ABSTRACT

Title of thesis: CLOCK SYNCHRONIZATION
AND TARGET LOCATION DETERMINATION
IN WIRELESS NETWORKS

Sangjin Han, Master of Science, 2014

Thesis directed by: Professor Ashok Agrawala
Department of Computer Science

In a distributed system most nodes maintain a local oscillator to derive time information for synchronization with other nodes. A number of clock synchronization techniques have been presented in the literature (e.g. NTP, PTP) which rely on the exchange of messages among nodes to share timing information and to adjust the offset or skew of the clocks. We present an approach which does not require any adjustments to the local clocks, but relies on achieving synchronization through clock mapping functions which map the time at one node to the time at another node. We further show how closed paths in a graph of nodes can be used to estimate the synchronization tolerance. Through experimental results using piecewise linear functions, we demonstrate the feasibility of this approach and show how clock synchronization of better than 100 ps can be achieved in Wi-Fi environments.

Using the techniques and relying on the hardware of SMiLE3 board, we also demonstrate the ability to measure distance with accuracy of a few inches and thereby the localization to accuracy better than one foot. Results of experiments

conducted for localization are also presented.

CLOCK SYNCHRONIZATION AND TARGET LOCATION DETERMINATION IN WIRELESS NETWORKS

by

Sangjin Han

Thesis 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
Master of Science
2014

Advisory Committee:
Professor Ashok Agrawala, Chair/Advisor
Professor Charles Silio
Professor Bobby Bhattacharjee

Acknowledgments

I would like to thank my supervisor Professor Ashok Agrawala, for giving me the opportunity to be listed as one of his students, to work with him, and to write this thesis. I thank him for being a beacon to me whenever I was wandering around with hesitance to move on to the next steps, struggled to survive as an international student, and suffered from unhappy incidents. I could not have enjoyed sitting in both my desk and his office and solving problems without his guidance throughout the entire graduate life here.

I thank Professor Charles Silio and Bobby Bhattacharjee for participating in my thesis committee. Their valuable time and effort tailored my thesis to represent a complete set of current research.

I would like to thank Dr. Reinhard Exel for all the support. He has consistently been proactive, kind, and passionate to every and any topic of the research as well as the key person who let us use the boards in our experiments. Unconditional help was given to me especially for the period to learn how to operate and control those boards for the first time. Moreover, I could not have looked our research in diverse aspects without his keen ideas which are presented in our recent joint paperwork.

I enjoyed all the times that I have worked with Kellen Harwell. I thank him for carrying out all the different experiments at places, providing us custom-made enclosures, different types of antennas, and even miscellaneous but absolutely necessary parts. I remember the day we went outside to the park and carried out a large experiment that went for half a day which became the core results of my work.

Special thanks to all of former and current members in MIND lab, Matthew Mah, Ravindra Passan, Dongwoon Hahn, Shivsubramani Krishnamoorthy, James Lamptom, Nick Gramsky, Preeti Bhargava, Aditya Nakshathri, Akanksha Chauhan, Parth Shah, Wentao Hu, Sonam Sobti, Anilesh Shrivastava, Nishit Pawar, for letting me have good memories with them.

Table of Contents

Chapter 1: Introduction

A distributed system consists of a collection of nodes connected using a network which enables messages to be sent from any node to any other node. Sharing a common notion of time between two or multiple nodes is critical for temporal cooperation and managing cooperative tasks. There are two primary reasons for synchronizing clocks in a distributed system:

1. To initiate events or actions either precisely at the same time or with a well-defined temporal relationship between them, and

2. To timestamp events such that an ordering of the events can be maintained and delays between two events taking place at different nodes can be quantified.

Examples for the first category are, for instance, time-division multiple access (TDMA) channel access which uses scheduled transmissions in a shared medium to avoid multi-user interference. This is particularly critical for energy-sensitive wireless sensor networks (WSNs) where long sleep intervals and precisely synchronized wake-up intervals are required to minimize the on time and energy consumption of the radios. Similarly, time-triggered frequency hopping or security key renewal fall in the same category. Timestamping of distributed measurements falls in the second category. Timestamped sensor data are required for coherent data fusion, such as wireless or

acoustic range estimation [1, 2], audio/video synchronization [3], or the test of electric power lines [4]. As a result, clock synchronization has become a basic service of distributed systems and a wide range of different protocols and mechanisms have been developed to achieve and maintain synchronization.

Initially, each node has only knowledge about its own time by reading from the local clock. Clocks are two-part devices: they are composed of an oscillating device with a predefined interval and a counter which accumulates the time intervals at every clock tick. Clock synchronization protocols exchange messages with adjacent nodes in the network. These messages contain estimates about the local clock which are consequently used to establish a common notion of time between two or multiple nodes in the network. Commonly-used synchronization protocols are, for instance, the Network Time Protocol (NTP) or the Precision Time Protocol (PTP).

Clock synchronization is inherently volatile. Even if two clocks have been perfectly synchronized at one instant in time, they will drift apart as the oscillators' intervals are intrinsically stochastic and impaired by environmental factors such as temperature, pressure, vibration, and supply voltage. Thus, periodic resynchronization is required. As a consequence, perfect synchronization (i.e. zero clock offset between two or multiple clocks) is not feasible for an extended period of time. Two clocks are said to be synchronized, when the difference of their readings is below a defined value, termed *synchronization tolerance*. The accepted value of synchronization tolerance depends on the target application.

To synchronize clocks, the timestamps for events taking place at one node have to be conveyed to the other node(s). Each node records a timestamp of a selected

event using its local clock and shares the timestamp with other nodes by message exchange. In most synchronization protocols, the events of particular egress and ingress frames are used for timestamping. Based on multiple timestamps encapsulated in messages, nodes are able to estimate both clock skew and offset at periodic intervals. These estimates can either be applied to the clock hardware, e.g. by changing the offset or clock increment or frequency, or the hardware clock remains unaltered and synchronization is established by a mapping function accounting for skew and offset. Adjustments of the clock hardware are common in IEEE 1588 equipment, where either the local clock frequency is altered by means of a voltage-controlled oscillator (VCO) or the clock increment (per tick) is changed by means of an Adder-based clock (ABC) [5]. One particular advantage of this method is that timestamps can be taken at every clock tick without any skew/offset mapping functions. When using the second approach, the value of a free-running counter is translated using a mapping function creating a virtual clock. Such mapping functions are used for instance, by the `adjtimex()` function in the Linux Kernel. A key advantage of virtual clocks is that multiple virtual clocks can be set up for a single counter which is not easily implementable for the VCO or ABC approach.

The majority of work in this thesis has been published in the literature [6]. Here we have added more topics such as implementation issues and results from location determination techniques. Accordingly, the outline of thesis is as follows.

In Chap. 2, we formulate the problem by defining the mapping functions. One node can define a mapping function with the reference time. With multiple nodes in a network, mapping functions from one node to any other nodes can be defined.

Mapping functions for synchronization in a network and its validity are discussed.

In Chap. 3, we discuss how to estimate those mapping functions. In consideration of practical clocks, we propose to use piecewise linear approximation with intervals which are agreeable and implementable, but not necessarily fixed.

In Chap. 4, given that FPGA devices have the capability of 802.11b physical layer, we explain experimental setups and results. Indoor experiment with 3 nodes in a network is first described, and outdoor experiment with 5 nodes follows. A method to assess the synchronization tolerance is discussed and results using the method are presented.

In Chap. 5, we study two cases of location determination based on the assumption that the network is synchronized by the algorithm we have discussed so far. The first case assumes that each pair of two nodes can compute Time-of-Arrival(ToA) based distance, and the second case assumes that Time-Difference-of-Arrival(TDoA) is considered instead.

Finally, in Chap. 6, concluding remarks are given with future works.

Chapter 2: Problem Formulation

The approach taken here is similar to that of PinPoint Technology [7–9]. Let the two nodes, a and b, have independent local clocks (the clock at node a will be referred as clock a). Let t_a and t_b be the clock readings at each of these nodes as recorded by their local clocks. Clearly

$$0 \leq t_a, t_b < \infty. \tag{2.1}$$

Note that both t_a and t_b are strictly monotonically increasing functions of the reference time t. Normally we expect all three of them to be very close to each other. The time recorded by clock a is related to the reference time by a function

$$\phi_a(t) = t_a \tag{2.2}$$

This function exists for all clocks. As (2.2) is a bijective function, an inverse function which maps the reference time to the time at clock a exists,

$$\psi_a(t_a) = t. \tag{2.3}$$

$$t_a = \phi_a[t] = \phi_a[\psi_a(t_a)] \tag{2.4}$$

$$t = \psi_a[\phi_a(t)]. \tag{2.5}$$

In order to determine the time t_a at node a when it is t_b at node b we defined a function

$$\phi_{ab}(t_b) = t_a, \tag{2.6}$$

which can be written as

$$\phi_{ab}(t_b) = \phi_a\left[\psi_b(t_b)\right] = \phi_a\left[t\right] = t_a. \tag{2.7}$$

The inverse function for this mapping is defined as

$$\psi_{ab}(t_a) = t_b. \tag{2.8}$$

Both these functions are defined for all values of time at both the nodes.

$$t_a = \phi_{ab}(t_b) = \phi_{ab}\left[\psi_{ab}(t_a)\right]. \tag{2.9}$$

As (2.9) shows ϕ_{ab} and ψ_{ab} are inverse functions. Further we note that

$$\phi_{ab}(t_b) = \psi_{ba}(t_b). \tag{2.10}$$

These relationships are always true even though the mapping functions ϕ and ψ may not be known precisely.

If we have N independent clocks, the same formulation can be extended to all the clocks at these nodes. For each clock we can define $\phi_i(t)$ and $\psi_i(t_i)$ for all $i, j = 1, \ldots, N$ as well as ϕ_{ij} and ψ_{ij} for all $i, j = 1, \ldots, N$. Note that $\phi_{ii}(t_i) = t_i$ and $\psi_{ii}(t_i) = t_i$. We can treat the arrangement of N clocks as a fully connected graph with the mapping functions for the edge between node i and node j be ϕ_{ij} and ψ_{ij}. In this graph the mapping of time from any node i to any node j can be carried out directly or following any path that begins with node i and ends in node

j. The fact that the result of the mapping will be the same irrespective of the path followed is established in Theorem 2.1.

