segments. The first segment is the space segment, a constellation of satellites orbiting the Earth. The second segment is the control segment. It consists of ground sites used to manage and maintain the GPS system. The third segment, the user segment, consists of all GPS receivers. Since none of the receivers transmit, an unlimited number of users is supported.

The ground-based GPS control segment is used to synchronize clocks and track satellites. Using atomic clocks, the GPS is able to attain synchronization to within 1 ns [69] to an ensemble of the satellite and ground-based clocks. Tracking of satellite positions is performed by ground-based radar, and orbit data are transmitted up to the satellites, to be retransmitted back to users.

Each GPS satellite continuously transmits data messages using code division multiple access (CDMA) technology, which allows all satellites to transmit simultaneously. Each data message is 1500 bits and is sent at 50 bits/second, requiring 30 seconds per message. The data is xor'ed with a pseudorandom code unique to each satellite with higher bitrate. The resulting bit stream is modulated onto the carrier medium. A receiver can recover the bit stream for any satellite using the satellite's published code.

There are three pseudorandom noise code types, coarse/acquisition (C/A), precision (P) and Y [76]. Each satellite has unique codes. A coarse/acquisition code is sufficient for standard position service (SPS) operation, which is the civilian version of GPS. A C/A-code is a 1.023 MHz code with length 1 ms, for a total of 1023 bits. It also improves the acquisition time for the P and Y codes. A P-code is a 10.23 MHz code with length of 7 days and allows more accurate positioning. Y-codes are encrypted versions of P-codes, designed for security purposes to prevent adversaries from spoofing GPS satellites.

Abbr.	Freq	Length
C/A	1.023 MHz	1 ms
P	10.23 MHz	7 days
Y	10.23 MHz	7 days

Table B.3: GPS Pseudorandom Noise Codes

Each message frame consists of 5 subframes, each 300 bits with a preamble for synchronization. The transmission of each subframe is synchronized to the atomic clock to make all satellites transmit their messages at the same time. The message contents by subframe are:

1. time values, Week number, satellite clock corrections, satellite health: if orbit has recently been perturbed, data is not used
2. Ephemeris
3. Ephemeris - precise orbit info updated every 2 hours, valid for 4 hours.
4. Almanac
5. Almanac - (1/25) 25 messages are required to reconstruct the entire almanac. This requires 12.5 minutes. Almanacs are updated daily, but can be used for weeks.

The ephemeris and almanac data span multiple subframes.

GPS provides two levels of service, standard positioning service (SPS) and precision positioning service (PPS). PPS provides several advantages over SPS. PPS uses both the L1 and L2 frequency bands, while SPS uses only L1. Usage of both bands allows better correction of atmospheric errors because the atmosphere affects each frequency band's speed differently. PPS also provides encryption capability to prevent satellite spoofing. PPS's higher chipping rate (more bits/s) improves receive timing precision. Most commercial grade GPS devices use SPS.

Band	Freq	Data
L1	1575.42 MHz	Nav + C/A, Nav + P(Y)
L2	1227.6 MHz	Nav + P(Y)

Table B.4: GPS Frequency Bands

Each GPS satellite transmits on the two frequency bands L1 and L2. C/A-modulated data is on L1. P(Y)-modulated data is on both L1 and L2. For SPS service, a user listens only to the L1 band. For PPS service, a user listens to both the L1 and L2 bands.

The outline of GPS receiver operation is as follows. A GPS receiver listens to the known L1 frequency. GPS receiver identifies a C/A code, either by brute force search or access to an almanac with rough initial position and time to determine satellites in view. If the GPS receiver does not have a current almanac, it retrieves it from the satellite corresponding to the C/A code. With the almanac data, the receiver can demodulate the signals from multiple satellites. The receiver records receive times of each subframe from each satellite. Once the receiver collects times for at least four satellites, it then solves for its location.

To solve for location, the receiver solves a modified trilateration problem. In addition to solving for the position solution (x, y, z), the receiver also solves for the receiver clock time when the messages were sent, t_s. By solving for this time, the receiver may use an inexpensive quartz clock rather than an atomic clock. For any satellite, the quantity $c(t_r - t_s)$ is called the pseudorange because quantity t_s is part of the solution. At least four satellites in view are required to solve for all four variables x, y, z, t_s. Solving for (x, y, z, t) is a nonlinear optimization problem, just as in trilateration.

Sources of error for GPS include

1. ionosphere propagation

2. satellite orbit errors

3. clock errors
4. multipath
5. troposphere propagation

Initially designed as a military system, GPS included selective availability to reduce location accuracy for non-military SPS uses. Clock errors were intentionally introduced to reduce location accuracy to $\sim$50m [39]. Selective availability was deactivated in May 2000.

B.1.1 Assisted GPS (A-GPS)

A-GPS uses an additional ground-based transmitter to reduce time to get a location fix. The data bit rate for between the ground-based transmitter and the mobile unit is faster than the rate between satellites and the mobile unit. Code information, ephemeris and almanac data are sent over this faster connection, leaving only the time information to be sent over the satellite link. From this data a receiver can both determine which satellites are visible and identify their code information, obviating a code search of all satellites.

B.1.2 Differential GPS (DGPS)

DGPS uses additional ground-based stations to increase GPS accuracy. DGPS relies on the principle that errors will be strongly correlated in space, so that known errors at one location may be used to correct similar errors in the same vicinity. Although originally designed to eliminate the selective availability introduced clock errors, D-GPS has been found to improve location error for errors from other sources as well.

DGPS stations are at known locations. Each station estimates its own location using the standard GPS procedure. Since each station knows its actual location, it computes the difference between the estimated location and its actual location. The station supplies this difference as a correction factor for its local area.

A number of agencies operate DGPS stations. The Department of Transportation's Federal Aviation Administration operates the Wide Area Augmentation System to enhance location of aircraft. The FAA seeks to have GPS location for blind instrument landings, which requires accuracy of 4 m laterally and 1m vertically. These operation requirements are more stringent than most other applications. The United States Coast Guard Navigation Center operates a DGPS service for ship navigation. The National Oceanic Atmospheric Administration's National Geodetic Survey operates Continuously Operating Reference Stations (CORS).

B.1.3 Planned GPS Enhancements

Further GPS enhancements are planned to improve GPS performance for both civilian and military users [39]. These enhancements will extend the existing system while preserving legacy functionality. Civilian GPS service will be improved with the addition of civilian codes in the L1 and L2 bands as well as the addition of a L5 band at 1176.45 MHz. New M-codes will be added for military usage to the L1 and L2 bands. These changes are designed to improve system accuracy while improving coverage.

B.2 Pseudolites

Pseudolite systems [22] are loosely based on the GPS concept, with GPS satellites replaced by false satellite (pseudolite) transceivers. Each transceiver can compute the distance to any other transceiver, so location is determined using trilateration. One of the original pseudolite applications was determining location for Mars rovers operating in a line-of-sight environment [51]. For precision location of the rovers, Lemaster and Rock developed pseudolite self-calibrating location techniques using both code-phase and carrier-phase measurements to achieve cm level accuracy[52]. Pseudolites for this application were not constructed with atomic clocks, but instead used temperature compensated crystal oscillators.

B.3 Long Range Aid to Navigation (LORAN), DECCA

LORAN and DECCA were TDOA based location systems primarily designed to assist naval navigation. LORAN was run by the US Coast Guard until 2010, and DECCA was the European counterpart to the LORAN system. The final incarnation of LORAN was LORAN-C. Below we will refer to LORAN-C simply as LORAN. The superior accuracy and worldwide availability of GPS made LORAN and DECCA obsolete.

