DESIGN OF CONCURRENT COOPERATIVE TRANSMISSION SYSTEMS ON SOFTWARE-DEFINED RADIOS

A Dissertation
Presented to
The Academic Faculty

By

Yong Jun Chang

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
in
Electrical and Computer Engineering

School of Electrical and Computer Engineering
Georgia Institute of Technology
December 2013

DESIGN OF CONCURRENT COOPERATIVE TRANSMISSION SYSTEMS ON SOFTWARE-DEFINED RADIOS

Approved by:

Dr. Mary Ann Weitnauer, Advisor
Professor, School of ECE
Georgia Institute of Technology

Dr. Matthieu Bloch
Professor, School of ECE
Georgia Institute of Technology

Dr. Xiaoli Ma
Professor, School of ECE
Georgia Institute of Technology

Dr. Ellen Witte Zegura
Professor, School of CS
Georgia Institute of Technology

Dr. Gregory David Durgin
Professor, School of ECE
Georgia Institute of Technology

Date Approved: November 13, 2013

To my beloved, Sunyoung, Yoonseo, and my son

ACKNOWLEDGMENTS

My half a decade in Georgia Tech was the most precious and valuable time in my life. This dissertation is the result of not only countless nights to roll carts out, but also unconditional support from my beloved wife and daughter. I wouldn't have been able to stand here without their love and sacrifice. Numerous people have helped and supported me to complete this dissertation. Words cannot adequately represent all my appreciation for them.

First and foremost, I would like to express the deepest appreciation to my advisor Dr. Mary Ann Weitnauer, who has shown her attitude, inspiration, and passion. Without her supervision and constant help to this dissertation could not have been possible. She has been an advisor, mentor, and also friend, whom I can open my heart to. It has been a real honor and privilege to work with her and to be her student. Thank you.

I would like to thank my committee members, Dr. Xiaoli Ma, Dr. Gregory David Durgin, Dr. Matthieu Bloch, and Dr. Ellen Witte Zegura, for their efforts and time spent on my dissertation. Their enlightening suggestions have greatly improved my research and the quality of this dissertation. I appreciate the faith and funding of the US Department of Defense (DoD) in giving me the opportunity to pursue my doctoral research in an uninterrupted manner. I specially express my thanks to Robert Scott Frazier in DoD, who has continuously supported my ideas during the projects, and provided insightful suggestions.

I thank my friends and colleagues at the Smart Antenna Research Lab and Georgia Tech, Dr. Jin Woo Jung, Dr. Aravind Kailas, Dr. Syed Ali Hassan, Haejoon Jung, Lakshmi Thanayankizil, Alper Akanser, Vivek Agate, Steve Williams, Jian Lin, Sunghwan Cho, Qiongjie Lin, Wang Feng, Abdul Qadir, Van Nguyen, Gao Zhen, Xia Nan, Wensi Wang and Jason Hongzhi, for spending time and making good memories. I specially would like to express my great appreciation to my friends, Jin Woo Jung, Haejoon Jung and Seok-chul Kwon, who have been spending their valuable time with me during my Tech life. I cannot leave out the words to thank to ever friendly staffs in ECE, Cordai Farrar, Patricia Dixon,

Angel Greenwood, Tasha Torrence, Daniela Staiculescu, and Gail Palmer.

Last but not the least, to my parents go many special thanks for their perpetual love and affection. They have believed and supported me all the time and everywhere. Finally, I would like to take the opportunity to thank all my teachers at Georgia Tech.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Concurrent cooperative transmission (CCT) occurs when a collection of power-constrained single-antenna radios transmit simultaneously to form a distributed multi-input and multi-output (DMIMO) link. DMIMO can be a means for highly reliable and low-latency cooperative routing, when the MIMO channel is exploited for transmit and receive diversity; in this context, the range extension benefit is emphasized. Alternatively, DMIMO can be a means for high-throughput ad hoc networking, when the MIMO channel is used with spatial multiplexing. In both cases, concatenated DMIMO links are treated.

The key contribution of this dissertation is a method of pre-synchronization of distributed single-antenna transmitters to form a virtual antenna array, in the absence of a global clock, such as a global positioning system (GPS) receiver or a network time protocol (NTP) to provide reference signals for the synchronization. Instead, the reference for synchronization comes from a packet, transmitted by the previous virtual array and simultaneously received by all the cooperative transmitters for the next hop. The method is realized for two types of modulation: narrowband non-coherent binary frequency-shift keying (NCBFSK) and wideband orthogonal frequency division multiplexing (OFDM). The pre-synchronization algorithms for transmission are designed to minimize the root-mean-square (RMS) transmit time, sampling and carrier frequency error between cooperative transmitters, with low implementation complexity.

Since CCT is not supported by any existing standard or off-the-shelf radios, CT must be demonstrated by using software-defined radios (SDRs). Therefore, another contribution is a fully self-contained and real-time SDR testbed for CCT-based networking. The NCBFSK and OFDM systems have been designed and implemented in C++ and Python programming languages in the SDR testbed, providing practical performance of the CCT-based systems.

CHAPTER 1

INTRODUCTION

In recent years, cooperative communication has been extensively studied because it is an enabling technology to compensate for limited resources in a wireless network, e.g., low number of antennas, transmit-power limitation, limited battery life, etc. Cooperative transmission (CT) is one of the cooperative communication techniques in which one or more radios that each have a single antenna assist another single-antenna radio to relay a single message, thereby forming a distributed multi-input and single-output (DMISO) link or a virtual array. The signal-to-noise ratio (SNR) advantage provided by DMISO, which comes from spatial diversity gains of the transmit array, can be used to reduce the power radiated from individual transmitters, to extend the range of the transmissions and overcome network partitions, or to reduce the bit error rate (BER) of the received signal. In contrast to a conventional multi-antenna transmitter, which is perfectly synchronized in time and frequency across multiple antennas, a virtual array superimposes signals transmitted from distributed radios that are not connected through a wire or a backbone, and therefore cannot be perfectly synchronized. Concurrent CT (CCT) is a non-coherent type of CT in which the elements of the virtual antenna array transmit at approximately the same time. To make CCT practical and feasible, the relative time and frequency offsets must be small enough to be comparable to the multipath delay and Doppler spreads of the channel. This dissertation is focused on the design of time and frequency pre-synchronization for distributed multi-input and multi-output (DMIMO) links, which are formed when a cluster of radios relays a packet that they received by DMISO, and experimental methodologies to demonstrate the advantages of CCT on software-defined radios (SDRs).

The challenge in achieving performance gain from CCT is in synchronizing the signals from each transmitter so that they are aligned in time and frequency within a required tolerance at a receiver. Each wireless node generates its carrier signal from a local oscillator that

is part of the hardware of the node. Although two arbitrary local oscillators are designed to have the same nominal frequency, they generally have slightly different frequency offsets with respect to each other because of tolerances in manufacturing and variations in temperature [1]. Furthermore, the processing time that is required to process the data packet in each transmitter is highly unpredictable [2, 3], so that the retransmission from the virtual antenna array has a large delay spread at the receiver. A clock that has been globally synchronized by a network time protocol (NTP) or by a global positioning system (GPS) can be provided to deal with the time and frequency impairment. However, these approaches require additional devices or a long convergence period for accurate time and frequency synchronization. In our method, the reference for synchronization comes from a packet that is received simultaneously by the transmitters in the virtual antenna array. This packet can be transmitted from the initiator node, which is called the "source node" in this dissertation, or it can be encoded with a form of transmit diversity from another virtual antenna array. In this dissertation, the synchronization algorithms for CCT are designed to minimize the root-mean-square (RMS) transmit time and frequency error between transmitters with low implementation complexity.

Since CCT is not supported by any existing standard or off-the-shelf radios, the advantage of CCT usually has been proven theoretically in the literature or demonstrated by using SDRs. A number of papers have been published in recent years describing SDR implementations of CCT [4–8]. Each one, however, falls short of realizing the self-contained and real-time SDR implementation described here in. In addition, most existing experimental studies of CCT that use SDRs have been focused mainly on the evaluation of the performance of the physical layer, and therefore lack support for the multiple layers of a wireless network.

1.1 Research Contributions

The contributions of this research are as follows:

- **Experimental testbed using Software-Defined Radio**

  The first contribution is a fully self-contained and real-time SDR testbed for CCT-based networking. GNU Radio and USRP systems have been selected for the base framework of the testbed. The framework has been modified to support real-time CCT operation on both of USRP1 and USRP2. The CCT algorithms designed in this research are coded in C++ and Python programming languages, and tested in the SDR testbed. Motivated by the fact that the existing SDR testbeds lack of multi-layer support, a modular protocol stack that can run on the SDR testbed has been developed. The modular implementation of the protocol stack enables a protocol in each layer to be swapped by another protocol on the fly, allowing a comparison of one protocol with another under the same conditions. The framework of the testbed enables evaluation of link layer and network layer protocols that have CCT in the physical layer.