Theorem 2.1. *Given a set of N clocks, the mapping functions ϕ_{ij} and ψ_{ij} $i, j = 1, \ldots, N$, the graph defined by the clocks as the nodes and the edges for which the mapping functions have been defined, and a time instant τ at an arbitrary node k, if we map this time instant going from node to node following an arbitrary path beginning with node k and ending at node k, we will get the same time instant τ.*

Proof. The proof is based on the decomposability of the mapping functions ϕ_{ij} according to (2.7). Let us consider the mapping going across two steps, from clock l to n via m. Let the starting time at node l for this mapping be t_l. Then the mapped time at node m, t_m is

$$t_l = \phi_{lm}(t_m) = \phi_l \left[\psi_m(t_m)\right]. \tag{2.11}$$

And the mapped time at node n, t_n is

$$t_m = \phi_{mn}(t_n) = \phi_m \left[\psi_n(t_n)\right]. \tag{2.12}$$

Combining these two we get

$$\begin{aligned} t_l &= \phi_l \left[\psi_m(t_m)\right] = \phi_l \left[\psi_m(\phi_m \left[\psi_n(t_n)\right])\right] \\ &= \phi_l \left[\psi_n(t_n))\right] = \phi_{ln}(t_n). \end{aligned} \tag{2.13}$$

Therefore, going through two steps of mapping has same result as mapping in a single step. This process can be generalized to any number of steps. When the path closes on the starting node (2.5) assures that we will get the same time instant that we started with. □

Note that the existence of the mapping functions is a direct consequence of the existence of the reference time to which the time at any clock i, t_i can be mapped through the functions ϕ_i and ψ_i. We only require that these functions exist and do not need to know them as long as we know ϕ_{ij} and ψ_{ij}, $i, j = 1, \ldots, N$. If all the mapping functions are known, the time at any clock can be mapped to the time at any other clock accurately. In this situation the two goals of synchronization identified in Chap. 1 can be met easily. For carrying out a coordinated action the time can be specified for any clock and all nodes participating in that action can infer from that the local time at which an event has to be carried out. Similarly, for establishing global ordering the time of all events at various nodes the local time can be mapped to the time at any one selected node and the events ordered according to that time.

When we know the functions ϕ_{ij} and ψ_{ij}, $i, j = 1, \ldots, N$ at any given time precisely, the synchronization of the clocks is perfect. In practice, the timestamps are derived from physical clocks and exchanged among the nodes using a communication scheme. It is well known that the oscillators used in physical clocks are not perfectly stable and have frequency fluctuations. The use of actual clocks which have a finite tick time imposes a precision limit. Additional sources of noise are introduced in the egress and ingress timestamping, sending, propagating, and receiving. Therefore, while the precise mapping functions are not known, we can develop estimates of these functions empirically. As the estimates of the mapping functions are likely to yield some imprecision, their use will result in finite, non-zero synchronization tolerance. One way to estimate the synchronization tolerance is to use the estimated

functions to map a time instant along various paths in a connected network starting and finishing at the same node. The error measured with this procedure will be the sum of the errors along the path and knowing the number of steps along the path we can estimate the tolerance of one step mapping from any node to any other node.

Chapter 3: Estimating the Mapping Functions

In a distributed system with N nodes one way of establishing the mapping functions is to send and receive messages timestamped by the local clocks. The delay from node a to node b consists of three components, the sending delay s_a, the propagation delay p_{ab}, and the receiving delay r_b. The sending delay s_a refers to the time delay between the instant of generating the send timestamp and actually transmitting the message, caused by the transmit circuitry. Note that we are using the same definition for send time as used in IEEE 1588 [10] which is: When the start-of-frame delimiter passes the reference plane (network connector). The propagation delay p_{ab} is the time it takes the electromagnetic signal to propagate from the sending node a to the receiving node b. The delay r_b is again caused by the internal processing delay of the receiver. Clearly, sending and receiving delays are likely to depend on the specific devices at each node. When identical devices are used it is reasonable to assume that their sending delays are the same and receiving delays are the same at each node and that the sending delay may be different from the receiving delay. The propagation delay will depend on the physical layout, the distance between the nodes, and the connectivity of the nodes. Figure 3.1 shows these timing relationships. The message sent from node a at $t_a(1)$, which is the

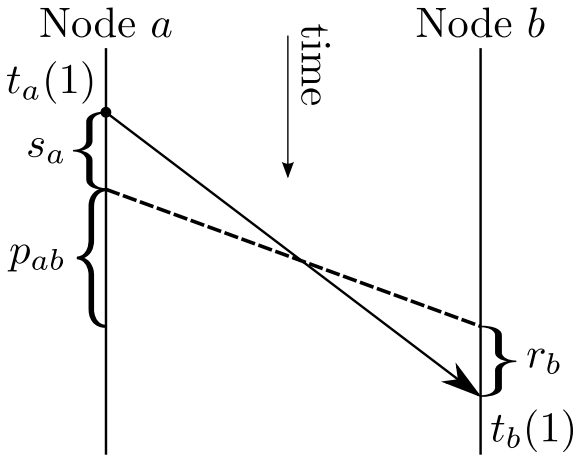

Figure 3.1: Transmission.

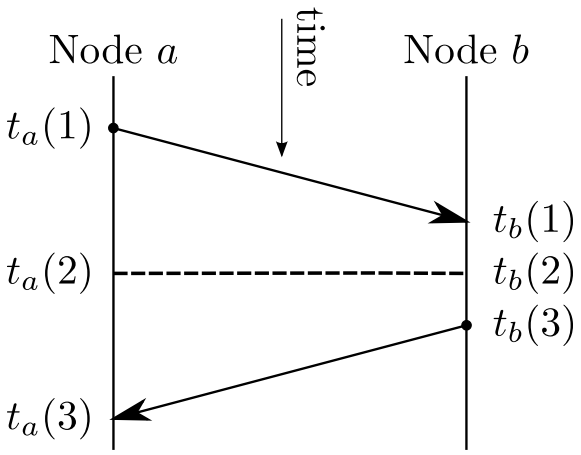

Figure 3.2: Round Trip timing.

timestamp recorded for this message, sustains a delay of s_a and actually propagates along the dashed line shown in the figure reaching node b and sustaining a receive delay of r_b, making node b record its arrival time as $t_b(1)$.

Figure 3.2 shows the transmission of two messages between the two nodes. In this figure we have suppressed the details of the sending and receiving times and included them in the transmission time. Node a sends a message at $t_a(1)$ which is received at Node b at $t_b(1)$ which then sends a reply at $t_b(3)$ received at Node

a at $t_a(3)$. Note that the arrow indicating the transfer of message takes the three components of time together. We further define

$$\begin{aligned} t_a(2) &= \left[t_a(1) + t_a(3)\right]/2, \\ t_b(2) &= \left[t_b(1) + t_b(3)\right]/2. \end{aligned} \tag{3.1}$$

If the two nodes have the same sending and receiving delays and the nodes have not moved during this interchange so that the propagation delays are the same then the two time instants, $t_a(2)$ and $t_b(2)$ represent the same time instant in reference time. Note that if this assumption is not true then the two times $t_a(2)$ and $t_b(2)$ will not be the same, resulting in a biased clock estimate. For the rest of this thesis we assume that this assumption holds. For the primary discussion here we assume that the nodes are using wireless communication and all nodes are within the listening range of all other nodes.

3.1 Using Simple Mapping Function

Let us start by considering linear mapping functions such that

$$\begin{aligned} t_a &= \phi_a(t) = \beta_a t + \alpha_a \\ t &= \psi_a(t_a) = \tfrac{1}{\beta_a}(t_a - \alpha_a) \end{aligned} \tag{3.2}$$

Here, β_a represents the drift rate of the clock a with respect to the reference time and α_a is the offset. For an ideal (reference) clock the drift rate β_a is unity. For most practical clocks the value of β_a is close to 1 but not precisely 1. Therefore we can define

$$\beta_a = 1 + \delta_a \tag{3.3}$$

Typical oscillators have frequency offsets δ_a in the range of $\pm 100 \times 10^{-6}$ or even smaller. Furthermore, β_a and δ_a are functions with bounded derivatives. The variance of the frequency change can be expressed by the Allan variance or its square root, the Allan deviation. Oscillators have Allan deviations between 1×10^{-6} to 1×10^{-15} depending on the type of oscillator (quartz, heated quartz, atomic) and the observation interval [11, 12].

We note that unless we observe the reference time (e.g. GPS-based clock), the determination of the true values of drift and offset for each node may not be possible in that we may only be able to determine the relative offset $(\alpha_a - {}^{\beta_a}/_{\beta_b}\alpha_b)$ and relative drift (β_a/β_b) or $(\delta_a - \delta_b)$.

Let us define $\delta_{ab} = \delta_a - \delta_b$. We note that as the drift rate of clocks is very close to 1 the error in this approximation is minimal.