LORAN and DECCA systems were accurate to approximately 460m [30]. Both systems used ground based transmitters that were high powered 200 kW - 2 MW and operated at 90-110 kHz frequencies. Since the systems were very similar, we will focus on the LORAN system.

The basic unit in the LORAN system was the chain. A chain combined a single master transmitter paired with secondary transmitters. Each chain contained 3-5 total transmitters. The transmitters were arranged in a star formation, with the master transmitter located at the center. TDOA was computed for the master with each of the secondary transmitters. The master transmitted its signal first, with each secondary transmitter sending successively in turn after specified delay periods.

All transmitters used cesium-based atomic clocks. LORAN masters were synchronized to UTC time within 100 ns. Secondary transmitters were not synchronized to UTC time [47].

A LORAN receiver, on the ship to be located, timestamped each transmitters signal. The location of LORAN transmitters and delays between master and secondary transmitters were published, so the receiver had sufficient data to solve for its location using TDOA and hyperbolic location.

B.4 Goodtry

Goodtry is a trilateration system developed at the Eberhard Karls University of Tubingen and the University of Stuttgart for 802.11 devices [42]. The system uses an unmodified wireless driver to timestamp with 1 μs precision. Enhanced versions of this system were demonstrated by Ciurana, Barcelo-Arroyo, and Izquierdo with a 44 MHz clock using hardware modifications to a wireless card [20, 21] and Goldman and Bateman with hardware modifications to correct timestamp bias and an field programmable gate array (FPGA) to timestamp packets [38].

Goodtry makes use of the short interframe spacing (SIFS) of the 802.11 protocol. The SIFS is the minimum time period for which the transmission medium will remain clear following the transmission of a packet. For certain sequences of packet transfers, the SIFS will be exactly the time between the reception of one transmission and the beginning of another transmission at a node. By default, the SIFS parameter is usually 10μs.

We present the case of a data packet followed by an ACK. A sender performing the distance measurement sends a data packet to a receiver. Following the proper receipt of the data packet, the receiver will wait the SIFS before transmitting an ACK. The sender of the data packet experiences the transmission idle for the SIFS plus the transit time for the data packet and returning ACK. By subtracting the SIFS from the observed idle time, the sender can compute the distance between the two nodes.

Similar situations involving a chain of packets with request to send, clear to send, data, and ACK packets allow TOA between the communicating nodes to be computed by any monitoring node.

In this system, the 1 μs precision must be overcome by statistical means. Since the range of 802.11 devices is < 300 m, all measured times are generally either 0 or 1. Goodtry makes multiple measurements and also uses remote measurements of the packets exchanged to increase sample size.

B.5 Mobile Phone Tracking

Under Phase 2 of Enhanced 9-1-1 (E911) service, the Federal Communications Commission (FCC) requires that mobile phone providers be capable of locating mobile phones for emergency calls to assist first responders [3]. The accuracy requirement is 50-300m depending on the technology used. Mobile phone providers are supposed to provide quarterly updates to the FCC about the progress of location technology.

Many mobile phone tracking technologies exist to meet this location requirement [15, 89]. One option to meet the FCC E911 requirements is the use of A-GPS. This requires separate hardware within the mobile phone to act as the GPS receiver. Triangulation based on Angle of Arrival (AOA) is another option, but requires special antenna hardware.

Since we have already described GPS above in section B.1, we will concentrate on the other timing based measurement systems. The specific method used to measure time varies depending on the mobile phone transmission technology.

B.5.1 Enhanced Observed Time Difference (E-OTD)

There are both TOA and TDOA systems falling under the umbrella of the term E-OTD [89]. These are location systems for GSM-based mobile phones.

The TDOA version functions as follows. Each base transceiver station (BTS) transmits messages. An additional location measurement unit (LMU) at a known location is used to determine the real-time difference between messages from separate BTSs. The LMU sends these differences to the mobile station (MS), which subtracts the differences reported by the LMU with the measured differences to compute TDOA and solve the hyperbolic location problem.

The TOA version is only slightly different. The LMU reports absolute times to the mobile unit instead of differences. Since the locations of the LMU and BTSs are known, the system can solve for the BTS send times as they would be measured with the LMU clock. The MS solves for its location (x,y) and its clock offset from the LMU using trilateration.

B.5.2 Advanced Forward Link Trilateration (A-FLT)

As with E-OTD, there are TOA and TDOA versions of A-FLT [89]. For CDMA (IS-95) phone systems, which have built-in synchronization, the process of measuring times is simpler than for GSM. In the TDOA version, the MS computes TDOA between the serving pilot and a neighboring pilot and solves for location using hyperbolic location. In the TOA version, the MS exchanges timestamped messages with each BTS. These message provide the distance between the MS and each BTS, so trilateration is used to compute location.

B.5.3 GSM Uplink Time of Arrival (UL-TOA)

The GSM UL-TOA system [89], despite its name, is based on hyperbolic location and not trilateration. The network locates an MS without aid from the MS.

The outline of steps in UL-TOA is as follows. The MS transmits access bursts for asynchronous handover. LMUs timestamp the arrival of the access bursts. The difference in timestamps is used to compute TDOA for hyperbolic location.

B.6 Combined RF and Ultrasound

TOA Distances can be measured by exploiting the large difference in speeds for radio $2.998 * 10^8$m/s versus ultrasound, approximately $3.30 * 10^2$m/s in air. The relatively slow speed of ultrasound significantly relaxes the clock accuracy and synchronization requirements of the location system. Ultrasound, however, is very sensitive to multipath effects and obstructions, and requires many nodes to maintain line-of-sight conditions.

B.6.1 Cricket

The Cricket location system [67, 73] is a trilateration system based on measuring the time difference between the arrival of simultaneous radio and ultrasound signals. Fixed-location beacons (anchor nodes) transmit concurrent RF and ultrasound signals, scheduled in a randomized fashion. Passive listeners timestamp the arrivals of the RF and ultrasound signals. The difference between these timestamps gives the TOA distance from the listener to the beacon. This allows the listener to compute its location given the beacon locations on its own.

B.6.2 Active Bat

In the Active Bat system [41], fixed receivers (anchor nodes) listen to ultrasound signals from mobile Bat units. A base station concurrently alerts receivers to zero their clocks and triggers an active Bat to transmit an ultrasound signal. The base station must schedule Bat units to time slots to eliminate interference between units. Each receiver records the time the ultrasound signal from a Bat unit arrives. This is the effectively the propagation time from the Bat to the receiver because the RF propagation time from the base station to the Bat is negligible compared to the propagation time of the ultrasound signal from the Bat to the receiver. Propagation times translate directly into distances based on the speed of ultrasound, and multiple distances lead to an instantiation of the trilateration problem.

Appendix C

Sensor Network Localization Problem

In chapters 1-6, we have examined only simple location problems where we locate a single node using anchor nodes. For TOA information we use trilateration, and for TDOA information we use hyperbolic location. To locate multiple nodes, these techniques treat each node independently. For TDOA, this simple approach uses all the available information. For TOA, mobile nodes can measure distances with other mobile nodes in addition to the anchor nodes. The solution for one mobile node location is dependent upon the solutions for other mobile nodes. This interdependency is addressed in the sensor network localization problem.