- **Pre-synchronization algorithm for CCT**

  The second contribution of this research is a pre-synchronization algorithm for CCT based on start-of-preamble (SOP) time estimation and T_{proc}, which is an upper bound on processing times in the radios, and a method of combining multiple estimates to minimize the retransmission synchronization error. Instead of using a global clock, such as a global positioning system (GPS) receiver or a network time protocol (NTP) to provide reference signals for the synchronization, the reference for pre-synchronization comes from a packet, simultaneously received by all the relays, in which the packet is encoded with a form of transmit diversity. The combining method for the pre-synchronization is designed to minimize the root-mean-square (RMS) re-transmission time, sampling frequency and carrier frequency errors between cooperative transmitters with low implementation complexity using approximated best linear unbiased estimator (ABLUE).

- **Pre-synchronization for narrow-band CCT using non-coherent BFSK**

  The third contribution is a time estimation algorithm for narrow-band CCT. The algorithm is designed to be robust to carrier frequency offset (CFO) and to satisfy the ABLUE combining rule with low implementation complexity. Non-coherent binary frequency-shift keying (NCBFSK) modulation with equal-gain combining is used for the implementation of the estimation algorithm. The performance of the algorithm is evaluated through simulation. In addition, the SNR advantage of CCT and statistical stability of multi-hop CCT have been demonstrated through the SDR testbed.

- **Time and frequency estimation algorithm for OFDM-based CCT**

  The fourth contribution is a time and frequency estimation algorithm for orthogonal frequency-division multiplexing (OFDM)-based CCT. In contrast to a narrow-band system, an OFDM system has been designed to cope with wideband transmission. However, an OFDM system is vulnerable to frequency offset between cooperative transmitters because the offset can induce inter-carrier-interference (ICI). In addition, the timing offset between transmitters creates additional delay spread so that the design of a guard interval has to embrace the additional timing spread. By using frequency-domain symmetric correlation, a high-accuracy time and frequency offset estimation algorithm has been developed to avoid large time and frequency spread for CCT. The performance of the estimation algorithm is evaluated through simulation as well as experiment on the testbed.

- **Network time synchronization using a cooperative digital-and-analog protocol**

  The last contribution of the research is a cooperative digital-and-analog protocol for rapid network time synchronization. Inspired by the high quality of the CT time synchronization developed in this research, and by the low and stable synchronization error statistics in multi-hop CCT, our proposed method exploits analog and digital

forms of CT at different stages of the synchronization process. The protocol consists of two stages: 1) CCT broadcasting of the seed-time and 2) analog CCT to compensate propagation errors. The proposed protocol has been developed using the SDR testbed and demonstrated in an indoor environment, and compared to the performance of the Timing-sync Protocol for Sensor Networks (TPSN), which is a well-known time synchronization protocol for a sensor network.

CHAPTER 2

ORIGIN AND HISTORY OF THE PROBLEM

2.1 Cooperative Relay Systems

Traditionally, a wireless "link" is understood to be a wireless connection between one transmitting node and one receiving node. Cooperative communication encompasses various communication techniques in which multiple nodes assist another in many different ways to enhance a communication link. From the fact that multiple nodes are used to improve the link quality, cooperative communication can be considered as a relaying technique where multiple relay nodes cooperate to deliver the source message. Because of available degrees of freedom of such systems, many different relaying architectures can exist. Some canonical relaying architectures are briefly introduced in this section.

2.1.1 Relaying topologies

Various relaying topologies can be categorized as in Figure 1. The traditional relaying, illustrated in the top-left of the figure, is realized by means of one or more serial relays delivering a packet from source to destination. The propagation path between source and destination is divided into small propagation distances by the relays. The path loss is known to increase with the propagation distance, therefore the signal-to-noise ratio (SNR) is inversely proportional to the distance, thereby SNR $\propto d^{-\alpha}$ where d is the distance between a transmitter and a receiver, and α is the path-loss exponent. Because of the non-linear path-loss behavior, the end-to-end detection performance can be improved by splitting the propagation path in the traditional relaying system compared to the direct transmission, at the cost of latency.

The supportive relaying, also known as the three-node cooperation model in [9], is the simplest cooperative relaying scheme where a third node supports the direct communication between source and destination. If the direct transmission from the source to the

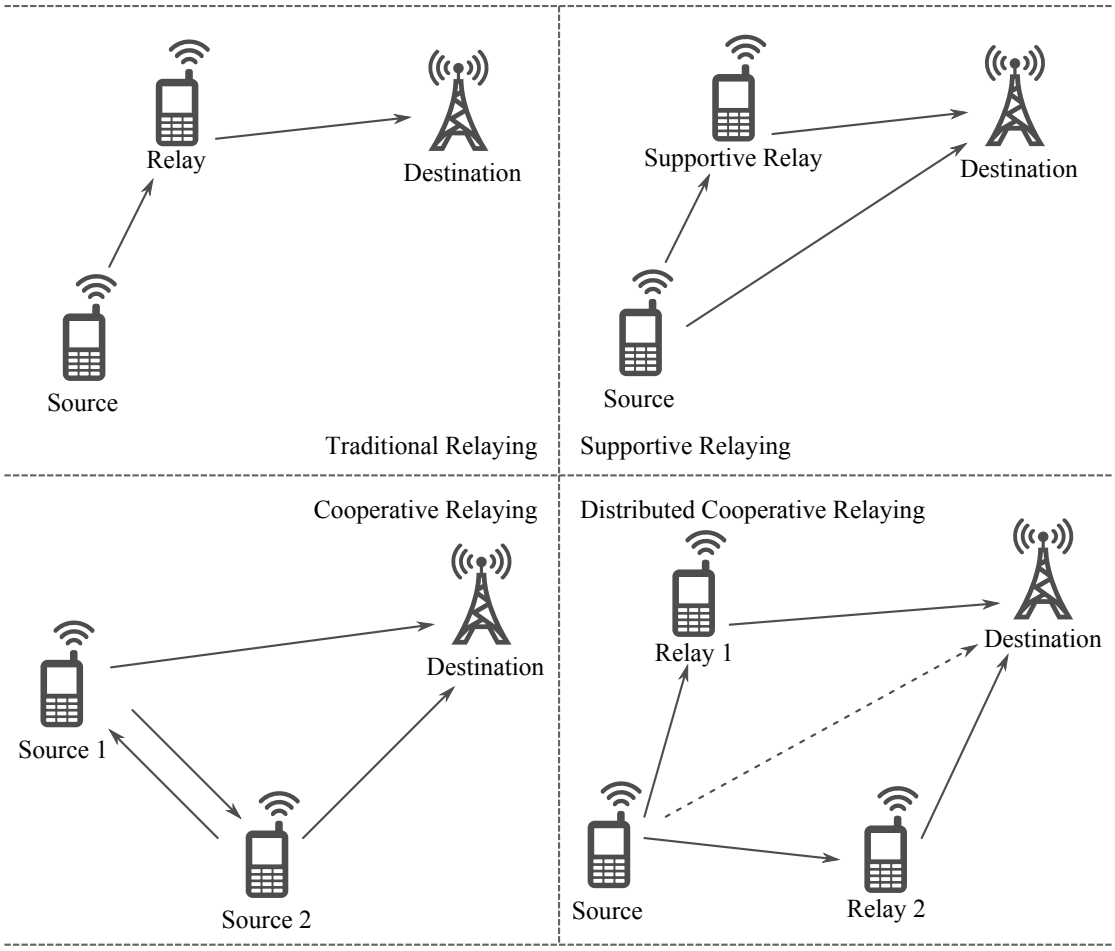


Figure 1: Illustration of various relaying topologies.

destination is not successful, the overheard information from the source is forwarded by the supportive relay to reach the destination through a different channel. Since the two communications take different paths and take place one after another, this example implements the concept of spatial diversity (i.e., cooperative diversity of [10]). The supportive relaying scheme can be extended to cooperative relaying, where at least two nodes are each other's supportive relay node to enhance the other's communication link [10]. In this case, one of the cooperative nodes acts as a source node in turns, and the other focuses its resource to relay the source information to the destination.

In distributed cooperative relaying, multiple relays can be deployed between source and destination to relay a source message to the destination node. Under the half-duplex

constraint where a wireless node cannot receive and transmit at the same time, the entire transmission is achieved in two phases. In the first phase, also referred to as a distribution phase, the source transmits the message to the destination, and the multiple relays overhear the source message. Depending upon the propagation condition, there may or may not a direct link between source and destination. In the second phase, the multiple relays including the source can choose one of the many possible physical-layer transmission techniques to create spatial diversity (e.g., orthogonal transmission, space-time coding, beamforming, etc.)

2.1.2 Relaying strategies

In a cooperative relaying system, a relay node can choose one of the relaying strategies in terms of how to create the retransmission signal. The relaying strategies can be distinguished by the amplify-and-forward, decode-and-forward, and compress-and-forward strategies:

- The amplify-and-forward (AF) strategy allows the relay to simply amplify the received signal from the source and to forward it to the destination. AF can be considered as a special type of transparent relaying in [11], where the relay performs a linear operation in the analog domain not attempting to decode the data.

- In the decode-and-forward (DF) strategy, also known as regenerative relaying [11], the relays process the overheard transmission in the digital domain. The relay can choose either of symbol-by-symbol decoding or full decoding [10]. Assuming a packet consists of multiple symbols, in the symbol-by-symbol strategy, the relay decodes (perhaps incorrectly) and re-encodes each symbol, and allows the destination to perform full decoding of the packet. In the full decoding strategy, in contrast, the relay retransmits the re-encoded symbols only when the decoding the packet is decoded successfully (i.e., without error). Whenever unrecoverable errors happen in the relay node, under full decoding, the relay does not participate in the cooperative

transmission.