In order to estimate the drift ratio we need to exchange another round of messages (Figure 3.3) sent at $t_a(5)$ and received at $t_b(5)$, and sent at $t_b(7)$ and received at $t_a(7)$ with $t_a(6)$ and $t_b(6)$ representing the midpoints as in (3.1). We can now obtain the drift ratio as:

$$\frac{\beta_b}{\beta_a} = \frac{t_b(6) - t_b(2)}{t_a(6) - t_a(2)} \simeq 1 + \delta_{ba}. \tag{3.4}$$

Once we have determined the drift ratio we can calculate the difference of the offsets by

$$\begin{aligned} (\alpha_a - \tfrac{\beta_a}{\beta_b}\alpha_b) &= t_a(2) - \tfrac{\beta_a}{\beta_b} t_b(2), \\ (\alpha_a - \delta_{ab}\alpha_b) &= t_a(2) - \delta_{ab} t_b(2). \end{aligned} \tag{3.5}$$

We observe that when $t = 0$, $t_a = \alpha_a$. Because of the different drift rates at different clocks the time scale at one clock is mapped to the time scale at the other clock by

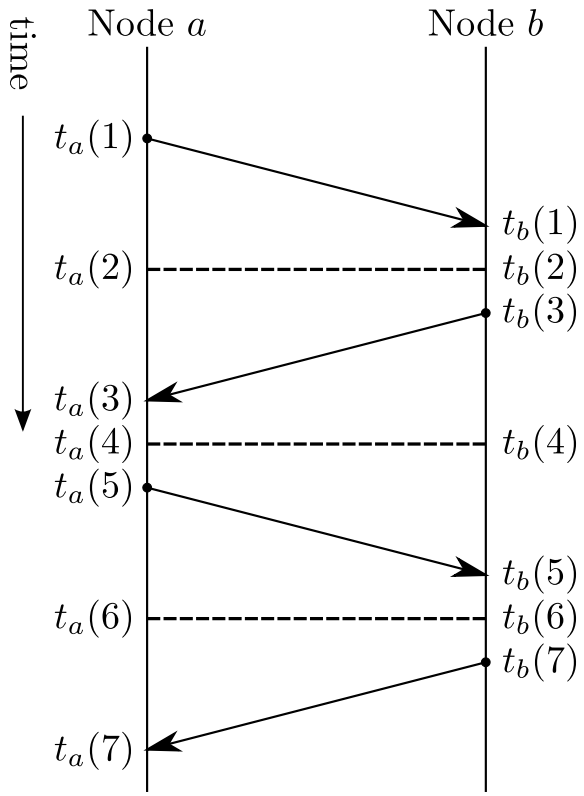

Figure 3.3: Two exchanges of round trip timing.

multiplication with the drift ratios. As a result, the determination of the difference of offsets can only be calculated by mapping the time scale at node b in which α_b is measured to the time scale at node a by multiplying it with the drift ratio ${}^{\beta_a}/_{\beta_b}$. Therefore, when the clocks are stable (constant drift rate and constant offset) we can determine the mapping function by exchanging a couple of messages and calculating

$$\begin{aligned} \phi_{ab}(t_b) &= \tfrac{\beta_a}{\beta_b} t_b + (\alpha_a - \tfrac{\beta_a}{\beta_b}\alpha_b) \text{ or} \\ \phi_{ab}(t_b) &= \delta_{ab} t_b + (\alpha_a - \delta_{ab}\alpha_b). \end{aligned} \tag{3.6}$$

3.2 Practical considerations

In order to record the time for an event which takes place at t_a we read the clock at node a, and record the time to be τ_a such that:

$$\tau_a = t_a + \epsilon_a \tag{3.7}$$

where ϵ_a is a term contributed by the measurement process including the discretization and jitter along with the electromagnetic noise distorting the signal during propagation. By adjusting the value of the offset α_a we can make sure that ϵ_a has zero mean value. For the discussion here we treat ϵ_a as iid (independent, identically distributed) with a normal distribution $\mathcal{N}(0, \sigma_a^2)$ with the variance σ_a^2. Therefore we have to treat τ_a as $\mathcal{N}(t_a, \sigma_a^2)$. As a consequence the drift ratios as calculated from the observed values of clock times using Eq. 3.4 follow a Cauchy Distribution [13]. However, as the drift ratios are close to one we can approximate the variance estimate of the drift ratio ${}^{\beta_a}/_{\beta_b}$ to be

$$\frac{2(\sigma_a^2 + \sigma_b^2)}{(t_a(6) - t_a(2))^2} \tag{3.8}$$

3.3 Piecewise linear mapping functions

The model presented in Section 3.1 does not perfectly match the complex behavior of real oscillators (e.g. see [12]). However, we can approximate the functions by a continuous, piecewise linear function. If the nodes exchange timestamped messages at regular intervals, we can estimate the parameters of the piecewise linear function between successive exchanges of messages. Such functions can then be used for mapping the time at one node to the time at another node to support the requirements of synchronization.

Let us consider the mapping function from clock a to clock b. This function is shown in the black line in Figure 3.4 and 3.5. If we experimentally observe the values of t_a and t_b at intervals which are $t_a(i+2)-t_a(i)$, for all i, apart, we can approximate the actual mapping function with the continuous, piecewise linear function shown in red. The resulting error caused by the mapping function is defined by the difference between the red and the black curve.

Let us take a detailed look at the approach taken in piecewise linear function estimation. The approach shown in Figure 3.4 is to obtain the piecewise linear function by observing the times at nodes a and b for four time instances A ($t_a(2)$, $t_b(2)$), B ($t_a(4)$, $t_b(4)$), C ($t_a(6)$,$t_b(6)$), and D ($t_a(8)$, $t_b(8)$). The actual mapping function is shown in black and has been exaggerated to improve the visibility. On observing B we estimate the mapping function from $t_a(2)$ to $t_a(4)$ to be the line segment AB. Similar process is repeated to get the line segments BC and CD. This we call the interpolated version of the estimate of the function as it only gives

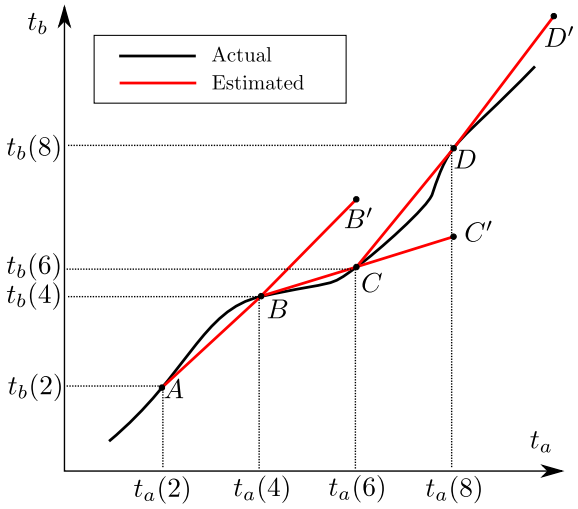

Figure 3.4: Piecewise linear mapping function

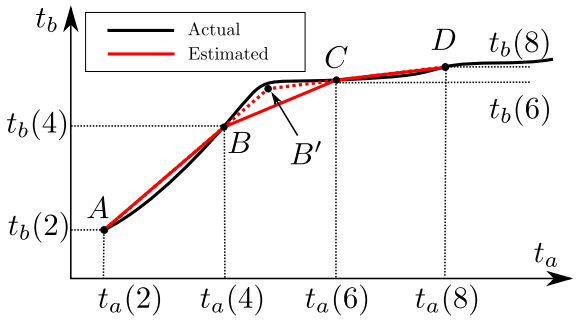

Figure 3.5: Piecewise linear mapping function with interpolation

the function for time instants which we have observed. As a consequence, at time $t_a(6)$ we can use the function to map any time on clock a for $t_a \leq t_a(6)$. If we want to estimate time beyond $t_a(6)$ we have to extrapolate the function for the duration till the next observation is made, i.e. $t_a(8)$ by using the line segment BC' which is obtained by extending the line BC. This we call the extrapolated version of the function. Clearly, as the actual function changes the errors in extrapolated function are likely to be larger than those in the interpolated version. Most of the time we expect the actual mapping function to be rather close to the straight line as the oscillator frequencies (reflected by the slope of the line) do not change rapidly. However in temperature controlled clocks periodic adjustments to the oscillator frequency are made changing the slope abruptly. Such changes are not likely to be at the time instants which are used for forming the mapping functions. One such situation is shown in Figure 3.5. The actual mapping function suddenly changes the slope at some time instant between B and C. Using the interpolation estimation function we will end up using the line segment BC which is likely to result in large errors for this segment. A way to avoid this is to observe the line segment after point C and note that there was a significant change in slope from AB to CD. In that case we find the intersection of the extensions of line AB and CD at point B' and use the line segments AB' and $B'D$ for the mapping function. In order to improve the estimates we may choose to use more than one segment prior and after the change in slope. This technique works for the interpolated function.

Note that it is not necessary that the observation points be equally-spaced. We can easily take the actual values of the observation instants in constructing the

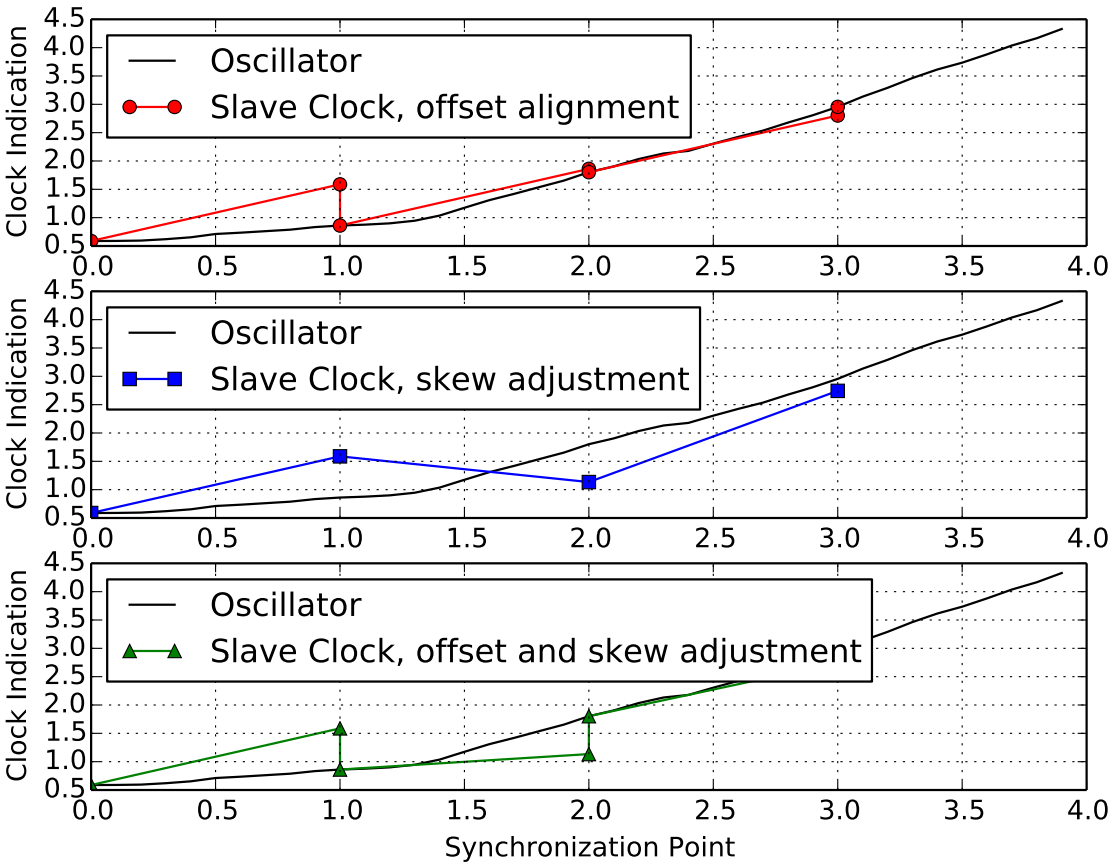

Figure 3.6: Synchronization: using (a) offset alignment, (b) skew adjustment, and (c) offset and skew adjustment

function. Also, note that the variability in the function shown in Figure 3.4 and 3.5 is exaggerated to illustrate the nature of the piecewise linear approximations and have been drawn to explain the phenomenon of estimating the piecewise linear function.