Sensor networks have been made feasible by the availability of relatively inexpensive sensor and communication hardware. Broadly, sensor networks cover a spatial area to collect information, which can be anything from temperature to humidity to animal movement. Sensor network nodes are generally autonomous and characterized by inexpensive hardware with computational and power constraints. Information must be routed through the sensor network to a collection point, with paths changing depending on node availability due to failures or power-saving features. Sensor networks are a very active research area, of which the study of location is but one part.

Informally, the sensor network localization problem is as follows.

Input : set of wireless nodes with noisy internode distances for some pairs

Output : Locations of all wireless nodes in $\mathbb{R}^d$.

A full mathematical treatment is given by Dattorro [27]. There are no known sources that include TDOA information; research has focused on the TOA case.

There are many variants of the sensor network localization problem. For a survey, see [84]. The main features are whether anchor nodes are available and whether ranges between nodes can be measured. Variants are either anchor-based or anchor-free, and either range-based or range-free.

Anchor nodes have known location either from manual input or external sources such as GPS and can constrain the solution to an absolute coordinate system. Without anchors, only relative locations can be determined; the solution may be rotated or reflected. The additional costs associated with anchor nodes mean they are relatively scarce. Both anchor-based and anchor-free variants may be used in conjunction with PinPoint TOA.

The capability to measure distances between nodes means these ranges are available for location. Without distances, only network connectivity is available, which generates less accurate solutions. The range-based methods are applicable to PinPoint TOA.

Sensor network localization is a non-convex optimization problem. The problem is generally attacked in two separate phases, initial placement and refinement. Techniques for refinement such as gradient descent or spring-mass localization have difficulties with local minima. The goal of the initial placement phase is a localization without local minima that is suitable for refinement.

C.1 Graph Rigidity Theory and Unique Localizability

The sensor network localization problem is difficult — Saxe showed the graph realization problem is NP-Hard [71]. Theoretical approaches to this difficult problem start with the case of no noise. When noise is present, optimization problem behavior may include local minima.

A fundamental question is whether a set of nodes and internode distances generate a unique, non-trivial solution. Graph rigidity theory has been applied to examine this problem when no noise is present [35]. This result establishes necessary conditions for a unique solution in terms of global rigidity.

The existence of a unique solution, however, does not mean there is an algorithm available to find this solution. The property of unique d-localizability [74] examines sufficient conditions for a unique and efficiently realizable solution. Using unique d-localizability, the sensor network problem can be solved using semi-definite programming.

To combine d-localizability and global rigidity, the concept of universal rigidity [90] was introduced. Universal rigidity is more restrictive than global rigidity, but retains the sufficient conditions to compute a solution. Research is ongoing to find the necessary and sufficient conditions for the existence of a unique and efficiently realizable solution.

C.2 Semidefinite Programming

Semidefinite programming (SDP) is a branch of optimization. A semidefinite nxn matrix S satisfies

$$\forall v \in \mathbb{R}^n, v^T S v \geq 0 \tag{C.1}$$

Semidefinite programming methods [12, 13, 16, 74] use a relaxation to produce a convex problem solvable as a semidefinite programming problem from the network localization problem.

The solution to the SDP problem may be high dimensional. Rank reduction must be applied to find a solution in the proper dimension space, which is likely either two or three dimensions. One option is to project the solution from the higher dimension space into the lower dimension space. This projection results in a suboptimal solution that can be refined using techniques such as gradient descent.

C.3 Spring-Mass Localization

An alternate approach to sensor network localization refinement described by Howard, Mataric, and Sukhatme models the network using masses and mechanical springs [44]. Each distance measurement is a spring. Springs apply force on the nodes to move them when the distance between nodes differs from the measured distance. In the basic spring model the force is proportional to the distance x from the equilibrium point.

$$F = -kx \tag{C.2}$$

The system estimates the nodes' location by finding the geometry with minimum energy.

Spring-mass localization techniques are distributed. Each node can update its estimated position using its current position, the positions of its neighbors, and its measured distances.

Priyantha, Balakrishnan, Demaine, and Teller studied fold-free initial configurations [68]. Fold-free configurations are designed to eliminate local minima during the spring optimization process. Dabek, Cox, Kaashoek, and Morris explored an adaptation of spring-mass localization for network coordinates in the Vivaldi system [26].

In Vivaldi, each node estimates its error

$$\text{error} = \sqrt{\sum (d_{\text{measured}} - d_{\text{estimated}})^2} \tag{C.3}$$

Adjustment steps are modified by

$$s = \frac{\text{local error}}{\text{local error} + \text{remote error}} \tag{C.4}$$

Appendix D

802.11 Implementation Considerations

This appendix addresses practical issues involved with implementing PinPoint on real 802.11 devices. The constraints imposed by available hardware and software features dictates system design. Features may also be undocumented or erroneously documented.

Many network applications such as Wireshark timestamp at the operating system level. Applications make system calls to read the CPU clock, and the resulting timestamps are subject to increased variability due to time sharing. As discussed in Chapter 3, this is insufficiently accurate. We need the hardware to support timestamping at a low level, and we require driver support to access these timestamps.

Windows-based computer systems support relatively primitive timestamping for most drivers available. Timestamping is often done by applications rather than using the 802.11 wireless card clock. As a result of interrupt dependent timings, the variability of timestamps increases unacceptably.

The only known drivers for 802.11 a/b/g cards that support the timestamping precision requirement are the madwifi-ng and ath5k drivers for GNU/Linux.

Driver	MAC ts	Precision	Sent ts	Received ts
NDIS				x
airpcap	x			x
madwifi-ng	x	x	x	x
ath5k	x	x		x

Table D.1: Supported Features for Selected Drivers

The 802.11 standard [2] specifies that timestamps must be available to a precision of 1 μs with accuracy of 100 microseconds per second (0.01%). Since a timing error of 1 μs will produce a distance error of more than 300 m, better than 1 μs precision is obviously desirable.

For timestamping sent messages, it is sufficient to send beacons. Beacons contain send times in the body of the frame. AP mode support is sufficient.

D.1 Drivers

Standard Windows drivers do not support MAC clock timestamping. The Windows Network Driver Interface Specification (NDIS) supports timestamping with a precision of 100 ns but the timestamps are system clock based and subject to the interrupt latency issue. Any application using the NDIS for time information will not be suitable for location determination.

Many GNU/Linux wireless network adapter drivers are not open source. The ndiswrapper program allows GNU/Linux use of Windows NDIS drivers, which are subject to the same interrupt latency issues associated with Windows.

D.1.1 Madwifi-ng

The madwifi-ng driver for GNU/Linux is the only known driver currently supporting all requirements for laptop compatible hardware. It requires a PCI-type interface such as PC card (also known as PCMCIA) or PCI-express. It does not support USB network cards.

The madwifi-ng driver is not sufficiently compatible with all recent kernel versions. The demonstration PinPoint TOA and TDOA system is operational with Suse 10.3 running the 2.6.22 Linux kernel. It is not operational with Suse 11.0 running the 2.6.25 Linux kernel. The cause of this incompatibility is not known.

To implement the reference clock with 25 ns precision, the madwifi-ng driver was modified to add an ioctl call setting the reference clock register. Since the register access is a simple write, the `ath_info` program, recently spun off from the madwifi project, should be able to replace the madwifi-ng driver changes. The `ath_info` program is capable of reading and writing to Atheros card registers.