- The compress-and-forward (CF) strategy allows the relay node to compress the received signal from the source and digitally forward it to the destination without decoding the signal.

Often, relaying strategies are analyzed in terms of outage, which is the probability that the received SNR, after diversity combining, falls below some desired level. The outage performances of AF and DF relaying strategies for a supportive relaying topology are summarized here [10]. Let $a_{s,r}$, $a_{s,d}$, and $a_{r,d}$ denote the channel fading coefficient of source-relay, source-destination and relay-destination channels respectively. The supportive relaying produces an equivalent one-input, two-output complex Gaussian noise channel. As [10] details, assuming equal transmit power for all the nodes with the average SNR γ, the mutual information of the AF strategy is given by

$$I_{AF} = \frac{1}{2}\log_2\left(1 + \gamma|a_{s,d}|^2 + f(\gamma|a_{s,r}|^2, \gamma|a_{r,d}|^2)\right)$$

as a function of the fading coefficients, where

$$f(x, y) := \frac{xy}{x + y + 1}.$$

The outage event for spectral efficiency R is given by I_{AF} and is equivalent to the event

$$|a_{s,d}|^2 + \gamma^{-1} f(\gamma|a_{s,r}|^2, \gamma|a_{r,d}|^2) < \frac{2^{2R} - 1}{\gamma}. \tag{1}$$

In contrast, the mutual information and its corresponding equivalent outage event for the fixed DF strategy are respectively given by

$$I_{DF} = \frac{1}{2}\min\left\{\log\left(1 + \gamma|a_{s,r}|^2\right), \log\left(1 + \gamma|a_{s,d}|^2, \gamma|a_{r,d}|^2\right)\right\}$$

and

$$\min\left\{|a_{s,r}|^2, |a_{s,d}|^2 + |a_{r,d}|^2\right\} < \frac{2^{2R} - 1}{\gamma}. \tag{2}$$

For Rayleigh fading, the authors in [10] have shown that the AF strategy is superior to the DF full decoding (FD) strategy for statistically symmetric networks. The vulnerability of the DF-FD strategy comes from loss of relay support when the channel gain between the source and the relay falls below the decoding threshold. In this case, the supportive DF relay does not contribute to the cooperation since it cannot decode the source message. On the other hand, the AF relay still can contribute to enhance the direct communication link. When the relay channel gain is guaranteed, however, DF outperforms AF because the amplified signal has insertion noise from the relay node while the DF relay does not. It is equivalent to have $|a_{s,r}|^2 \to \infty$ from (1) and (2), and the equivalent outage events for AF and DF can be written as respectively

$$\text{AF Outage:} \quad |a_{s,d}|^2 + \frac{1}{2}|a_{r,d}|^2 < \frac{2^{2R}-1}{\gamma} \tag{3}$$

and

$$\text{DF Outage:} \quad |a_{s,d}|^2 + |a_{r,d}|^2 < \frac{2^{2R}-1}{\gamma}. \tag{4}$$

It can be readily shown that the event (3) has higher probability than the event (4). The error probability performance of the AF and the DF strategy is studied in [12, 13]. As the outrage performance, letting the channel gain between source and relay be large enough, the DF scheme has better error probability performance than the AF scheme.

In contrast, for the no-relay non-CT case, there is only the $|a_{s,d}|^2$ term on the left:

$$\text{Non-CT Outage:} \quad |a_{s,d}|^2 < \frac{2^{2R}-1}{\gamma}, \tag{5}$$

which clearly has the highest probability. We notice that the expected value of the left of the DF case is twice that of the no-relay case,

$$E\{|a_{s,d}|^2 + |a_{r,d}|^2\} = 2E\{|a_{s_d}|^2\}. \tag{6}$$

This factor of two is the array gain. If we compare DF to no-relay with the constraint of equal total transmit power, we must halve the power of source and relay by two; yielding

$$\text{DF(power constraint) Outage:} \quad \frac{1}{2}|a_{s,d}|^2 + \frac{1}{2}|a_{r,d}|^2 < \frac{2^{2R}-1}{\gamma}. \tag{7}$$

From (6) and (7), we observe that DF no longer has an array gain advantage. However, the probability of (6) is still lower than DF outage probability (7), because the variance of the LHS of (5) is lower than the variance of the LHS of (7). The factor by which the LHS of (6) could be increased to make the probabilities equal is called diversity gain.

2.2 Cooperative Transmission

In a distributed cooperative relaying scenario, the multiple relays including the source create a virtual array-transmission, referred to as cooperative transmission (CT), in which diverse versions (i.e., versions ideally that undergo independent channel fading) of the same message, transmitted by different radios, are combined in the physical layer of the receiver. Compared to the direct transmission or the traditional relaying with the same transmit power per node, CT provides a signal-to-noise ratio (SNR) advantage through spatial diversity and array gains, which can be used to reduce radiated energy per node or extend the range and overcome network partitions. Figure 2 shows an example of CT when the network partition exists between the source and the destination. The example shows that the SNR from the direct transmission at the destination is far below the decoding or even detecting threshold, therefore the direct link is not available. Similarly, each of the relays cannot overcome network partition by itself. By exploiting multiple relays and performing CT, the cooperative array gain may allow the CT to overcome the network partition. Even when the partition is dynamic so that some of the relays have deep fading while the others not, the spatial diversity gain can be achieved in CT without choosing the best relay or requiring feedback from the destination to select the best relaying strategy.

In terms of channel usage, CT can be categorized by orthogonal CT, co-channel CT and mixed CT [10]. In orthogonal CT, the multiple transmissions from the relays will be orthogonal in time, frequency, or code. If a receiver can receive the signal in all orthogonal channels, then maximum-ratio combining (MRC) optimizes the received SNR, and both

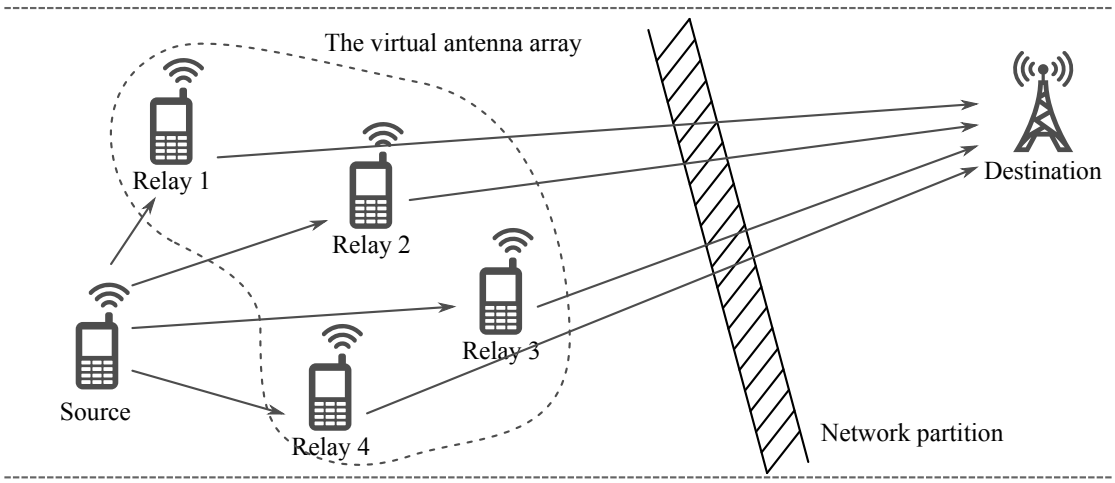


Figure 2: Illustration of the cooperative transmission when the network partition exists and there is no direct link between the source and the destination.

array gain and diversity gain can be achieved. On the other hand, all the relay transmissions are transmitted through the same channel in co-channel CT. Co-channel CT can be further divided into coherent and non-coherent. In non-coherent co-channel CT, channel status information (CSI) is not available to the transmitter so the phases are random and the signals could cancel each other out, i.e., there could be CT "self-fading." In this case, only array gain can be achieved. On the other hand, coherent co-channel CT, also known as cooperative beam-forming, requires CSI to be fed back from the receiver to each relay transmitter, so that the phases of the transmitters can be synchronized [14]. While this type of CT has the highest gain, the network overhead to provide this feedback may not be possible in many applications. To form "mixed" CT, orthogonal CT and non-coherent co-channel CT can be combined where the orthogonal resources are limited. In the mixed CT strategy, some relay transmitters are allowed to use the same orthogonal channel for relaying operation.

Orthogonal CT, the subject of the research in this dissertation, requires no transmitter-side CSI and provides diversity and array gain. The orthogonality can be achieved in different ways. One way that is assumed by many authors is that each cooperating radio's transmission occurs orthogonally in time [10, 15, 16]. This technique has the benefit

that the synchronization of time and frequency at the receiver is done separately for each transmitter, using the same techniques that are used for non-CT communications. Another benefit is that the range of interference is the same as a single-input-single-output (SISO) transmission. The disadvantage of this approach is that the rate of the transmission is not efficient, because there are no CTs allowed in a link. In addition, more energy must be expended in the preambles for synchronization. Alternatively, if the cooperative transmissions can take place synchronously and simultaneously, i.e., well within the root-mean square (RMS) propagation delay spread of the channel, then a higher-rate code (e.g., the Alamouti code [17]) can be used [18]. This type of CT is referred to as concurrent CT (CCT) in which the cooperating radios transmit through orthogonal channels at approximately the same time.