Figures 3.6 is based on the measurements conducted on real clocks and show the occurrences of the phenomena noted above in practical clocks. The top one shows synchronization by adjusting only the offset, the middle one adjusts only the skew (and therefore no time jumps exist), and the bottom one combines skew and offset

adjustments. Any practical system using this approach has to take into account the oscillator stability and the accuracy of timestamping. When the oscillator is very stable the changes in the skew are small and slow leading to an accurate estimation using piecewise linear functions. If the timestamping is not accurate leading to a larger value for σ_a then a suitable filter reflecting past several observations may have to be used.

Chapter 4: Experimental Results

In order to conduct the experimental verification of the result presented in Chap. 3, we used SMiLE 3 boards provided to us through a collaboration with Austrian Academy of Sciences. The SMiLE 3 boards are wireless devices using an FPGA chip where 802.11b physical layer is run [14]. The board is shown in figure 4.1. The hardware is able to timestamp ingress and egress frames with a resolution of 88.78 ps. Starting from these capabilities we present the details of the experimental setup and results.

Figure 4.1: SMiLE3 Board.

4.1 Timestamps on SMiLE 3 board

In the way SMiLE 3 board collects the timestamp using a 32 bit register. This register rolls over every $2^{32} \times 88.78\,\text{ps} = 382.3\,\text{ms}$. Because our experiments run longer, we have to remove this wrap-around by adding 2^{32} so that the time can be treated as a continuous variable.

In our experimental setup, we collect the timestamps from all the nodes at a central server. Each node sends a UDP packet to server whenever it has a timestamp to report. The UDP packet includes the received timestamp of the current message as well as the transmitted time of the latest sendout of the reporting node.

4.2 Scheduling of Wireless Tranceivers

With message exchanges between two nodes, it is clear that the received time is available to the receiver when the message arrives and is timestamped. There are two ways to make the transmit time available to the receiver. One is to send a transmit time inside the message which is supposed to be sent at the transmit time, by setting the timeout with the transmit time onto the outgoing message. Then, the transmit time is available to the receiver as soon as the message is received and decoded. Note that a proper compensation has to be applied in order to match the time which is written in the message to the time at which the actual transmission is made. The other is to timestamp the message at the exact time of departure, and deliver the information as a part of message at the next transmission. Both

the approaches yield equally good results when proper compensation is used. In the experiments reported here we have used both these approaches. The second approach is commonly used with IEEE 1588 protocol. We discuss how to schedule messages to exchange timestamp information for different synchronization methods.

In terms of synchronization methodology, it has been popular to use a master-slave approach. The master supplies the time reference to one or more slaves, which adjust their clocks accordingly. The slave device can change their offset value of the clock reading, change the skew (rate), or adjust both of them as already seen in Section 3.3. In order for a slave to adjust its clock, the following steps can be taken to compute offset and skew values for adjustment.

1. Master sends a Sync message with its own transmit timestamp (t_1).

2. Slave records the received time of the message which contains t_1 and denote the time as t_2.

3. When scheduled transmission takes place, slave records the actual transmitted time by reading a clock value right after the transmission is finished. We denote this timestamp as t_3.

4. Master records its received time of the message sent by slave and also deliver this information to the slave when the next transmission is available. We denote this timestamp as t_4.

5. Slave can compute the ratio of skews between master and itself only if it has at least two consecutive pairs of either (t_1, t_2) or (t_3, t_4). Since we assume

master serves as the reference time, slave changes its clock frequency such that the ratio becomes 1.

When a network lets nodes run independently, steps node should follow are relatively easy. A node sends a packet to other nodes. On receive of message from any other nodes, a node records the received time. If the sendtime is to be sent in a subsequent message, the node sends another message with that timestamp. Otherwise the transmit timestamp inside a message should represent the transmitted time of the current message. This is the approach used in our work.

4.3 Generating a Mapping Function

We use the timestamps to construct a mapping function between two nodes. (see Section 3.3)

As noted in Chap. 3, the midpoints of send and receive times on the two nodes are exactly same time instant in real time. We use this fact to construct the mapping function by defining an approximation using a linear fit between those two points.

4.4 3 Node Case

The first experiment, Expt. I (see Figure 4.2), uses three nodes, a, b, and c placed at fixed locations 4 ft apart. The boards were organized so that each of them sends a message 90 ms apart which is received by the other two nodes.

The send timings of the three nodes are offset from each other so that a may

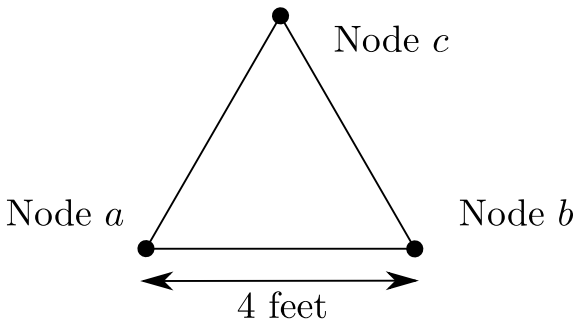

Figure 4.2: Topology - 3 node case.

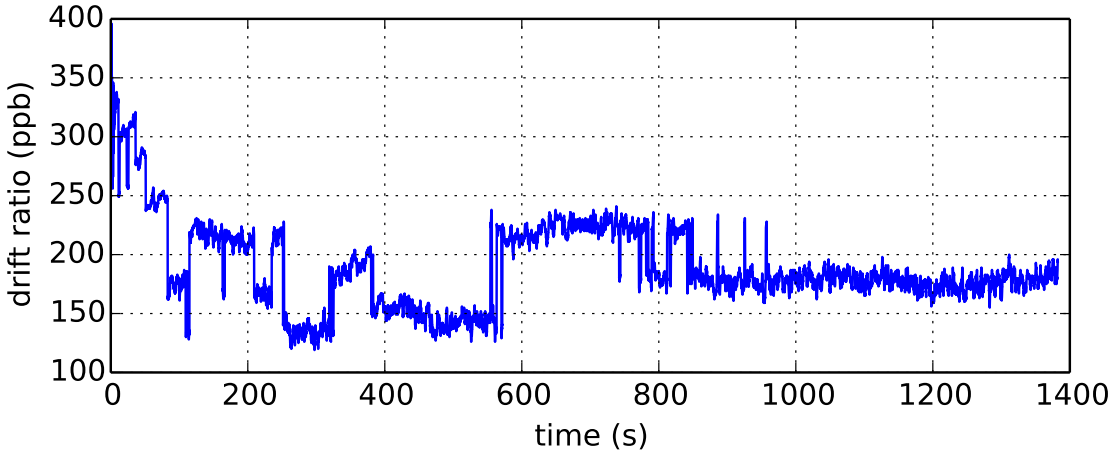

Figure 4.3: Clock skew measured by the linear mapping functions.

start at time 0 ms, b at time 30 ms, and c at time 60 ms, and each repeating sending every 90 ms. This experiment was run for about 1400 seconds and a total of 15000 rounds of communications were recorded. From the empirical observations we estimated the relative drift rate of clocks a and b and the results are presented in Figure 4.3. An interesting observation here is that the local slope of the function changes suddenly at many locations. This is due to the type of temperature-controlled oscillators used in the SMiLE3 boards which make adjustments in the clock interval when a change in the temperature is observed. [15, 16] This results in a sudden change in

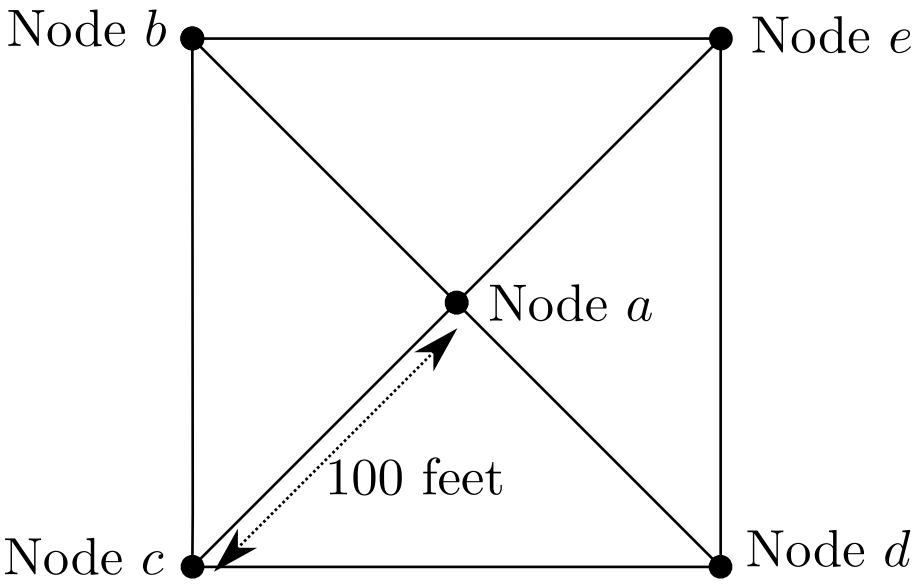

Figure 4.4: Topology - 5 node case.

the drift value which normally changes slowly. In order to make better estimates taking into account the sudden change, we interpolate the instant of sudden change in slope by examining the slope on two sides and find the exact time of the changes. Clearly, this can only be done when we can interpolate and are using the mapping function for a time in the past. The extrapolation, as shown in Figure 3.4 will yield more errors when the slope (skew) changes suddenly.