Upon receipt of a packet, the hardware fills in the receive descriptor, which contains a 15 bit timestamp field. The 15 bit timestamp is extended to a 64 bit timestamp in the driver during interrupt handling by reading the MAC clock. A rollover of the 15 bit timestamp is detected if the receive descriptor timestamp is greater than the 15 least significant bits of the 64 bit MAC clock timestamp. One rollover of the 15 bit timestamp can be corrected in making this extension. Any further rollovers will result in erroneous 64 bit times.

There are four different modes supported by the madwifi-ng driver.

ad-hoc supports communication without an AP infrastructure

AP mode supports basic AP functionality such as beacons and association

monitor mode provides the ability to process packets outside the standard network stack, including management packets such as beacons, and packets from any source when in promiscuous mode. With optional radiotap headers, packet receive times are available.

station mode is the standard client mode of operation

Although it is possible to configure the driver to run in multiple modes simultaneously, there are significant problems in practice.

1. AP + station mode: Bringing up the station VAP causes the card to search for an BSS to associate, and it is not possible to fix a specific channel. This search will be over all 802.11 channels, and the AP VAP will not function during this period. This behavior makes it impossible to setup two or more laptops in AP+sta mode to both send beacons and communicate with each other. This problem is documented in two madwifi-ng tickets: http://madwifi-project.org/ticket/980 and http://madwifi-project.org/ticket/2019.

2. AP + monitor mode: Without a monitor mode VAP, the AP responds to pings. When monitor mode is enabled, the AP does not respond to pings. This problem is documented in a madwifi-ng ticket: http://madwifi-project.org/ticket/1955.

3. Ad-hoc + monitor mode: Ad-hoc mode does not function properly with monitor mode.

Furthermore, the madwifi-ng driver is no longer under active development. The above problems are not expected to be resolved with the madwifi driver, though they may be resolved in the future with the ath5k driver.

Driver stability is another significant problem. A card reset changes the reference clock speed back to 1 MHz and resets the clock counter to zero. One cause of resets in madwifi-ng is three consecutive failures to send beacons. With multiple nodes running at the regular beacon interval (bintval=100) with the highest reference clock speed of 40 MHz for 400 beacons per second, card resets are a frequent occurrence, particularly when other network traffic is present.

PinPoint with the madwifi-ng driver has been most successful running cards in AP and monitor modes simultaneously with the maximum (bintval=1000) beacon interval value. At this higher beacon interval, fewer packets are sent by each node

(40 packets per second at 40 MHz), resulting in less contention for the channel and fewer missed beacons and card resets. A drawback of this configuration is that no data communication is possible over the madwifi-ng network card; only beacons can be sent between nodes. This is sufficient for timestamping; send times are included in the beacon payload, and receive times are recorded through monitor mode. A second network interface is necessary to communicate the collected timestamps from PinPoint nodes to another node or server.

D.1.2 ath5k

The open source ath5k driver is designed to replace the madwifi-ng driver, supporting the same physical hardware. Development work in the open source community is explicitly focused on improving the ath5k driver over the madwifi-ng driver. Prior to Nov. 2008, the madwifi-ng driver was partly based on a binary hardware abstraction layer (HAL). Since the source code to this binary HAL was not available, the open source community desired to develop a completely open source version.

As of Jan. 2009, the ath5k driver did not sufficiently support sending timestamped messages because AP mode was not found sufficiently functional and ad-hoc mode sends beacons only intermittently. As of kernel 2.6.34, the stability of the station-only mode was not good enough to warrant development over alternative platforms such as the RN-134.

D.2 RN-134

With embedded devices, the same timestamping capabilities are needed for a PinPoint system. As of Jan 2011, we have only identified the G2 Microsystems (acquired by Roving Networks in 2010) devices as supporting better than 1 μs timestamping. The Roving Networks RN-134 supports timestamping of all incoming packets with a 44

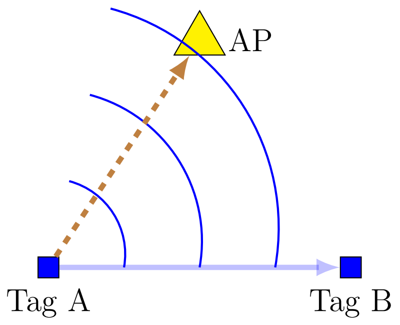


Figure D.1: UDP-based PinPoint for RN-134

MHz clock. There is also an undocumented feature that provides a timestamp for the last outgoing packet with the same 44 MHz clock simply by reading a 64-bit hardware register. If a packet is retransmitted, the retransmission and not the original packet is timestamped.

A different message passing design for the RN-134 is necessary because send times cannot be included inline as with beacons. Each node associates with an AP and sends a stream of UDP packets through the AP to a server. UDP transport was chosen over the more expensive TCP because timestamp information can be dropped or received out of order without significant performance loss. The 802.11 protocol also handles retransmissions of unacknowledged data packets. Each node listens in promiscuous mode to timestamp receive times for other nodes' packets, which have the AP as the destination MAC address.

Each packet contains:

1. sequence number n
2. send time of packet $n-1$
3. MAC addresses of nodes to listen to
4. last five receive timestamps for each MAC address

The received timestamp bias is dependent upon packet length. For this reason, it

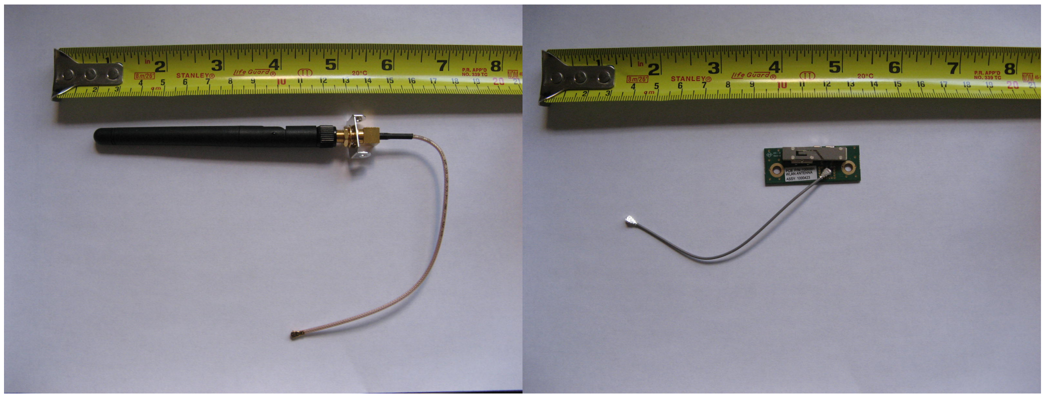

(a) Antenna 1 (b) Antenna 2

Figure D.2: External antennas

is important to fix packet lengths rather than leave them variable. For the demonstration system, the set of MAC addresses was fixed.

All RN-134 experiments covered in this dissertation used the minimum bit rate of 1 Mbps to maximize range. Even with this bit rate, the effective range of communication between nodes was only 20m.

To achieve even the 20m range, it was necessary to enclose the RN-134 within an aluminum case and ground both the antenna and battery to this case. Without this enclosure, the nodes were unable to communicate with each other beyond eight meters.

A further variable affecting system performance is antenna selection. The ceramic, onboard antennas for the RN-134 were found to have very poor performance for communication between RN-134 nodes. Communication between an AP and the RN-134 was not as poor. Two antennas with better performance are shown in Figure D.2. These are still under evaluation.