2.3 Concurrent Cooperative Transmission

CCT allows cooperating nodes to transmit the diversity versions of the message at approximately the same time. The major challenge in achieving performance gain from CCT is in synchronizing the signals transmitted from multiple transmitters so that they are aligned in time and frequency within the required tolerance of the receiver. Each wireless node generates its carrier signal from a local oscillator mounted on the node. Although two local oscillators are designed to have the same nominal frequency, they generally have a slightly different frequency offset because of tolerances in manufacturing and variations in temperature. A typical crystal oscillator has ±5∼10 ppm tolerance which corresponds to ±12∼24 Khz at 2.4 Ghz carrier frequency [1]. An oscillator for a special application (e.g., GPS) has higher accuracy, for example, ±0.1∼1 ppm [19]. The impairment of a local oscillator creates different types of offsets in a relay node. In a digital transceiver system, the signal processing is done by logical circuits where the operating clock is derived from the local oscillator. Therefore, the signal processing time for the same operation on two arbitrary relays can differ because of the difference of the operating clocks. In addition, since

the ADC and the DAC are operated by the operating clock, two relays have a slight different sampling frequency, which causes mutual inter-symbol-interference (ISI). The local oscillator is also used to synthesize a carrier frequency [20] causing the cooperative message received from a destination node to have multiple carrier frequency offsets (CFOs).

Since the relay operation takes a random processing time associated with checking error, parsing a header, and re-encoding a received packet, the retransmissions from the multiple relays cannot be perfectly aligned in time. The impairment of the performance of CCT caused by time synchronization errors was theoretically treated in [21]. The authors showed that time jitters larger than 10% of the bit duration degrade the bit-error rate (BER) performance of the system. The processing time is on the order of tens of microseconds in a commercial wireless sensor mote [22], and tens of milliseconds in a software-defined radio (SDR) system [2]. Therefore, synchronization for relays, at least for SDRs, is necessary.

The existing frequency synchronization schemes for CT can be classified into three types: post-synchronization, self-cancellation, and pre-synchronization. In the post-synchronization schemes [23–25], all the CFOs are estimated and compensated at the receiver, so the complexity is very high, hindering a real-time operation. An effective inter-carrier-interference (ICI) self-cancellation scheme was proposed in [26, 27] for orthogonal frequency-division multiplexing (OFDM) systems based on symmetrical conjugated mapping, but its bandwidth efficiency is low, and it may not be applicable when the number of relays is larger than two. A pre-synchronization method was proposed by [28, 29], in which the CFOs between source and relays are pre-compensated at relays. However, these schemes are designed for the two-relay, two-hop scenario, in which the source is required to reach the destination. In [30], the authors propose the pre-synchronization scheme that the CFOs between users and the base station are estimated during the downlink communication; and these CFOs are then pre-compensated for the uplink transmissions. However, these pre-synchronization schemes are not suitable for consecutive CCT because they are designed especially for the two-relay and two-hop scenario.

The time and frequency synchronization for OFDM-based cooperative systems has been discussed in [31, 32]. In [31], the use of orthogonal training preambles has been proposed to estimate multiple timing and frequency offsets. The method estimates the multiple offsets at the destination and to feedback the estimates back to the relays so that the data can arrive in a synchronous manner. However, the feedback operation requires extensive resources in some applications. Furthermore, the feedback might not be possible if the cooperation creates range extension where the destination cannot reach back to the relay by itself. In [32], the authors proposed to have a training phase before data transmission. During the training period (TP), unit-amplitude phase shift keying (PSK) training signals are transmitted from the source to k^{th} relay and from k^{th} relay to the destination. From the fact that the training signals from all the relays are linearly independent, the destination can estimate multiple offsets from the training signals. However, the TP takes additional resources for the estimation and needs to be performed frequently if the offsets are changed over time.

The time and frequency synchronizations using external resources are proposed in [33, 34]. In [33], the authors proposed the alignment of transmit time using a global positioning system (GPS). In [34], a combination of GPS and network synchronization based on the precise timing protocol (PTP) is proposed to synchronize the primary clock in distant cooperative base stations. However, these approaches require additional devices or a convergence period for accurate synchronization of times.

2.4 Multi-hop Cooperative Transmission

In ad hoc networks, the network is made up of multiple nodes connected by links that are typically determined by the transmission range of a wireless node. Any pairs of two nodes in the network can be a source and a destination node. Assuming that the nodes in the network are not in a single-collision domain, the link between a source and a destination forms

multi-hop relaying. Distributed multiple-input-multiple-output (DMIMO) has been gaining attention recently in the context of multi-hop networks as a means for highly reliable and low-latency cooperative routing, when the MIMO channel is exploited for transmit and receive diversity [9,33,35], or as a means for high-throughput ad hoc networking, when the MIMO channel is used with spatial multiplexing [36]. In both cases, a cooperative route is realized as a series of relaying clusters, where one cluster relays a packet to the next cluster.

Two types of CT models for a DMIMO network are suggested in [9] and illustrated in Figure 3. The upper figure illustrates the use of cooperative gateways for the multi-hop network. The network is divided into predefined clusters, and a gateway node or cluster head is selected in each cluster. Each gateway is responsible for delivering the message to the gateway node in the next cluster by collecting cooperative relays within its transmission range and exploiting CT. In this scenario, each set of gateway-relays-gateway links can be considered as a distributed cooperative relaying topology in Section 2.1.1, thereby any relaying strategy can be chosen for this model. For synchronization, the recruited relay nodes can rely on the synchronization data available in the source packet, unless there exists external synchronization resources. The relays-to-gateway link is also called a distributed multiple-input-single-output (DMISO) link. The end-to-end performance of this model is discussed in [37]. The authors showed that the use of multiple DMISO links in a mobile ad hoc network provides an improvement of end-to-end throughput and delay, and robustness to link failures.

The lower scenario in Figure 3 illustrates a multi-hop DMIMO route where the nodes in each cluster relay the message transmitted from the previous cluster, without gateway nodes. Since there is no collection and redistribution of data within a cluster, there is less end-to-end delay. The participation or transmission by a relay is opportunistic and depends on two conditions: 1) successful decoding as evidenced by passing the cyclic redundancy check (CRC), and 2) passing the network layer test, such as being a member of the route or having a non-expired time-to-live. For example, in the first hop, a collection of the nodes

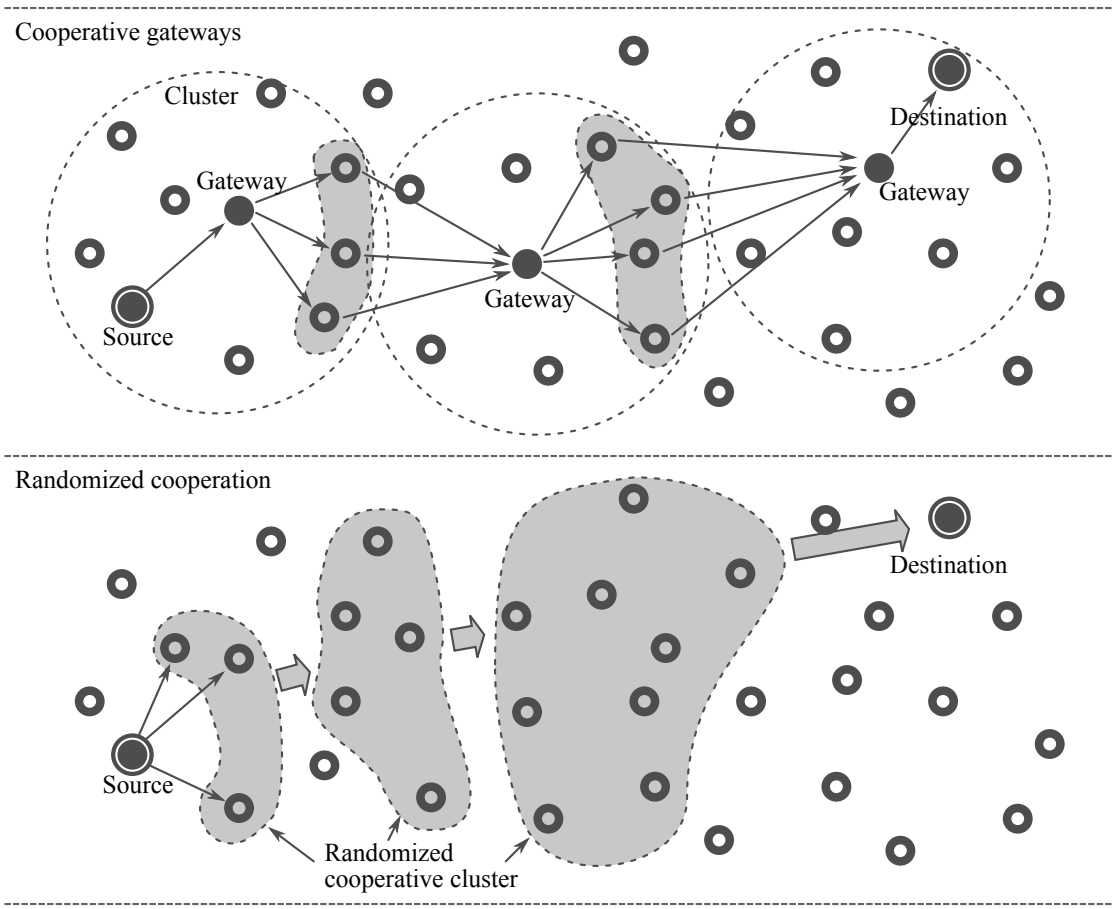


Figure 3: Illustration of two types of CT for an ad hoc network. The upper figure shows to use cooperative gateways while the lower figure shows to use randomized cooperation.

that successfully decode the source message becomes a relay cluster. In [9], the authors claim that the use of the randomized cooperative coding in this scenario provides comparable cooperative gain to the one of a centralized scenario in which orthogonal channels or codes are explicitly assigned to the relay nodes. Because this architecture requires no recruitment and redistribution in each hop, the broadcasting and unicasting protocols using this architecture are simple with low overhead [38, 39]. However, without having external synchronization resources, the transmit synchronization is more challenging in this scenario because of the superposition of reference signals in each receiver that are imperfectly synchronized.