4.5 5 Node Case

In the second experiment, Expt. II, conducted outdoors, we used a 5 node topology as shown in Figure 4.4. The four corner nodes, were placed approximately 142 ft apart so that node a in the center was approximately 100 ft from the corner nodes.

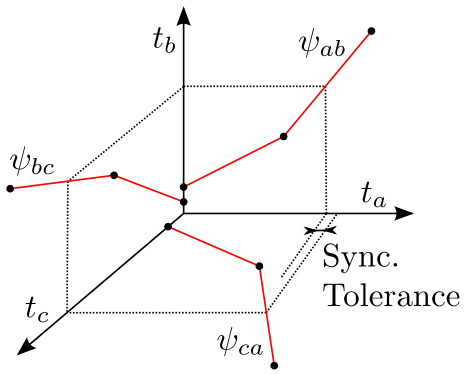

Figure 4.5: Synchronization tolerance.

4.6 Estimating Synchronization Tolerance

We propose to use the results of Theorem 2.1 to estimate the synchronization tolerance for the experimental setup of three nodes presented in Section 4.4. This approach of following a closed path in a connected graph of nodes is very similar to that used in [17]. Using the empirical observations, we constructed a piecewise linear function by using a new estimate for the parameters of the linear mapping function each time a node sends a message, i.e. every 90 ms. Note that the linear segments of the functions at the nodes are not synchronized. The basic idea of this approach of estimating the synchronization tolerance is shown in Figure 4.5. As shown in that figure we start with a random time for clock a, map it to clock b using the mapping function. Then we map clock b to clock c and finally back to clock a. We note that the mapping of t_b to t_c does not result in the initial value t_a, giving us the synchronization tolerance. The synchronization tolerance for node a against the local time t_a is plotted in Figure 4.6.

Figure 4.8, shows the histogram of the synchronization tolerance for the path

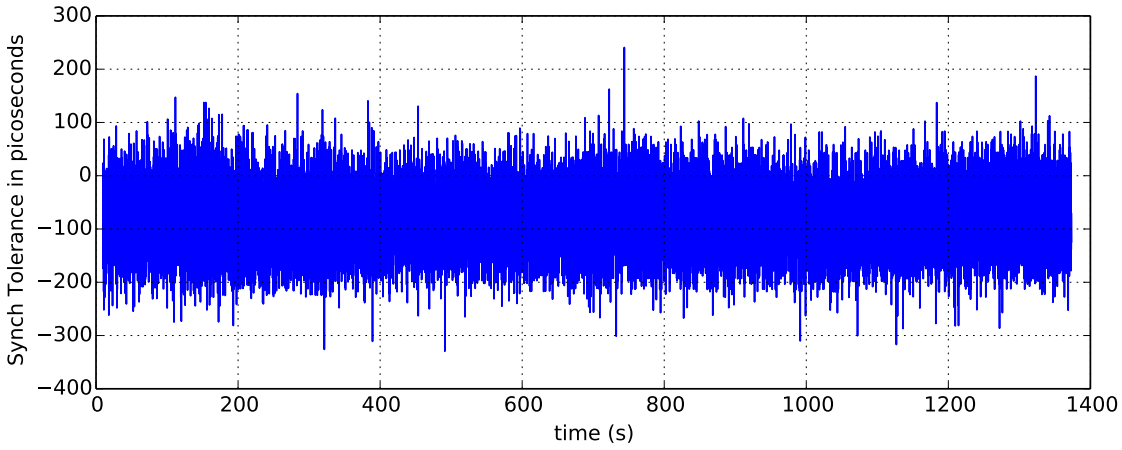

Figure 4.6: Expt. I, path *a*-*b*-*c*-*a*.

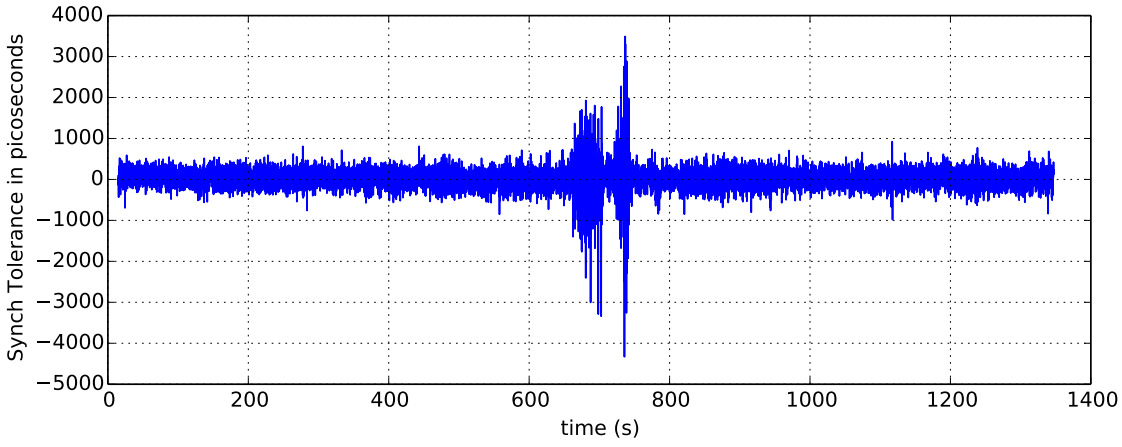

Figure 4.7: Expt. II, path *b*-*c*-*d*-*b*

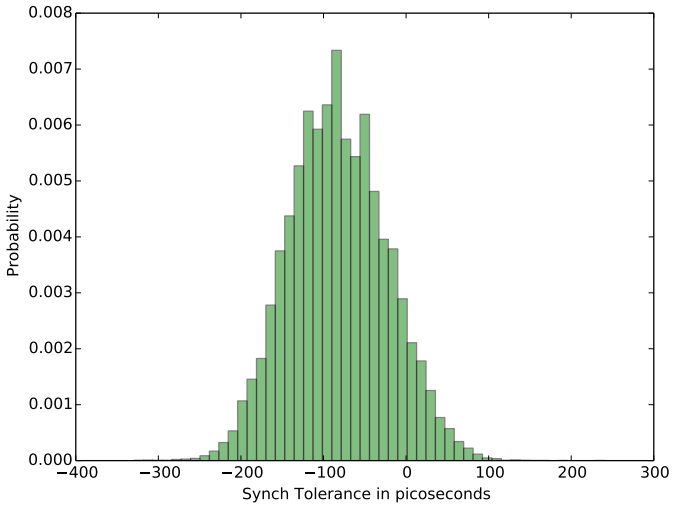

Figure 4.8: Histogram of Synch Tolerance for path *a*-*b*-*c*-*a*.

$a-b-c-a$. We note that the tolerance has a mean value of 82.56 ps and a standard deviation of 60.22 ps with a max to min spread of 604.6 ps. As the oscillators on the nodes are independent from each other, the noise contribution can be added up when traversing a path between multiple nodes. Therefore we estimate the clock synchronization tolerance and the error in one step mapping to have the standard deviation to be 34.62 ps. In Section 2 we noted that, if our assumption that the send time delay and receive time delay are the same for all nodes does not hold, we will see some bias. We believe that the observed value of the mean synchronization tolerance is a reflection of such bias.

In the second set of experiments we used the same methodology and calculated the synchronization tolerance along 10 3-node paths and two 5-node paths. The summary of the results is presented in Table 4.1. Figure 4.9 shows histogram of one of the paths. We note that most of the synchronization tolerance are within 444 ps but there are also some large deviations (a few nanoseconds) around 700 seconds of

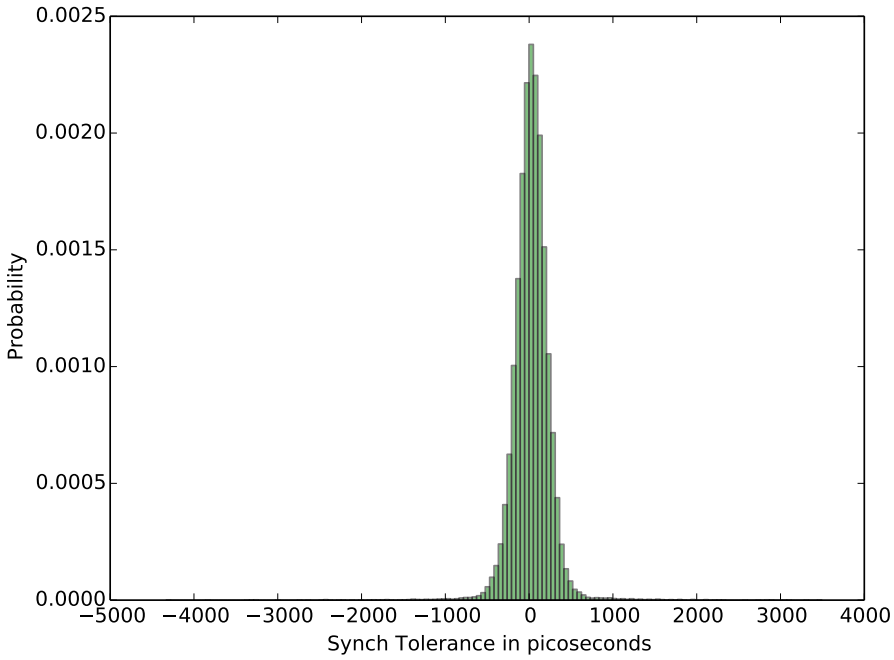

Figure 4.9: Histogram of Synch Tolerance for path *b-c-d-b*

the experiment. These deviations were caused because of physical interference along the paths. When the direct path is obstructed due to the presence of people, the signal may travel along some alternate path which may be much longer. Actually, some people did move between nodes c and d at that time. As a consequence a bias is introduced. Such sudden jumps can be eliminated using filtering techniques which were not deployed in this experiment. The larger mean values (when compared to the first experiment with 3 nodes conducted indoors in a controlled environment) are a direct consequence of the movement of people.