Appendix E

The Hyperbola and Hyperboloid

E.1 Hyperbola

The hyperbola is a basic two-dimensional conic section determined by two foci f_1, f_2 and a difference of distances $d(f_2, q) - d(f_1, q)$. The hyperbola is the locus of points that satisfy

$$|d(f_2, q) - d(f_1, q)| = 2b \tag{E.1}$$

Each hyperbola has two branches that satisfy the following equations:

$$d(f_2, q) - d(f_1, q) = 2b \tag{E.2}$$

$$d(f_2, q) - d(f_1, q) = -2b \tag{E.3}$$

Figure E.1 shows one branch of a hyperbola. The explicit equation for the hyperbola is the following equation, where $c^2 = a^2 + b^2$.

$$-\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \tag{E.4}$$

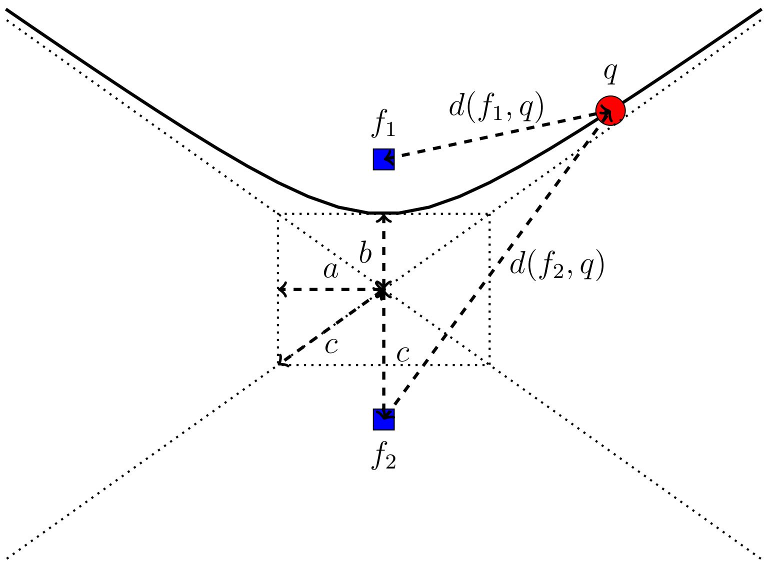


Figure E.1: Hyperbola Branch, $d(f_2, q) - d(f_1, q) > 0$

The conjugate hyperbola, which opens in the x direction, is given by

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1 \tag{E.5}$$

The positions $f_1 = (x_1, y_1)$, $f_2 = (x_2, y_2)$ and the TDOA quantity $d(f_2, q) - d(f_1, q)$ define a hyperbola with the following parameters:

$$c = \frac{1}{2} d(f_2, f_1) \tag{E.6}$$

$$b = \frac{1}{2}[d(f_2, q) - d(f_1, q)] \tag{E.7}$$

$$a = \sqrt{c^2 - b^2} \tag{E.8}$$

$$x_{center} = \frac{x_1 + x_2}{2} \tag{E.9}$$

$$y_{center} = \frac{y_1 + y_2}{2} \tag{E.10}$$

$$\theta = \begin{cases} 0 & \text{if } y_1 = y_2, \quad x_1 > x_2 \\ 180 & \text{if } y_1 = y_2, \quad x_1 < x_2 \\ \arccos\left(\frac{y_1 - y_2}{d(f_1, f_2)}\right) & \text{if } y_1 \neq y_2, \quad x_1 \leq x_2 \\ -\arccos\left(\frac{y_1 - y_2}{d(f_1, f_2)}\right) & \text{if } y_1 \neq y_2, \quad x_1 > x_2 \end{cases} \tag{E.11}$$

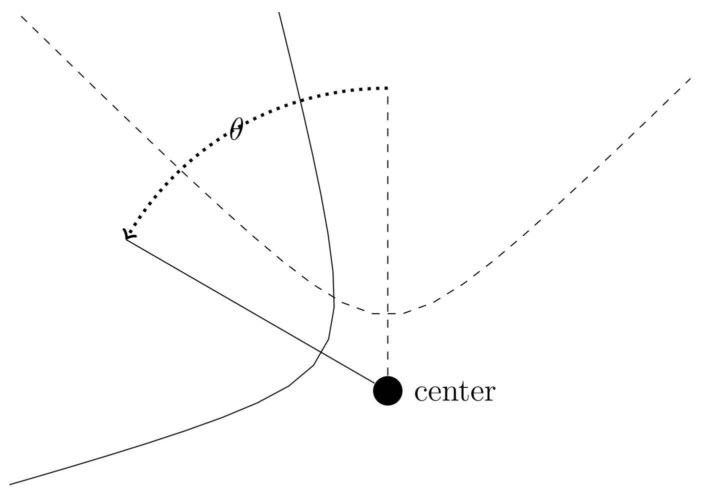


Figure E.2: Rotation by $\theta = 60$ degrees

TDOA measurements provide $d(f_2, q) - d(f_1, q)$. For a to be real, $|d(f_2, q) - d(f_1, q)| \leq d(f_2, f_1)$. If this inequality is violated, measurement errors must be present. For the TDOA measurement to be meaningful, $f_1 \neq f_2$. Otherwise $d(f_1, q) = d(f_2, q)$ for all points in space.

The rotation convention takes the y+ direction as $\theta = 0$. Rotation is about the hyperbola center in the counter-clockwise direction by the angle θ. This is illustrated in Figure E.2.

E.2 Hyperboloid

The hyperboloid is a three-dimensional version of the hyperbola. It is defined by foci and difference of distances as the hyperbola, $f_1, f_2, d(f_2, q) - d(f_1, q)$. The parameters a, b, c are identical for the hyperboloid. The center computation has the additional z

dimension and an additional rotation parameter ρ is required.

$$c = \frac{1}{2} d(f_2, f_1) \tag{E.12}$$

$$b = \frac{1}{2} [d(f_2, q) - d(f_1, q)] \tag{E.13}$$

$$a = \sqrt{c^2 - b^2} \tag{E.14}$$

$$x_{center} = \frac{x_1 + x_2}{2} \tag{E.15}$$

$$y_{center} = \frac{y_1 + y_2}{2} \tag{E.16}$$

$$z_{center} = \frac{z_1 + z_2}{2} \tag{E.17}$$

$$\theta = \begin{cases} 0 & \text{if } y_1 = y_2, \quad x_1 > x_2 \\ 180 & \text{if } y_1 = y_2, \quad x_1 < x_2 \\ \arccos\left(\frac{y_1 - y_2}{d(f_1, f_2)}\right) & \text{if } y_1 \neq y_2, \quad x_1 \leq x_2 \\ -\arccos\left(\frac{y_1 - y_2}{d(f_1, f_2)}\right) & \text{if } y_1 \neq y_2, \quad x_1 > x_2 \end{cases} \tag{E.18}$$

$$\rho = \begin{cases} 0 & \text{if } x_2 = x_1, z_2 = z_1 \\ \frac{\pi}{2} & \text{if } x_2 = x_1, z_2 \neq z_1 \\ \arctan(\frac{z_2 - z_1}{x_2 - x_1}) & \text{otherwise} \end{cases} \tag{E.19}$$