2.5 Experimental Analysis of Concurrent Cooperative Transmission

While cooperative communications has a rich theoretical history in the literature, efforts to implement cooperative systems have been much more limited. A number of papers have been published in recent years describing cooperative implementations [4–8, 40–42]. Each one, however, falls short of realizing the complete, real-time transceiver. Some platforms are based on commodity wireless devices, such as sensor motes (e.g., Mica2 or MicaZ) [34,40,41] or wireless network interface cards (NICs) (e.g., IEEE 802.11b NIC) [42]. Even though diversity combining can be achieved at the packet level in these commodity devices, they do not allow us to access the physical layer. Therefore, a hard-decision packet combining method can be used for a commodity wireless device. However, the packet combining based on hard-decision values cannot achieve full diversity [43]. In addition, the packet combining does not allow pre-synchronization and diversity transmission in the physical layer. Aside from commodity devices, some testbeds are based on a programmable digital signal processor (DSP) or a field programmable gate array (FPGA)-based SDR platform [4,5,8]. While such platforms can provide the necessary functions and high performance, the cost of development and the lack of flexibility hinder their use. Conversely, SDR based on a general-purpose processor is a promising way for development and configuration. GNU Radio [44] and universal software radio peripheral (USRP) [45] is one of the most widely used SDR platforms. In [6], the authors presented the performance of a DF system built using GNU Radio and USRP. They clearly demonstrated an improvement in bit-error-rate (BER) by using DF, but their transceiver design allowed only a single transmission per time slot due to the challenges of synchronizing multiple transmitting nodes. Finally, in [7], the authors presented multi-relay transmission using GNU Radio and USRP. While this paper demonstrated significant performance gains by simultaneous transmission from multiple relays, it required a wired connection to provide reference clocks for time and frequency synchronization. Furthermore, most existing experimental research studies for CCT using SDR have been focused on evaluating the performance of the physical layer,

because there is a lack of multi-layer supports in existing SDR platforms.

CHAPTER 3

IMPLEMENTATION ON SOFTWARE-DEFINED RADIO TESTBED

3.1 Overview

This chapter focuses on implementation of SDR-based testbed that includes all physical layer algorithms designed in this research and implementation of a modular protocol stack. The main objective is to design of a fully self-contained wireless node that performs real-time signal processing, synchronization, and control systems in a SDR-based software and hardware platform. In addition, another objective is to design a software framework that can run on the implemented wireless node to evaluate performance of CCT-based higher layer protocols (e.g., broadcasting, routing and network-time synchronization) in real-world environments. Since a SDR system allows us to design any transceivers in software, it can be an effective solution to investigate real-world issues of cooperative communication systems. A SDR system generally consists of a RF front-end, analog-to-digital converter (ADC), digital-to-analog converter (DAC), digital down-converter (DDC), and digital up-converter (DUC) which are implemented in hardware, and a programmable baseband-processor such as a personal computer (PC) or a FPGA as shown in Figure 4. Since the baseband signal in software is digitally sampled in hardware from/to analog signal, all transceiver algorithms need to be designed in the digital domain, sometimes referred to as the sample domain.

A family of lightweight broadcasting and routing protocols based on a simple form of CCT called the opportunistic large array (OLA) have been developed in [35, 38, 39]. For evaluating performance of the protocols in network-layer point of view, MATLAB or customized simulation tools have been used in the literature. However, the use of simulation alone is less attractive than an experimental testbed because a testbed can reveal practical

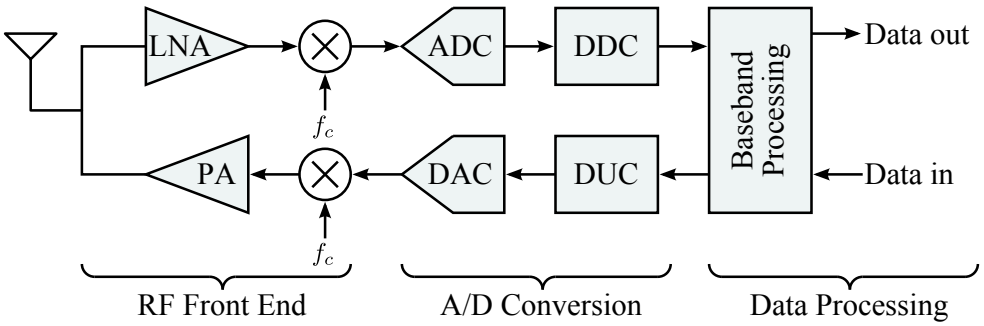


Figure 4: Block diagram of a generic SDR.

limitations of the protocols and represents actual wireless channel environments. To evaluate the performance of CCT-based network-layer protocols, we designed a programmable and modular protocol stack that can be used with the SDR testbed.

This chapter is organized as follows. Section 3.2 introduces the SDR platform for the testbed design. In Section 3.3, the non-deterministic processing time for the relay operation is discussed. In Section 3.4, a time-management solution for USRP1 is developed, which is seminal to achieve pre-synchronization of cooperative relays. In Section 3.5, the design of Python protocol stack (PPS) is described and an example of the usage of PPS is provided. Section 3.6 summarizes this chapter.

3.2 Implementation on GNU Radio and USRP

For the implementation, GNU Radio is selected for the SDR software platform [44]. GNU Radio is an open-source software toolkit that provides signal-processing blocks to implement software radios. It can be used with readily-available low-cost external RF hardware to create SDRs, or without hardware in an emulation environment. Currently, GNU Radio is widely used in hobbyist, academic and commercial environments to support both wireless communications research and real-world radio systems. GNU Radio applications are primarily written using the Python programming language, while the supplied

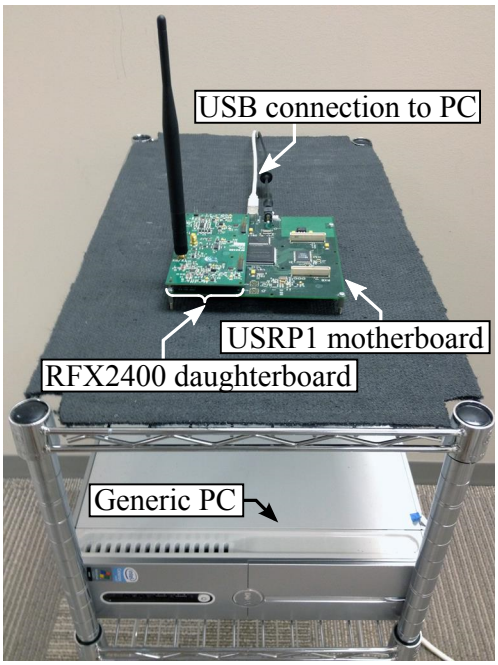


Figure 5: Photograph of a software-defined radio node.

performance-critical signal-processing path is implemented in C++ using processor floating-point extensions, where available. Thus, the developer is able to implement real-time, high-throughput radio systems in a simple-to-use, application-development environment.

GNU Radio utilizes the Universal Software Radio Peripheral (USRP) developed by Ettus Research [45] as the RF front-end and the converters. In particular, the first and second generation of USRP (i.e., USRP1 and USRP2) are used for the implementation and experimental studies reported here. As a RF front-end, RFX2400 and WBX daughterboards are used to provide filtering of the RF signal and conversion from RF to IF and vice-versa. The RFX2400 operates in the 2.4 GHz to 2.483 GHz range with a peak output power of 50 mW, while WBX operates in the 50 MHz to 2 GHz range. The output signal from a daughterboard is then connected to the USRP motherboard, where it is sampled by the ADC and then be converted to baseband by the DDC implemented in the on-board FPGA. The transmission path is similar, but consists of a DUC and a DAC. The baseband signal sampled on the USRP board is now sent via USB interface or Ethernet connection to the host computer where the GNU Radio software is running.

Figure 5 shows a SDR node that consists of a USRP1, RFX2400, and a generic PC. The SDR node is installed in a cart so that it can be moved on a floor easily for various topologies in an indoor environment.