Table 4.1: Statistics of the synchronization tolerance (ps)

path	mean	std	path	mean	std
a-b-c-a	-88.68	99.76	a-d-e-a	43.83	172.98
a-b-d-a	-18.83	159.35	a-c-e-a	288.70	293.24
a-e-b-a	-38.95	150.34	a-d-c-a	-98.24	213.01
b-c-d-b	28.49	229.95	b-d-e-b	-14.05	167.68
b-c-e-b	161.03	295.42	c-d-e-c	-146.54	353.61
c-d-b-c	28.46	218.78	d-b-c-d	28.41	235.47
a-b-c-d-e-a	53.57	238.92	c-d-a-e-b-c	-29.39	251.08

Chapter 5: Location Determination – an Extension from Synchronization

Once we have observed the timestamps or send/receive packets of messages between nodes, we can use that information for determining the propagation delay and thereby the distance between the nodes. The approach is identified in [9]. In this chapter we present the experimental results for determining location.

Provided that the direct propagation delay is observed, we can find the pairwise distances between the nodes. When the propagation delay is not observed and we have only received timestamps from a node at mutiple other nodes, instead of using Time-of-Flight approach, we use Time-Difference-of-Arrival approach. In this chapter we present on both. [7, 18]

5.1 Time of Flight

If there is only one target, we use the coordinates of anchor nodes and distances between the target and each of the anchor nodes. Once distances are known, finding a target's location is to choose a point from some area which can be seen by drawing either circles or spheres with radius of those distance and with centers at positions of the anchor nodes. The least square solution for the location is available in [19].

In addition to Expt. II in Section 4.5, we have a total of 5 different locations of node a. The first location of the target is the center of the square where each corner has an anchor as shown in Fig. 5.1. In Table 5.1, statistics for each location with about 5,000 samples of ToF are presented. From the second to the fifth column, the distance between two nodes, which is denoted as a vertex at the top of the column, is calculated. Ruler distance is obtained by the ruler measurement. ToF Mean is the average ToF distane over the samples. Std denotes the standard deviation over the same set. Corresponding coordinates are presented at the last column. The standard deviation is computed for each of x and y values separately. The results of the rest of locations follow in Fig. 5.2 $\sim$ 5.5 and in Tables 5.2 $\sim$ 5.5. Note that in most cases the standard deviation for both ToF distances and coordinates is less than a foot.

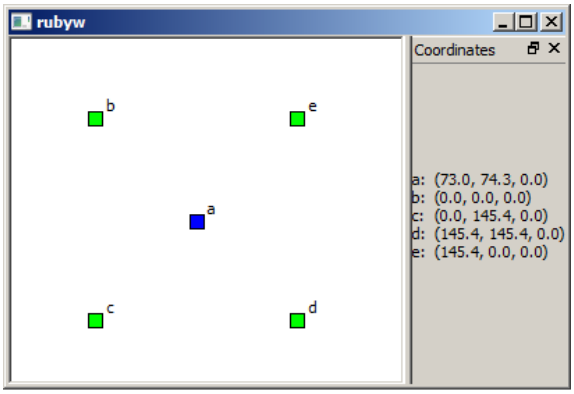

Figure 5.1: Location 1 (ft): (72.7,72.7)

Table 5.1: ToF statistics for Location 1 (ft)

Vertex	$\overline{ab}$	$\overline{ac}$	$\overline{ad}$	$\overline{ae}$	Coordinate (x, y)			
Ruler	102.83	102.83	102.83	102.83	(72.71, 72.71)			
ToF Mean	104.67	101.99	101.99	102.87	(73.60, 74.25)			
Std	0.24	0.16	0.35	0.24	x	0.17	y	0.12

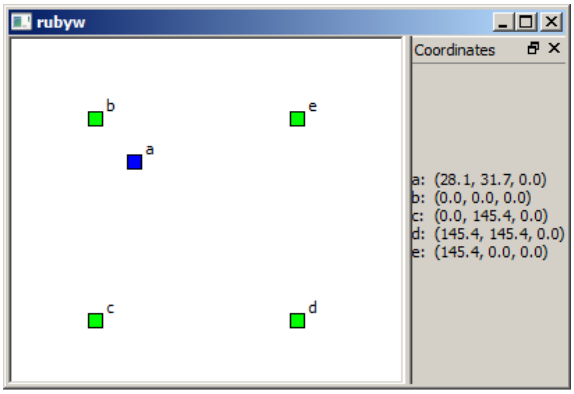

Figure 5.2: Location 2 (ft): (29.7,29.7)

Table 5.2: ToF statistics for Location 2 (ft)

Vertex	$\overline{ab}$	$\overline{ac}$	$\overline{ad}$	$\overline{ae}$	Coordinate (x, y)			
Ruler	42.83	119.05	162.83	119.05	(29.71, 29.71)			
ToF Mean	46.83	113.76	162.49	118.75	(27.88, 32.06)			
Std	0.13	0.76	0.27	0.16	x	0.25	y	0.42

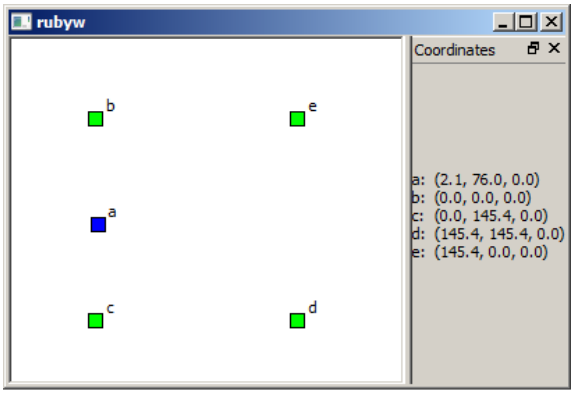

Figure 5.3: Location 3 (ft): (0,72.7)

Table 5.3: ToF statistics for Location 3 (ft)

Vertex	$\overline{ab}$	$\overline{ac}$	$\overline{ad}$	$\overline{ae}$	Coordinate (x, y)			
Ruler	72.71	72.71	162.58	162.58	(0, 72.71)			
ToF Mean	76.75	69.53	157.51	158.76	(2.23, 75.72)			
Std	0.29	0.21	0.24	0.40	x	0.31	y	0.22

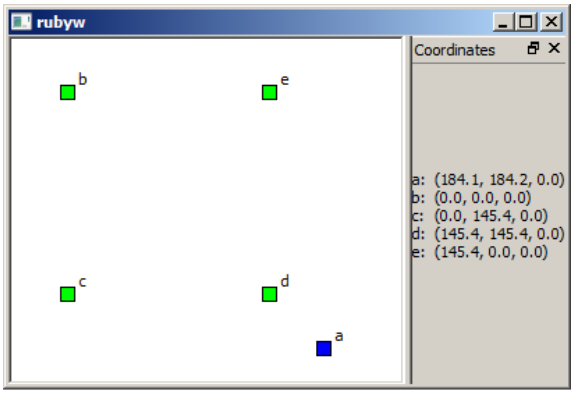

Figure 5.4: Location 4 (ft): (182.8,182.8)

Table 5.4: ToF statistics for Location 4 (ft)

Vertex	$\overline{ab}$	$\overline{ac}$	$\overline{ad}$	$\overline{ae}$	Coordinate (x, y)			
Ruler	258.48	186.55	52.82	186.55	(182.77, 182.77)			
ToF Mean	254.68	184.73	51.00	184.66	(184.59, 184.50)			
Std	0.36	0.30	0.20	0.58	x	0.71	y	0.57

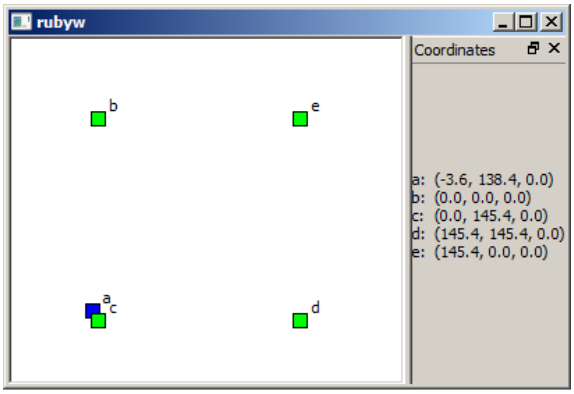

Figure 5.5: Location 5 (ft): (0,138.4)

Table 5.5: ToF statistics for Location 5 (ft)

Vertex	$\overline{ab}$	$\overline{ac}$	$\overline{ad}$	$\overline{ae}$	Coordinate (x, y)			
Ruler	138.42	7.00	145.58	200.76	(0, 138.42)			
ToF Mean	138.31	9.70	151.22	197.63	(-2.60, 137.95)			
Std	0.30	0.11	0.46	2.00	x	1.91	y	0.96

5.2 Time Difference of Arrival

When TDoA is used, a pair of anchor nodes compare their receive timestamps of single message transmitted from the target. The difference of these timestamps is called Time Difference of Arrival. This value is a signed value so that only one hyperbola branch is drawn for two positions of the anchors. Hence, with 4 anchor nodes, we have a total of $\binom{4}{2} = 6$ hyperbola branches. In ideal situation, all of 6 branches have to intersect at exactly one point where the target is located. In practice, due to noise etc, location of the target is obtained by the gradient descent method from [9]. In similar to the previous section, the results are presented in Fig. 5.6 $\sim$ 5.10 and Tables 5.6 $\sim$ 5.10. Note that consensus synchronization [20] is used only for the anchor nodes so that TDoA values are consistent in terms of consensus clock skew.