The equation for the hyperboloid is given by the following:

$$-\frac{x^2 + z^2}{a^2} + \frac{y^2}{b^2} = 1 \tag{E.20}$$

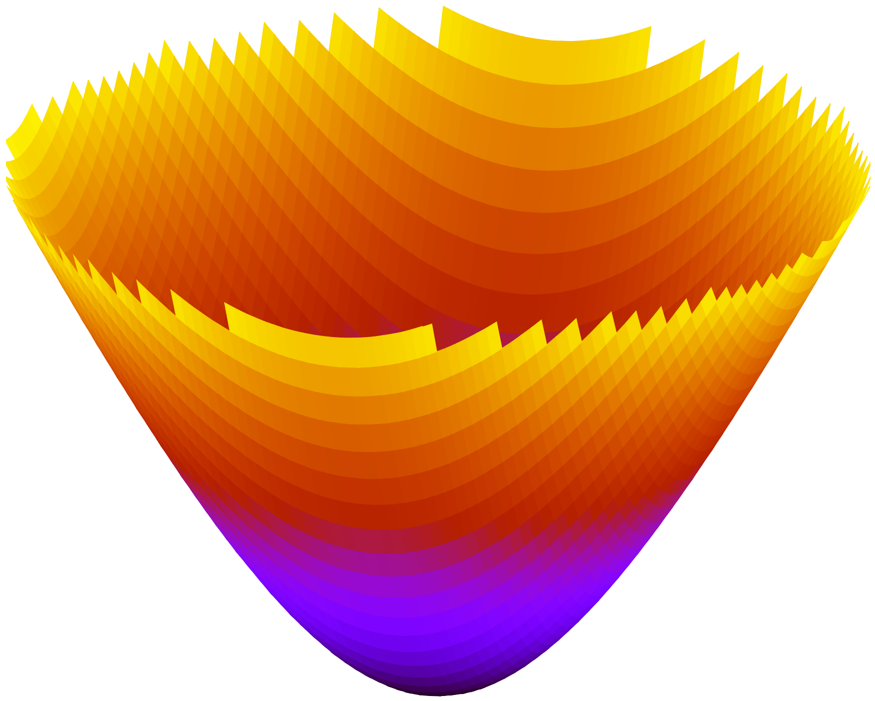


Figure E.3: Hyperboloid

Appendix F

Clock Oscillators

Modern clocks combine a frequency source with a counter [48] as shown in Figure F.1. The counter increments with each event from the frequency source. The stability of the frequency source determines clock accuracy. More stable sources naturally are more expensive. Research continues on developing frequency sources of increasing stability and decreasing cost.

For the international metric second standard, the frequency source is radiation resulting from transitions of cesium-133. Atomic clocks based on the cesium standard are too expensive for most applications, which use cheaper alternatives to trade accuracy against cost.

The basic frequency source in use is the quartz crystal oscillator (XO), which is manufactured to resonate at a particular frequency. Oscillations in the piezoelectric

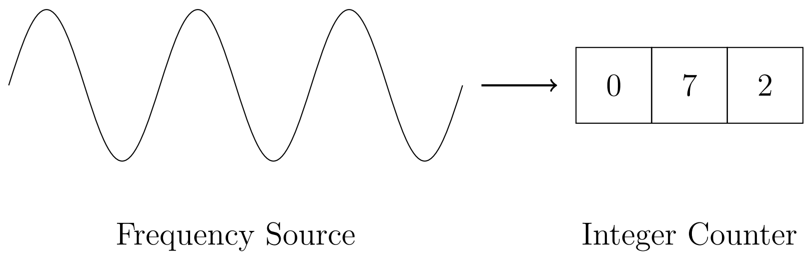


Figure F.1: Frequency Source Clock Model

Abbr	Accuracy	Cost ($)
VCXO	$[10^{-5}, 10^{-4}]$	< 5
TCXO	$2 * 10^{-6}$	$10 - 100$
OCXO	10^{-8}	$200 - 2000$
MCXO	$5 * 10^{-8}$	< 1000
RbXO	10^{-9}	10k
Cesium	$[10^{-12}, 10^{-11}]$	50k

Table F.1: Clock Oscillator Accuracy, Cost [79]

crystal generate changes in voltage that are measured as events for the clock counter. The frequency of oscillations, however, will vary due to factors such as temperature and crystal aging. Temperature is the main source of short-term frequency variation. Aging may occur over years, resulting in long-term frequency variation. There are several mechanisms to correct for this variation and improve accuracy at the price of increased cost, size, and power consumption.

- Voltage Control (VC) - The resonance frequency of the crystal can be altered over a small range by applying a voltage to the piezoelectric crystal.
- Temperature Compensation (TC) - Feedback from a temperature sensor is used to adjust clock frequency.
- Oven Control (OC) - A temperature controlled chamber is maintained for sensitive components.
- Microcomputer Compensation (MC) - Designed to be more accurate than TC without the power requirements of OC.
- Rubidium - A rubidium time source is periodically used to correct the crystal oscillator.

Due to manufacturing differences in the physical hardware and operational variables such as temperature, any two clocks are expected to have slightly differing frequencies. These differences in frequency will cause clocks to drift apart over time.

For wireless cards, the 802.11 standard [2] states that clocks should drift at no more than 100 parts per million (ppm). In practice, we have found most drift rates to be less than 10 ppm. To achieve this accuracy, it is sufficient for commodity products to use voltage controlled crystal oscillators (VCXO).

Bibliography

[1] 1588-2002 IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems.

[2] 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, 12 2007.

[3] FCC Consumer Facts Wireless 911 Services. http://www.fcc.gov/cgb/consumerfacts/wireless911srvc.html, March 2009.

[4] Madwifi project. http://madwifi-project.org/, March 2009.

[5] Android developer challenge gallery. http://code.google.com/android/adc/adc_gallery/, March 2011.

[6] DW Allan. Time and frequency (time-domain) characterization, estimation, and prediction of precision clocks and oscillators. *IEEE Transactions on Ultrasonics, Ferroelectrics and Frequency Control*, 34(6):647–654, 1987.

[7] D.W. Allan, National Bureau of Standards Boulder CO Time, and Frequency Division. Clock characterization tutorial. 1983.

[8] Isaac Arnsdorf. The museum is watching you. The Wall Street Journal, 18 August 2010.

[9] P. Bahl, R. Chandra, J. Padhye, L. Ravindranath, M. Singh, A. Wolman, and B. Zill. Enhancing the security of corporate Wi-Fi networks using DAIR. In *International Conference On Mobile Systems, Applications And Services: Proceedings of the 4 th international conference on Mobile systems, applications and services*, pages 1–14, 2006.

[10] P. Bahl and V. Padmanabhan. RADAR: An in-building RF-based user location and tracking system. In *IEEE INFOCOM*, volume 2, pages 775–784. Institute of Electrical Engineers, Inc (IEEE), 2000.

[11] R. Beyah, S. Kangude, G. Yu, B. Strickland, and J. Copeland. Rogue access point detection using temporal traffic characteristics. In *Global Telecommunications Conference, 2004. GLOBECOM'04. IEEE*, volume 4, 2004.

[12] P. Biswas, T.C. Lian, T.C. Wang, and Y. Ye. Semidefinite programming based algorithms for sensor network localization. *ACM Transactions on Sensor Networks (TOSN)*, 2(2):220, 2006.

[13] P. Biswas, TC Liang, KC Toh, Y. Ye, and TC Wang. Semidefinite programming approaches for sensor network localization with noisy distance measurements. *IEEE Transactions on Automation Science and Engineering*, 3(4):360–371, 2006.

[14] N. Bulusu, J. Heidemann, and D. Estrin. Gps-less low-cost outdoor localization for very small devices. *Personal Communications, IEEE*, 7(5):28 –34, oct 2000.