3.3 Unpredictable processing time in SDR

In a cooperative relaying system, the packet processing time can be defined to start when the end of the source-transmitted packet arrives at the relay's antenna and ends when the start of the relayed packet is transmitted at the antenna. The processing time involves extensive signal processing including packet decoding/encoding, checking error, parsing a header, etc. As discussed in Section 2.3, the signal processing time is not deterministic in most practical radio systems because of operating clock offset and non-real-time characteristics of software executed on an operating system (OS). In conventional off-the-shelf radios, high-layer protocols are implemented by software that runs on a processor while the physical-layer protocol is implemented in a communication chipset by hardware logics. Some off-the-shelf radios use a real-time operating system (RTOS) which is an OS intended to serve real-time applications to reduce the variance of the processing time. Even though a RTOS is designed to meet a processing deadline deterministically, the time jitter in a RTOS is still order of milliseconds [46], which is relatively large compared to the typical propagation delay in an indoor environment.

The variance of the processing time is larger in SDR systems than in conventional radio systems because all baseband processing is performed by software. Figure 6 shows a block diagram of the USRP1 and GNU Radio. As shown in the figure, USRP1 is composed of a motherboard and a daughterboard. The motherboard contains a FPGA, ADC, DAC, and USB controller etc., while the daughterboard serves as RF front-end. The signal sampled at the USRP1 is transferred to a PC for the baseband processing. The figure shows a latency in each component. The processing time for a relay operation, can be defined as the sum, $\sum_{k\in\{R,T\}} \Delta_{kUSRP} + \Delta_{kUSB} + \Delta_{kBus} + \Delta_{kBlocks}$. Among those latencies, Δ_{RUSRP} and Δ_{TUSRP} can be

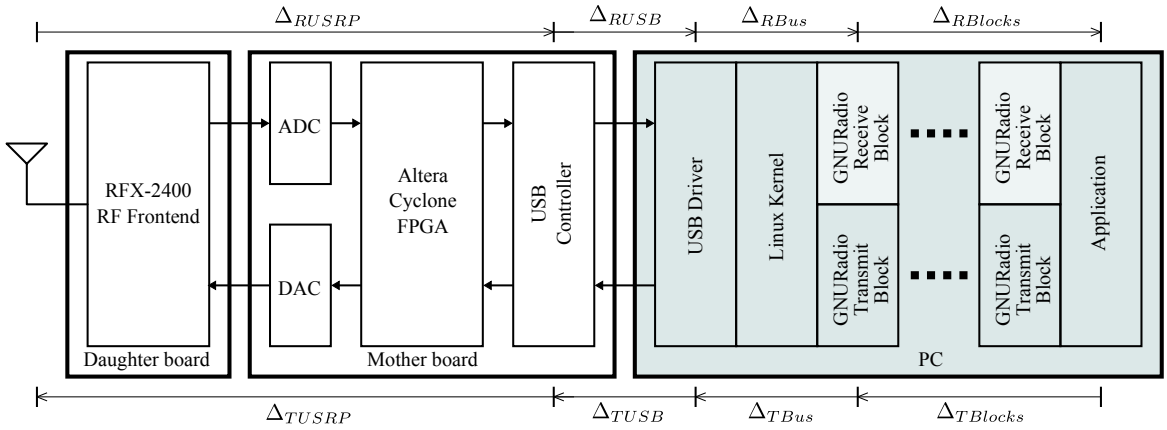


Figure 6: Block diagram of a USRP1 and GNU Radio system and unpredictable latencies caused by USB polling (Δ_{USB}), OS scheduling (Δ_{Bus}) and buffering (Δ_{Blocks}).

considered as deterministic because the dedicated FPGA and other hardware components have a fixed group delay. The sum of the remaining terms, which is referred to as the random processing time, is highly unpredictable and its statistical properties depend on various random factors.

The minimum and maximum values of the measured processing times when binary frequency-shift keying (BFSK) modulation/demodulation is used, are shown in Table 2. A 2.2 GHz Intel Celeron Processor with 2 GB RAM was used for the measurement. The results show that the random processing times vary on the order of milliseconds, where the variance depends on the size of a buffer in each GNU Radio block and the sampling rate.

Table 1: Measurement of ρ in GNU Radio and USRP1.

Block buffer sizes	32 KB		4 KB	
Sample rates	1.28 Ms/s	2.56 Ms/s	1.28 Ms/s	2.56 Ms/s
Min ρ	19.6 ms	16.4 ms	13.1 ms	10.7 ms
Max ρ	51.2 ms	33.2 ms	17.1 ms	14.0 ms

3.4 Time Management in USRP1

To realize pre-synchronization in a SDR system, the SDR node needs to manage its received and transmitted signal in time. USRP provides a method for time management by timestamping and burst-flagging through USRP Hardware Driver (UHD). While the time

management solution in USRP2 has high accuracy because it is controlled by hardware, the one for USRP1 is software-emulated. In USRP1, the clock is implemented in the PC and used for a reference time to timestamp received samples and to schedule transmission time. In a USRP1 board, the common operating clock is provided by a local oscillator to the ADC, DAC, and the FPGA. Therefore, all digital processing in the USRP1 board is deterministic in time. However, because the PC and the FPGA are connected by USB connection in USRP1, the PC's clock is not aligned to the FPGA's clock. In other words, if the PC were to decide that the time to fire is "now!", the FPGA would get the command some long and random time later, because of the unpredictable processing time and the slow and uncertain timing of the USB transmission.

To avoid using the software-emulated clock in USRP1, we implemented a time-management block in a USRP1's FPGA. One of the objectives of this block is to insert a time-tag signal samples so that a signal-processing block in a PC can specify the time of a special moment such as a start-of-packet. The other objective is to enable the USRP1 to send packets at a specific time. A 64 Mhz crystal oscillator runs on the USRP1 and provides an operating clock to the FPGA, ADC, DAC, and USB controller. The clock rate is down-sampled to make the sampling rate f_g which is used in DDC in the FPGA. To implement the time-tag method, a 32-bit hardware counter with the clock rate f_g, was built into the FPGA as shown in Figure 7. The event trigger and multiplexer in the figure inserts the lower 16 bits and the upper 16 bits of the counter value, called "rx time-tags," into the I and Q data streams, respectively, at every G samples just after the 16-bit time-tag indicator, 0xFFFF[1], as shown in Figure 8. The 16-bit indicator is used to identify the time-tag in the following data processing block. The time-tagging block and the 32-bit counter are programmed in VHSIC (Very High Speed Integrated Circuit) Hardware Description Language (VHDL).

[1] *The prefix "0x" represents a hexadecimal number.*

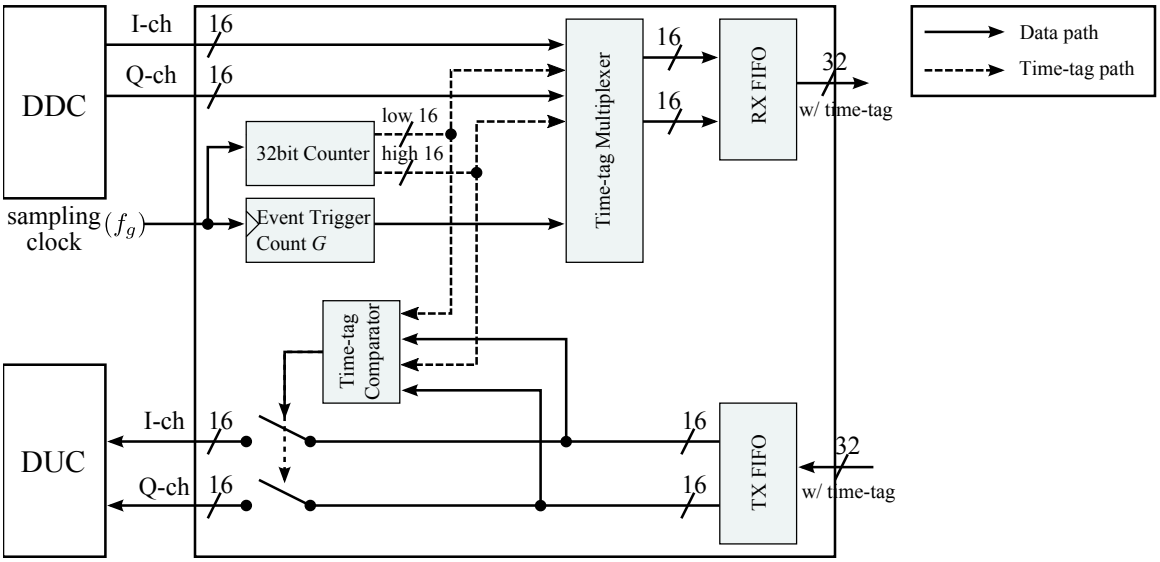


Figure 7: Block diagram for the proposed in-band processing in the FPGA of a USRP1 board.

If the sampled I/Q data is identical to the indicator value, the data should be modified in its least significant bit (LSB) to 0xFFFE to avoid the ambiguity with the signature and the sampled value. It is noted that this happens very rarely because the sampled data becomes 0xFFFF only when the sampled signal is saturated. Periodicity G can be controlled by the user with a register implemented in the FPGA. It is noted that the time difference in seconds between consecutive time-tags is G/f_g.