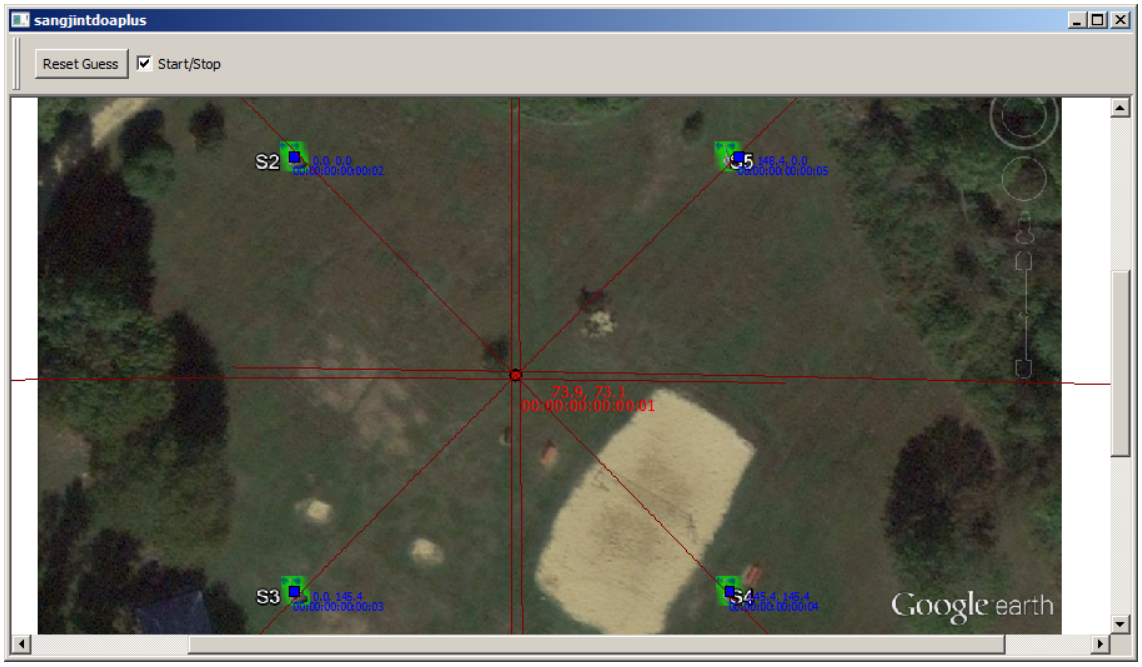

Figure 5.6: Location 1 (ft): (72.7,72.7)

Table 5.6: TDoA statistics for Location 1 (ft)

Foci	b, c	b, d	b, e	c, d	c, e	d, e	Coordinate (x, y)			
Ruler	0	0	0	0	0	0	(72.71, 72.71)			
TDoA Mean	2.60	2.62	1.89	0.07	-0.66	-0.78	(73.38, 73.87)			
Std	0.38	0.40	0.37	0.42	0.46	0.42	x	0.21	y	0.21

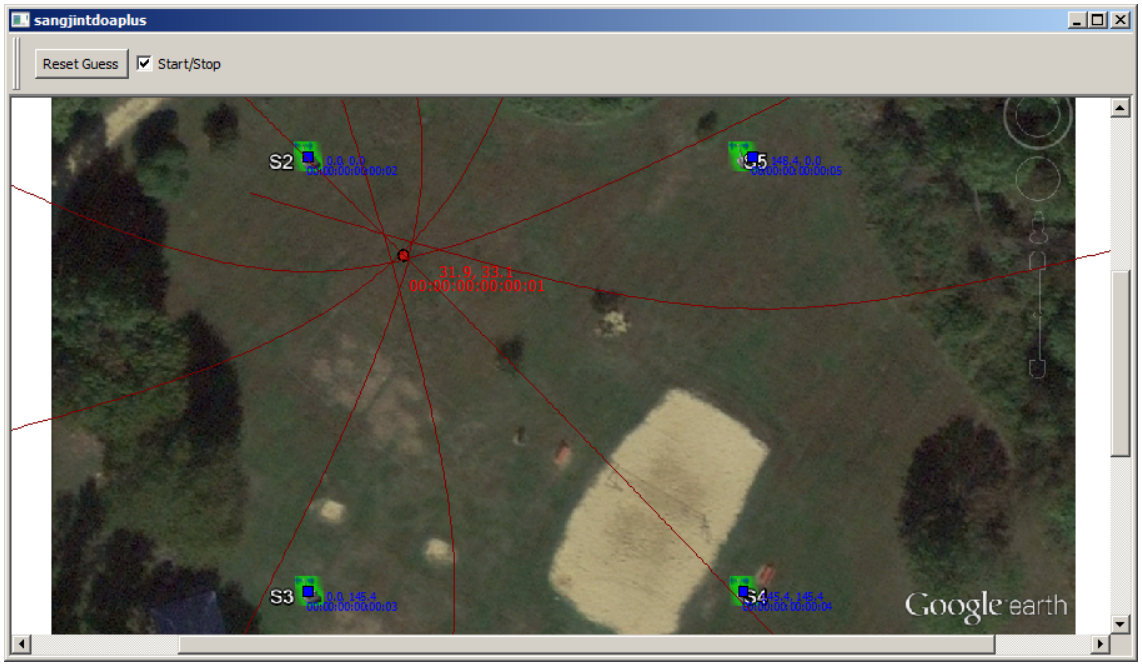

Figure 5.7: Location 2 (ft): (29.7,29.7)

Table 5.7: TDoA statistics for Location 2 (ft)

Foci	b, c	b, d	b, e	c, d	c, e	d, e	Coordinate (x, y)			
Ruler	-76.22	-120	-76.22	-43.77	0	43.77	(29.71, 29.71)			
TDoA Mean	-66.97	-115.59	-71.84	-48.59	-4.78	43.74	(30.34, 34.31)			
Std	0.82	0.40	0.23	0.83	0.83	0.30	x	0.32	y	0.40

Figure 5.8: Location 3 (ft): (0,72.7)

Table 5.8: TDoA statistics for Location 3 (ft)

Foci	b, c	b, d	b, e	c, d	c, e	d, e	Coordinate (x, y)			
Ruler	0	-89.87	-89.87	-89.87	-89.87	0	(0, 72.71)			
TDoA Mean	7.11	-80.73	-81.99	-87.86	-89.09	-1.23	(5.30, 75.87)			
Std	0.53	0.39	0.64	0.42	0.60	0.55	x	0.32	y	0.24

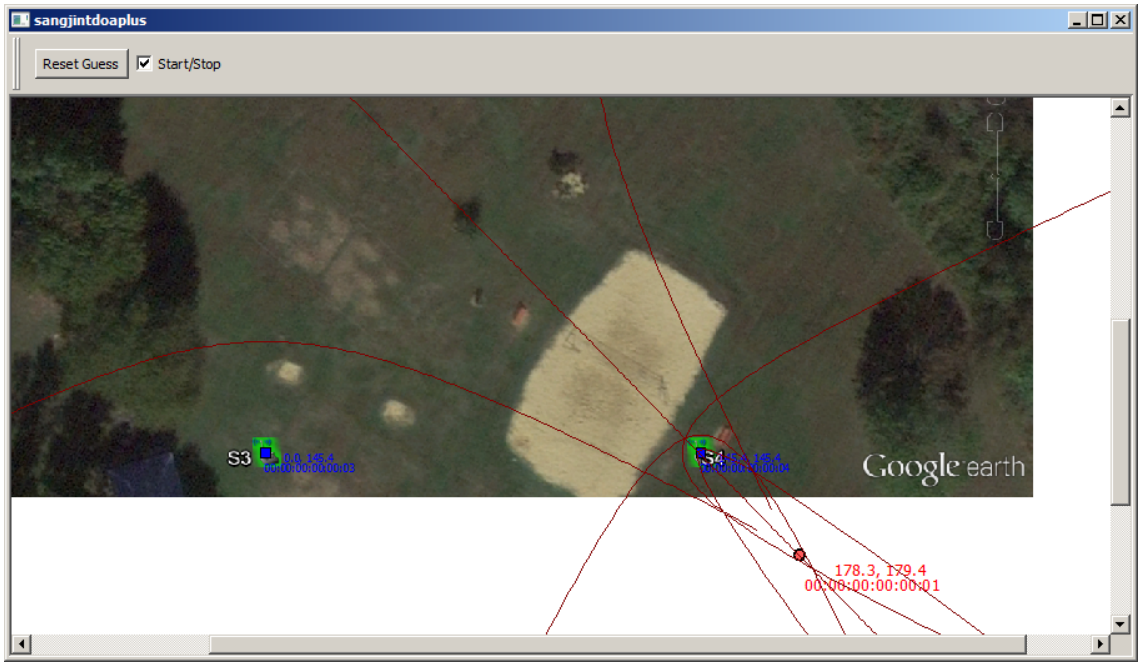

Figure 5.9: Location 4 (ft): (182.8,182.8)

Table 5.9: TDoA statistics for Location 4 (ft)

Foci	b, c	b, d	b, e	c, d	c, e	d, e	Coordinate (x, y)			
Ruler	71.93	205.65	71.93	133.72	0	-133.72	(182.77, 182.77)			
TDoA Mean	69.95	203.76	69.97	133.76	0.07	-133.66	(178.71, 178.62)			
Std	0.57	0.45	0.77	0.42	0.77	0.64	x	1.91	y	1.53

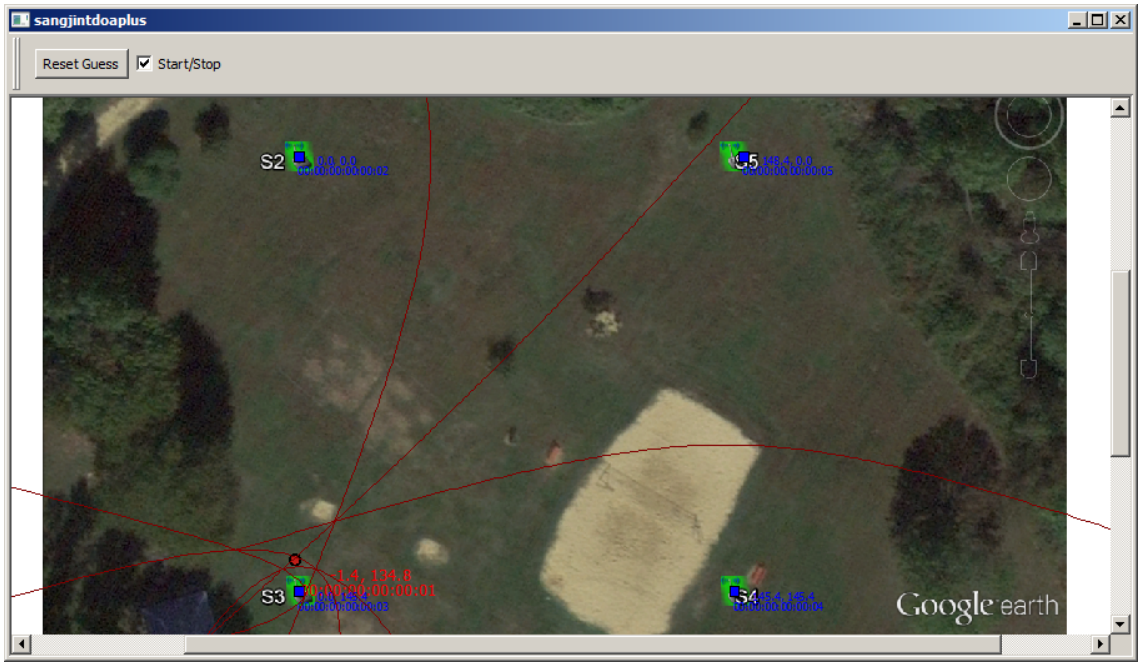

Figure 5.10: Location 5 (ft): (0,138.4)