[15] J. Caffery Jr and GL Stuber. Subscriber location in CDMA cellular networks. *Vehicular Technology, IEEE Transactions on*, 47(2):406–416, 1998.

[16] M.W. Carter, H.H. Jin, M.A. Saunders, and Y. Ye. Spaseloc: An adaptive subproblem algorithm for scalable wireless sensor network localization. *SIAM Journal on Optimization*, 17(4):1102–1128, 2007.

[17] Vijay Chandrasekhar, Winston KG Seah, Yoo Sang Choo, and How Voon Ee. Localization in underwater sensor networks: survey and challenges. In *Proceedings of the 1st ACM international workshop on Underwater networks*, WUWNet '06, pages 33–40, New York, NY, USA, 2006. ACM.

[18] MK Chirumamilla and B. Ramamurthy. Agent based intrusion detection and response system for wireless LANs. In *Communications, 2003. ICC'03. IEEE International Conference on*, volume 1, 2003.

[19] S.C. Chou, W.T. Hsieh, F.L. Gandon, and N.M. Sadeh. Semantic web technologies for context-aware museum tour guide applications. 2005.

[20] M. Ciurana, F. Barcelo-Arroyo, and F. Izquierdo. A Ranging Method with IEEE 802.11 Data Frames for Indoor Localization. pages 2092–2096, 2007.

[21] M. Ciurana, F. Barcelo-Arroyo, and F. Izquierdo. A ranging system with ieee 802.11 data frames. In *Radio and Wireless Symposium, 2007 IEEE*, pages 133 –136, 2007.

[22] H.S. Cobb. *GPS pseudolites: Theory, design, and applications.* PhD thesis, Stanford University, 1997.

[23] I. Constandache, R.R. Choudhury, and I. Rhee. Towards mobile phone localization without war-driving. In *INFOCOM, 2010 Proceedings IEEE*, pages 1 –9, 2010.

[24] F. Cristian, H. Aghili, and R. Strong. Clock synchronization in the presence of omission and performance failures, and processor joins. 1986.

[25] F. Cristian and C. Fetzer. Probabilistic internal clock synchronization. In *Reliable Distributed Systems, 1994. Proceedings., 13th Symposium on*, pages 22–31, Oct 1994.

[26] F. Dabek, R. Cox, F. Kaashoek, and R. Morris. Vivaldi: A decentralized network coordinate system. In *Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 15–26. ACM, 2004.

[27] J. Dattorro. *Convex Optimization & Euclidean Distance Geometry*. Lulu. Com, 2006.

[28] M.H. DeGroot. Reaching a consensus. *Journal of the American Statistical Association*, 69(345):118–121, 1974.

[29] B. Denis, J. Pierrot, and C. Abou-Rjeily. Joint distributed synchronization and positioning in UWB ad hoc networks using TOA. *IEEE transactions on microwave theory and techniques*, 54(4):1896, 2006.

[30] Coast Guard Dept. of Transportation. *Loran-C User Handbook*, 1974.

[31] A.K. Dey. Understanding and using context. *Personal and ubiquitous computing*, 5(1):4–7, 2001.

[32] J. Duckworth, D. Cyganski, S. Makarov, W. Michalson, J. Orr, V. Amendolare, J. Coyne, H. Daempfling, D. Hubelbank, H. Parikh, et al. WPI precision personnel locator system: Evaluation by first responders. *Proceedings of ION GNSS,(Fort Worth, Texas)*, 2007.

[33] J.C. Eidson, M.C. Fischer, and J. White. IEEE-1588 standard for a precision clock synchronization protocol for networked measurement and control systems. In *34 th Annual Precise Time and Time Interval (PTTI) Meeting*, pages 243–254, 2002.

[34] J. Elson, L. Girod, and D. Estrin. Fine-grained network time synchronization using reference broadcasts. *ACM SIGOPS Operating Systems Review*, 36:147–163, 2002.

[35] T. Eren, OK Goldenberg, W. Whiteley, Y.R. Yang, A.S. Morse, BDO Anderson, and PN Belhumeur. Rigidity, computation, and randomization in network localization. In *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, volume 4, pages 2673–2684. IEEE, 2004.

[36] BT Fang. Simple solutions for hyperbolic and related position fixes. *Aerospace and Electronic Systems, IEEE Transactions on*, 26(5):748–753, 1990.

[37] Miroslav Fiedler. *Special matrices and their applications in numerical mathematics*. Martinus Nijhoff Publishers, 1986.

[38] Stuart A. Golden and Steve S. Bateman. Sensor measurements for wi-fi location with emphasis on time-of-arrival ranging. *IEEE Transactions on Mobile Computing*, 6(10):1185–1198, 2007.

[39] M.S. Grewal, L.R. Weill, A.P. Andrews, and J. Wiley. *Global positioning systems, inertial navigation, and integration, 2nd edition*. Wiley New York, 2007.

[40] D. Halperin, W. Hu, A. Sheth, and D. Wetherall. 802.11 with multiple antennas for dummies. *ACM SIGCOMM Computer Communication Review*, 40(1):19–25, 2010.

[41] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster. The anatomy of a context-aware application. *Wireless Networks*, 8(2):187–197, 2002.

[42] C. Hoene and J. Willmann. Four-way TOA and software-based trilateration of IEEE 802.11 devices. *Personal, Indoor and Mobile Radio Communications,*

2008. PIMRC 2008. IEEE 19th International Symposium on, pages 1–6, Sept. 2008.

[43] Y.W. Hong and A. Scaglione. A scalable synchronization protocol for large scale sensor networks and its applications. *IEEE Journal on Selected Areas in Communications*, 23(5):1085–1099, 2005.

[44] A. Howard, M.J. Mataric, and G. Sukhatme. Relaxation on a mesh: a formalism for generalized localization. In *IEEE/RSJ International conference on intelligent robots and systems*, volume 2, pages 1055–1060. Citeseer, 2001.

[45] S. Jana and S.K. Kasera. On fast and accurate detection of unauthorized wireless access points using clock skews. In *Proceedings of the 14th ACM international conference on Mobile computing and networking*, pages 104–115. ACM New York, NY, USA, 2008.

[46] Yunye Jin, M. Motani, Wee-Seng Soh, and Juanjuan Zhang. Sparsetrack: Enhancing indoor pedestrian tracking with sparse infrastructure support. In *INFOCOM, 2010 Proceedings IEEE*, pages 1 –9, 2010.

[47] C. Justice, N. Mason, D. Taggart, R.L. Sydnor, United States Naval Observatory, National Aeronautics, Space Administration, and United States. Loran-C Time Management. 1994.

[48] G. Kamas and MA Lombardi. *Time and frequency users manual NIST Special publication 559*, 1990.

[49] T. Kleine-Ostmann and AE Bell. A data fusion architecture for enhanced position estimation inwireless networks. *IEEE communications letters*, 5(8):343–345, 2001.

[50] R. Klukas and M. Fattouche. Line-of-sight angle of arrival estimation in the outdoor multipath environment. *Vehicular Technology, IEEE Transactions on*, 47(1):342–351, Feb 1998.

[51] E.A. LeMaster. *Self-Calibrating Pseudolite Arrays: Theory and Experiment.* PhD thesis, Stanford University and Dept. of Aeronautics and Astronautics, 2002.

[52] E.A. LeMaster and S.M. Rock. Self-Calibration of Pseudolite Arrays Using Self-Differencing Transceivers. In *Proceedings of the Institute of Navigation GPS-99 Conference, Nashville, TN*, pages 1549–1558, 1999.