For transmission, the message to be transmitted is encoded and modulated to discrete samples. Before sending the discrete samples to the USRP1, a time-tag, that corresponds to desired transmit time, is attached to the front of the transmit samples by the "time-tag attacher" block which is described in the following subsection. In the FPGA of the USRP1, the attached "tx time-tag" is compared to the 32-bit counter value as shown in Figure 7. The transmission will be held until the "tx time-tag" matches the counter value. If the time-tag has expired, "time-tag comparator" in the figure will flush out the entire packet.

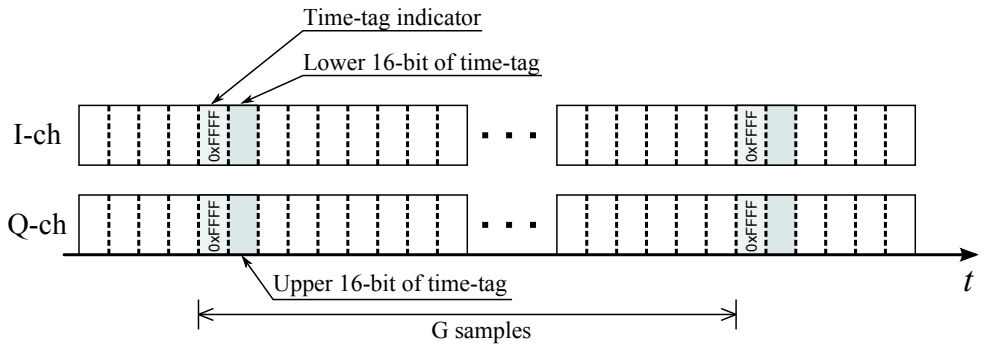


Figure 8: Illustration of the "time-tags" insertion in a in-band stream.

3.5 Python Protocol Stack

In a SDR system, signal-processing blocks are implemented in software so that any physical layer algorithm can be realized without hardware constraints. The programmable nature of a SDR system inspires the implementation of a programmable protocol stack, which can run on the top of the SDR system. A protocol stack is widely implemented in software-implemented network simulators [47, 48] to support various and programmable protocols. Such network simulators, however, cannot be easily combined with a SDR system because they are built in different frameworks. From this fact, a modular/programmable protocol stack, named as Python Protocol Stack (PPS), is designed to evaluate CCT-based network layer protocols in actual wireless environments. PPS is a full-featured network protocol stack that is tightly coupled with GNU Radio framework and is written in Python programming language. The design objective of PPS is to create a GNU Radio-based protocol framework that is structured much like the actual network protocol stacks used in operating systems or mobile network devices.

PPS consists of two major Python classes: protocol class and packet class. The protocol class is a mother class of a protocol implementation while the packet class is designed for packetization and de-packetization of data traveling through multiple protocol layers.

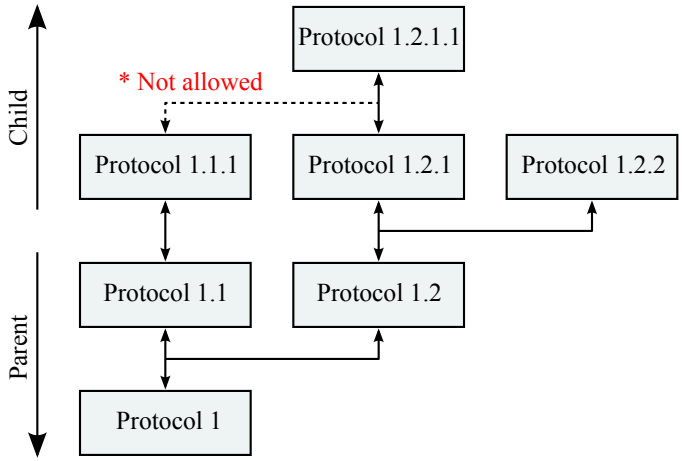


Figure 9: Example of connecting multiple protocol classes.

3.5.1 Protocol Class

PPS has clear and distinct separation between protocol layers. Each layer can be programmed by a user and be replaced with another easily. The protocol layer is constructed as a Python class, in which the class is inherited from the mother class *pps_protocol*. *pps_protocol* has two pointer variables *parent* and *child*. The pointers in the protocol can indicate any other protocol classes to construct the hierarchical stack. Figure 9 shows an example of connecting multiple protocol classes. In the figure, upper protocols are children of lower protocols. It is pointed out that a protocol is allowed to have a single parent protocol while it can have multiple children protocols. In an open systems interconnection (OSI) model, a parent protocol represents a lower layer protocol. For example, Protocol 1 in the figure is a physical layer protocol while Protocol 1.1 and Protocol 1.2 are link layer protocols.

The protocol class has two major functions: *data_request* and *data_indication*. *data_request* is used to transmit data, in which the data is transferred to the parent protocol class. When the function is called, the protocol processes the packet and delivers it to the parent by calling the *data_request* function of the parent protocol. Therefore, the calling *data_request* function in the highest protocol automatically invokes calling *data_request* in the all protocols within the connection. *data_indication* is called in the reverse direction. When the

lowest protocol is notified that there is a received packet from a signal-processing block, it processes the packet and passes it to its children by calling their *data_indication* function. Since a protocol can have multiple children, multiple *data_indication* functions might be called within a parent protocol. For example in the figure, once Protocol 1.2 has a received packet from Protocol 1, it will call the *data_indication* function in both of Protocol 1.2.1 and Protocol 1.2.2.

3.5.2 Packet Class

In PPS, a packet is managed by the Python class, named *pps_packet* in PPS. *pps_packet* consists of a set of protocol data units (PDUs) that are appended and removed as the packet moves up and down through the protocol stack. The packet class also has a metadata that contains the information related to reception (e.g., received signal strength indicator (RSSI) value, SOP time, CFO, and etc.) and transmission (e.g., orthogonal sub-channel , scheduled transmit time, etc.)

3.5.3 Building a protocol stack

Building and running a protocol stack in PPS, requires creating and connecting class objects. List 3.1 shows an example of how to create protocol classes and how to connect them together.

- **Creating class objects**

  In the example, *pps_phy*, *pps_mac*, *pps_net*, and *pps_udp* classes are created by being inherited from *pps_protocol* class. For *pps_mac* and *pps_net* class, the MAC and IP address arguments are needed to create the class. To create *pps_udp* class, a port number is required. The port number corresponds to the UDP socket to be opened.

- **Connecting classes**

  All protocol classes can be connected by the function *pps_connect*. In this function, a former argument becomes a lower layer protocol. Through this operation, all parent and child pointers in each protocol class are connected to each other.

Listing 3.1: Example codes for creating and connecting protocol stacks

```
1 #=============================================
2 # Create a Physical layer (Open GNURadio Core)
3 #=============================================
4 phy = pps_phy()
5
6 #=============================================
7 # Create a MAC/Link layer
8 #=============================================
9 mac = pps_mac(MACAddr)
10
11 #=============================================
12 # Create a Network layer
13 #=============================================
14 net = pps_net(IPAddr)
15
16 #=============================================
17 # Create a UDP socket
18 #=============================================
19 socket = pps_udp(PortNumber)
20
21 #=============================================
22 # Bind the MAC/Link and network layer
23 #=============================================
24 pps_connect(phy,mac,net,socket)
25
26 #=============================================
27 # Send a packet
28 #=============================================
29 socket.connect( IPAddr_Dest, PortNumber_Dest )
30 socket.data_request( 'Hello!' )
```

- **Sending a message**

  If the node is configured as a source node, the connect() function in *pps_udp* should be called prior to send a message as shown in line 29. The arguments of the function connect() are the IP address and port number of a sink node. After the UDP connection, the *data_request* function can send a message to the sink node.

Once all classes are created and connected correctly, a user can send and receive a message through the *pps_udp* class. PPS provides end-to-end connections between two nodes at the transport layer that are specified using a tuple of source IP, source port, destination IP, and destination port similar to the ubiquitous sockets API in Linux/Unix environments.

Therefore, the user can communicate with another user through just accessing *pps_udp* object.

3.6 Summary

In this chapter, the SDR testbed based on GNU Radio and USRP systems was designed for CCT. In addition, PPS, that is a modular protocol stack written in Python programming language, was introduced with the example of how to create and use it. The USRP1's FPGA code has been modified to support a real-time process for receiving and transmitting that is required for CCT. PPS is implemented in such a way that it can be worked with the SDR testbed developed for CCT. PPS is also designed to provide modularity where each layer protocol can be replaced with another on the fly so that different protocols can be compared with each other under nearly the same condition.

CHAPTER 4

PRE-SYNCHRONIZATION FOR CONCURRENT COOPERATIVE TRANSMISSION

4.1 Overview

While multiple antennas are perfectly synchronized in a conventional MIMO system by sharing the same reference clock across the antennas, spatially separated radios form a virtual antenna array in CCT causing time and frequency offsets between distributed antennas. Assuming there is no additional resource to provide a reference clock to the distributed relays, a pre-synchronization method is developed in such a way that the reference for synchronization comes from a packet, transmitted by the previous virtual array and simultaneously received by all the cooperative transmitters. In this chapter, we propose a method to estimate the reference time based on start-of-packet (SOP)-time estimation. In addition, we also design a method to combine multiple SOP-time estimates to minimize the variance of retransmission error for multi-hop CCT. The combined SOP-time is derived in such a way that the variance of transmission error in each hop is statistically convergent when the proposed combining rule is used. The claim has been proved through a computer simulation in this chapter, and through experiments in the sequel chapters.