Table 5.10: TDoA statistics for Location 5 (ft)

Foci	b, c	b, d	b, e	c, d	c, e	d, e	Coordinate (x, y)			
Ruler	131.42	-7.16	-62.34	-138.58	-193.76	-55.17	(0, 138.42)			
TDoA Mean	128.47	-12.81	-59.40	-141.37	-187.89	-46.48	(-1.92, 135.79)			
Std	0.48	0.62	2.05	0.57	2.00	2.07	x	0.81	y	0.48

5.3 Discussion

In table 5.11, errors between locations with ruler measurements and each of location determination outcomes were shown for all 5 locations. In this table, we used location coordinates from Fig. 5.1 $\sim$ 5.10, single coordinate value for each location. Hence, errors reflect not only wireless communication noise, but also the error from the ruler measurement. In the followings, we have stated various sources of these errors.

1. Ruler measurement is done manually. The basic difference between physical location and the computed location based on ruler measurement contributes to the part of error.

2. In the ToF method, all five nodes participate to the consensus synchronization. Thus, the consensus drift, defined as an average of all drifts of nodes, not being necessarily 1 induces the error between ToF distances and physical distances. Moreover, the positive (negative) distance error value tends to push (pull) the location of target towards the outside of the anchor square. Hence, larger error on specific distance vertex contorts a geometrical configuration of all distances, influencing the final location error.

3. In the TDoA method, error due to the consensus drift applies too. Here we have only 4 anchor nodes that participate the consensus synchronization. Errors in TDoA value either squeeze or stretch the hyperbolic branch, and they affect the accuracy of the final location as well.

4. The geometrical relationship between the target and the anchors is also related to the final location error of the target. If the target tends to be inside a polygon of which edges are the anchor positions, each of ToA/TDoA errors mitigate the other errors. The best example of this mitigation is the case where the target is in the center of the polygon and the anchors form an equilateral. The worst example, on the other hand is the case where anchors are inside a small area and the target is far from the area. [21, 22]

Table 5.11: Comparison between different measures

	Location 1		Location 2		Location 3		Location 4		Location 5	
Coordinates	x	y	x	y	x	y	x	y	x	y
Ruler	72.7	72.7	29.7	29.7	0.0	72.7	182.8	182.8	0.0	138.4
ToA	73	74.3	28.1	31.7	2.1	76	184.1	184.2	-3.6	138.4
Errors (ft)		1.63		2.56		3.91		1.91		3.6
Errors (m)		0.49		0.78		1.19		0.58		1.09
TDoA	73.9	73.1	31.9	33.1	6.6	75.4	178.3	179.4	-1.4	134.8
Errors (ft)		1.26		4.05		7.13		5.64		3.86
Errors (m)		0.39		1.23		2.17		1.71		1.18

Chapter 6: Concluding Remarks

We presented a technique for clock synchronization using mapping functions which allows the mapping of the time at one node to the time at any other node in a set of connected nodes. The approach requires the exchange of messages which are then used for estimating empirically the mapping functions. The experimental results show that a very high degree (¡100 ps) of synchronization can be achieved using the SMiLE 3 boards, an unprecedented achievement. The fact that no dedicated master node is necessary for synchronization eliminates the single point of failure as it is the case in several other schemes. Further, the availability of fast, inexpensive processors allows easy translation of the local time using a mapping function. The mapping can also be implemented directly in hardware.

In the discussion we assumed a wireless environment in which all nodes are within range of each other. When this connectivity is not available, the synchronization can still be achieved by using the scheme of mapping functions, but it requires the mapping to be conducted through connected nodes. Otherwise, a more elaborate scheme of communications has to be developed so that a node maintains the mapping function for all other nodes. Schemes similar to those presented in [17] can be incorporated for this purpose.

We also presented the techniques for determining location using time of flight and time difference of arrival approaches using the extensions of basic clock synchronization techniques to determine the time delays. Our experimental results show that the SMiLES 3 boards can give locations with an accuracy of a few inches, again an unprecedented achievement.

A parameter of critical importance in any clock synchronization scheme is the synchronization tolerance achieved. In the proposed scheme we control the time interval between the message exchanges. By increasing the time interval the communication overhead can be reduced, but it is likely to yield larger values of synchronization tolerance. Thus, the synchronization interval can be used to achieve desired synchronization tolerances. By making the message exchange interval dynamic, we can adjust the interval taking into account recent readings. Further studies are required to develop the relationships between the message exchange interval, clock characteristics, and the synchronization tolerance achieved. The work presented here can be extended in multiple directions. We conducted our experiments in environments in which multipaths did not have any significant impact. In most indoor environments multipath becomes a major factor. When determining location indoors, we believe that the approaches presented here have to be complemented by heuristic techniques, reflecting knowledge about the environment, to determine accurate locations.

The SMiLE 3 boards have a simple design using a receiver chip that converts the received signal to baseband, an FPGA and a processor. We believe that all this can be reduced to a one or two chip design which can replace most of the Wi-Fi

chip sets. Then all such devices using these chips can be located accurately opening up a new era for Location Based Services.

Bibliography

[1] I. Skog and P. Handel. Synchronization by Two-Way Message Exchanges: Cramer-Rao Bounds, Approximate Maximum Likelihood, and Offshore Submarine Positioning. *Signal Processing, IEEE Transactions on*, 58(4):2351–2362, April 2010.

[2] R. Exel, G. Gaderer, and P. Loschmidt. Localisation of Wireless LAN Nodes Using Accurate TDoA Measurements. In *Wireless Communications and Networking Conference (WCNC), 2010 IEEE*, pages 1–6, 2010.

[3] G.M. Garner and R. Hyunsurk. Synchronization of Audio/Video Bridging Networks Using IEEE 802.1AS. *Communications Magazine, IEEE*, 49(2):140–147, 2011.

[4] A. Flammini, S. Rinaldi, and A. Vezzoli. The Sense of Time in Open Metering System. In *Smart Measurements for Future Grids (SMFG), 2011 IEEE International Conference on*, pages 22–27, 2011.

[5] P. Loschmidt, R. Exel, and G. Gaderer. Highly Accurate Timestamping for Ethernet-Based Clock Synchronization. *Journal of Computer Networks and Communications*, 2012:11, 2012.

[6] Sangjin Han, Ashok K Agrawala, Matthew Y Mah, Reinhard Exel, and Thomas Bigler. Clock synchronization - an approach using mapping functions. In *European Wireless 2014 (EW2014)*, Barcelona, Spain, May 2014.

[7] Moustafa Youssef, Adel Youssef, Chuck Rieger, Udaya Shankar, and Ashok Agrawala. PinPoint: An Asynchronous Time-Based Location Determination System. In *Proceedings of the 4th International Conference on Mobile Systems, Applications and Services*, MobiSys '06, pages 165–176, New York, NY, USA, 2006. ACM.

[8] Ashok Kumar Agrawala, Ronald L Larsen, A Udaya Shankar, Douglas C Szajda, et al. Method, system and computer program product for positioning and synchronizing wireless communications nodes, May 2007. US Patent 7,224,984.

[9] Matthew Yew Mun Mah. *Time-based Location Techniques Using Inexpensive, Unsynchronized Clocks in Wireless Networks*. PhD thesis, 2011.

[10] IEEE. IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems, Version 2008. *IEEE Std 1588-2008 (Revision of IEEE Std 1588-2002)*, pages c1–269, July 2008.

[11] IEEE. IEEE Standard Definitions of Physical Quantities for Fundamental Frequency and Time Metrology—Random Instabilities: IEEE 1139-2008. *IEEE Standard*, February 2009.

[12] W.J. Riley. Handbook of Frequency Stability Analysis. NIST, July 2008.

[13] Dana B Kamerud. Solution to problem 6104: the random variable X/Y, X, Y normal. *American Mathematical Monthly*, 85:206–207, 1978.

[14] Reinhard Exel. *Time-Based Radio Localization in IEEE 802.11b Wireless Local Area Networks*. PhD thesis, 2012.

[15] Steve Fry. The design and performance of precision miniature tcxos, April 2006.

[16] G. E. Buroker and M.E. Frerking. A digitally compensated tcxo. In *27th Annual Symposium on Frequency Control. 1973*, pages 191–198, 1973.

[17] Roberto Solis, Vivek S Borkar, and PR Kumar. A new distributed time synchronization protocol for multihop wireless networks. In *Decision and Control, 2006 45th IEEE Conference on*, pages 2734–2739. IEEE, 2006.

[18] 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, March 2010.

[19] Jr. Caffery, J.J. A new approach to the geometry of toa location. In *Vehicular Technology Conference, 2000. IEEE-VTS Fall VTC 2000. 52nd*, volume 4, pages 1943–1949 vol.4, 2000.

[20] Morris H DeGroot. Reaching a consensus. *Journal of the American Statistical Association*, 69(345):118–121, 1974.

[21] R. Akl, K. Pasupathy, and M. Haidar. Anchor nodes placement for effective passive localization. In *Mobile and Wireless Networking (iCOST), 2011 International Conference on Selected Topics in*, pages 127–132, Oct 2011.

[22] J.N. Ash and R.L. Moses. On optimal anchor node placement in sensor localization by optimization of subspace principal angles. In *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, pages 2289–2292, March 2008.