[53] A.R. Lopez. Gps landing system reference antenna. *Antennas and Propagation Magazine, IEEE*, 52(1):104 –113, Feb 2010.

[54] 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, 2007.

[55] D. Lucarelli and I.J. Wang. Decentralized synchronization protocols with nearest neighbor communication. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 62–68. ACM New York, NY, USA, 2004.

[56] H.J. Luinge. *Inertial sensing of human movement.* PhD thesis, Twente University, 2002.

[57] M. Mah, N. Gupta, and A. Agrawala. Pinpoint time difference of arrival for unsynchronized 802.11 wireless cards. In *INFOCOM, 2010 Proceedings IEEE*, pages 1 –5, 2010.

[58] C.D. Mano, A. Blaich, Q. Liao, Y. Jiang, D.A. Cieslak, D.C. Salyers, and A. Striegel. RIPPS: Rogue identifying packet payload slicer detecting unauthorized wireless hosts through network traffic conditioning. *ACM Transactions on Information and System Security-TISSEC*, 11(2), 2008.

[59] D.E. Manolakis. Efficient solution and performance analysis of 3-D position estimation by trilateration. *Aerospace and Electronic Systems, IEEE Transactions on*, 32(4):1239–1248, 1996.

[60] D.D. McCrady, L. Doyle, H. Forstrom, T. Dempsey, and M. Martorana. Mobile ranging using low-accuracy clocks. *Microwave Theory and Techniques, IEEE Transactions on*, 48(6):951–958, Jun 2000.

[61] D. Mills, D. Plonka, and J. Montgomery. Simple Network Time Protocol (SNTP) Version 4 for IPv4, IPv6 and OSI. RFC 4330, 2006.

[62] David Mills. Network time protocol version 4 reference and implementation guide electrical and computer engineering technical report 06-06-1. Technical report, University of Delaware, 2006.

[63] S.J. Murdoch. Hot or not: revealing hidden services by their clock skew. In *Proceedings of the 13th ACM conference on Computer and communications security*, pages 27–36. ACM New York, NY, USA, 2006.

[64] Dragoş Niculescu and Badri Nath. VOR base stations for indoor 802.11 positioning. In *MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking*, pages 58–69, New York, NY, USA, 2004. ACM.

[65] Alex Olshevsky and John N. Tsitsiklis. Convergence speed in distributed consensus and averaging. *SIAM Journal on Control and Optimization*, 48(1):33–55, 2009.

[66] S. Parichha and M. Molle. Localization and clock synchronization need similar hardware support in wireless LANs. In *Proceedings of the International Symposium on Precision Clock Synchronization for Measurement, Control and Communication*, pages 22–26, 2008.

[67] N.B. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. In *Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 32–43. ACM, 2000.

[68] Nissanka Priyantha, Hari Balakrishnan, Erik Demaine, and Seth Teller. Anchor-free distributed localization in sensor networks. 2003.

[69] W.G. Reid. Continuous Observation of Navstar Clock Offset from the DoD Master Clock Using Linked Common View-Time Transfer. In *Proceedings of the 28th Annual Precise Time and Time Interval (PTTI) Applications and Planning Meeting*, pages 3–5, 1996.

[70] R.I. Reza. Data fusion for improved TOA/TDOA position determination in wireless systems. Master's thesis, Virginia Polytechnic Institute and State University, 2000.

[71] J.B. Saxe. Embeddability of weighted graphs in k-space is strongly NP-hard. In *Proc. 17th Allerton Conf. Commun. Control Comput*, pages 480–489, 1979.

[72] B. Schilit, N. Adams, and R. Want. Context-aware computing applications. In *Mobile Computing Systems and Applications, 1994. Proceedings., Workshop on*, pages 85 –90, December 1994.

[73] Adam Smith, Hari Balakrishnan, Michel Goraczko, and Nissanka Bodhi Priyantha. Tracking Moving Devices with the Cricket Location System. In *2nd International Conference on Mobile Systems, Applications and Services (Mobisys 2004)*, Boston, MA, June 2004.

[74] A.M.C. So and Y. Ye. Theory of semidefinite programming for sensor network localization. *Mathematical Programming*, 109(2):367–384, 2007.

[75] M. Trinh. *Achieving global synchrony in a distributed system with free-running local clocks*. PhD thesis, University of Maryland, College Park, 2006.

[76] United States Department of Defense. *Global Positioning System Standard Position Service Performance Standard 4th edition*, September 2008.

[77] A. Urruela and J. Riba. Novel closed-form ML position estimator for hyperbolic location. 2, 2004.

[78] D. Veitch, J. Ridoux, and S. B. Korada. Robust synchronization of absolute and difference clocks over networks. *Networking, IEEE/ACM Transactions on*, 17(2):417–430, April 2009.

[79] J.R. Vig. Quartz crystal resonators and oscillators for frequency control and timing applications. a tutorial. Technical report, US Army Communications-Electronics Command, 2000.

[80] W. Wei, K. Suh, B. Wang, Y. Gu, J. Kurose, and D. Towsley. Passive online rogue access point detection using sequential hypothesis testing with TCP ACK-pairs. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 365–378. ACM New York, NY, USA, 2007.

[81] M.P. Wylie and J. Holtzman. The non-line of sight problem in mobile location estimation. In *Universal Personal Communications, 1996. Record., 1996 5th IEEE International Conference on*, volume 2, pages 827–831. IEEE, 1996.

[82] B. Yang and J. Scheuing. Cramer-Rao Bound and Optimum Sensor Array For Source Localization From Time Differences of Arrival. In *Acoustics, Speech, and*

Signal Processing, 2005. Proceedings.(ICASSP'05). IEEE International Conference on, volume 4, 2005.

[83] B. Yang and J. Scheuing. A theoretical analysis of 2d sensor arrays for TDOA based localization. In *2006 IEEE International Conference on Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings*, volume 4, 2006.

[84] A. Youssef. *Salam: a scalable anchor-free localization algorithm for wireless sensor networks*. PhD thesis, University of Maryland, College Park, 2006.

[85] M. Youssef, A. Youssef, C. Rieger, U. Shankar, and A. Agrawala. PinPoint: An Asynchronous Time-Based Location Determination System. In *Proceedings of the 4th international conference on Mobile systems, applications and services*, pages 165–176. ACM New York, NY, USA, 2006.

[86] Moustafa Youssef and Ashok Agrawala. The Horus WLAN location determination system. In *MobiSys '05: Proceedings of the 3rd international conference on Mobile systems, applications, and services*, pages 205–218, New York, NY, USA, 2005. ACM.

[87] Kegen Yu, Jean-philippe Montillet, Alberto Rabbachin, Paul Cheong, and Ian Oppermann. UWB Location and tracking for wireless embedded networks. *Signal Process.*, 86(9):2153–2171, 2006.

[88] X. Yun, E.R. Bachmann, H. Moore, and J. Calusdian. Self-contained position tracking of human movement using small inertial/magnetic sensor modules. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 2526–2533. IEEE, 2007.

[89] Y. Zhao. Mobile phone location determination and its impact on intelligent transportation systems. *IEEE Transactions on Intelligent Transportation Systems*, 1(1):55–64, 2000.

[90] Zhisu Zhu, A.M.-C. So, and Yinyu Ye. Universal rigidity: Towards accurate and efficient localization of wireless networks. In *INFOCOM, 2010 Proceedings IEEE*, pages 1 –9, 2010.