The remainder of this chapter is organized as follows. Section 4.2 describes a timing model that is used in the following sections. In Section 4.3, we propose a pre-synchronization algorithm based on SOP-time estimation and a fixed amount of processing time T_{proc}. We develop the combining rule and analyze the convergence property for multi-hop CCT in Section 4.4. In Section 4.5, the convergence property of the combining rule is verified using a Monte-Carlo simulation, and Section 4.6 summarizes this chapter.

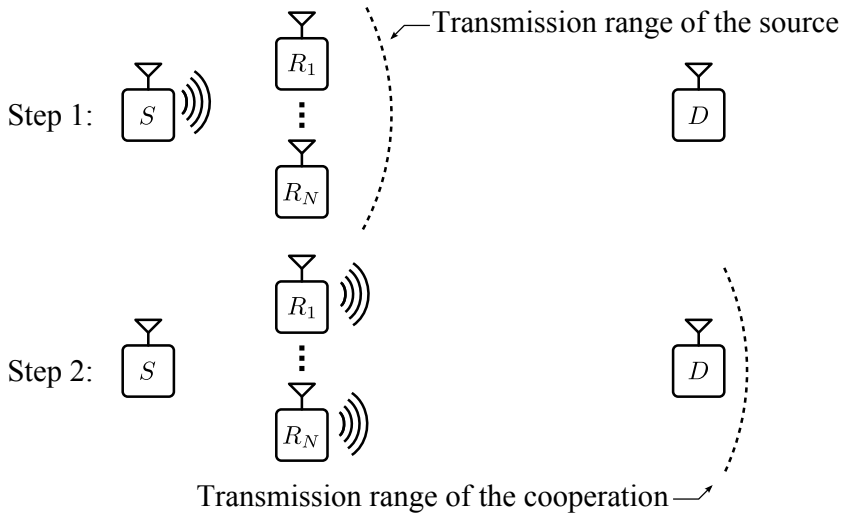


Figure 10: Illustration of CCT in a two-hop network.

4.2 Practical Timing Model

Without being assisted by external references, a distributed relay can derive its transmit time autonomously by estimating the SOP-time of a received signal, in which the reference signal for pre-synchronization comes from a single transmission or simultaneously CCTs from multiple transmitters in a previous cluster. Figure 10 illustrates an example of a two-hop network that uses CCT to relay the source message. Since CCT is a relaying scheme, it consists of two steps: 1) distribution phase and 2) cooperation phase. In the first step, the source, which is denoted as 'S' in the figure, broadcasts or multicasts a source message to the cooperators $R_1, \cdots, R_N$ which are located within its transmission range. Once the source message is received and correctly decoded by the cooperators, they retransmit the source message to the destination node creating a virtual antenna array. In the proposed scheme, the source message transmitted from the source node is used as a reference for the synchronization.

Ideally, synchronous cooperating transmitters start their transmissions, or "fire," at the same time. However, several random phenomena cause the firing times to be slightly different, giving them a non-zero variance as discussed in Section 3.3. In this section, the variable times for these random phenomena are defined.

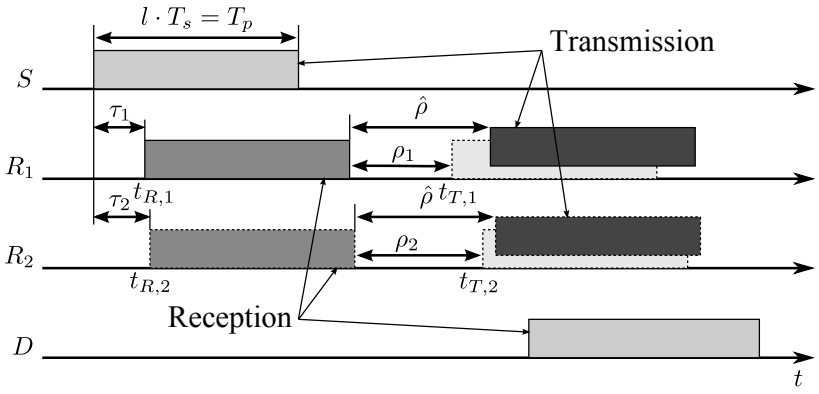


Figure 11: Timing diagram for the two-hop scenario.

Figure 11 shows the timing diagram for the two-hop scenario that is illustrated in Figure 10. The top trace represents the source packet, which is assumed to have the deterministic duration $T_p = l \cdot T_s$, where T_s is the sample period of the source clock and l is the number of samples in the source packet. Let $t_{R,i}$ be the time that the source-transmitted waveform arrives at the antenna of the i^{th} relay node; this time is named as SOP time. The estimate of $t_{R,i}$ can be defined as

$$\hat{t}_{R,i} = T + \tau_i + \omega_i, \tag{8}$$

where τ_i is the propagation time between the source and relay i, and ω_i is the SOP-time estimation error.

When node R_i relays the source packet, the retransmission time can be defined as

$$\begin{aligned} t_{T,i} &= \hat{t}_{R,i} + T_p + \rho_i \\ &= T + \tau_i + \omega_i + T_p + \rho_i. \end{aligned} \tag{9}$$

Assuming that the statistics of all propagation times, packet-detection errors and processing delays do not vary across the relays, the sample variance of retransmission time can be written as

$$s_{t_T}^2 = \sigma_\tau^2 + \sigma_\omega^2 + \sigma_\rho^2, \tag{10}$$

where σ_τ^2, σ_ω^2 and σ_ρ^2 are the variance of a propagation time, SOP-time estimation error and

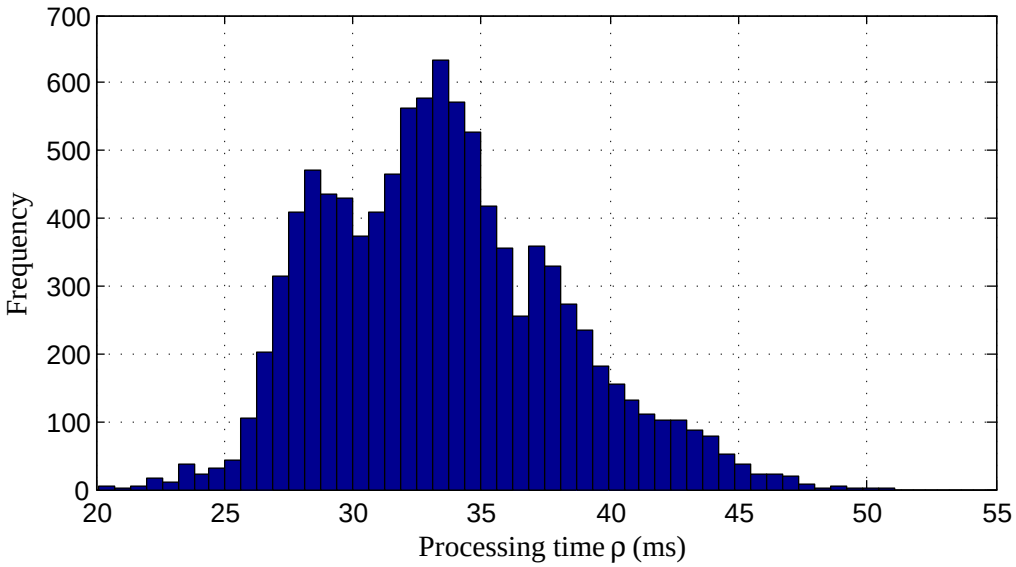


Figure 12: Empirical histogram of the processing time when $1/T_s$=1.28 Ms/s with 32 KB buffer.

processing delay respectively. Figure 12 shows that empirical histogram of the measured processing time when $1/T_s = 1.28$ Ms/s with 32 KB buffer size. The result shows that the variance of the processing time $\sigma_\rho^2 \approx 20.9$ ms. Furthermore, the measurement also informs that the statistical characteristic of the processing time does not follow a standard distribution. From (10), the variance of the retransmission time can be reduced by lowering the variance of each element. In this research, it is assumed that the nodes do not move so that the propagation time is deterministic in each node. It is also assumed that the propagation delay difference between relays is negligibly small. In following section, a method to reduce the variance σ_ρ^2 will be discussed while the variance of the SOP-time estimation will be discussed in the following chapters.

4.3 Time Synchronization with T_{proc}

To reduce the variance of the processing time σ_ρ^2, it is proposed that all cooperating nodes wait to transmit for a fixed period T_{proc} after the end of the source-transmitted packet arrives at the antenna. In symbols, the i^{th} node will transmit at time $\hat{t}_{R,i} + T_p + T_{proc}$. T_{proc} is selected so that all the processing has been done and transmission is ready when the time comes to

fire with $\alpha \times 100\%$ of probability, where α is a design parameter such that $0 \leq \alpha \leq 1$. Therefore, T_{proc} can be expressed as

$$T_{proc}(\alpha) = \arg\min_{t \in mT_s} \{Pr(\rho < t) \geq \alpha\}, \tag{11}$$

where $m \in \mathbb{Z}$. $T_{proc}(\alpha)$ should be determined prior to network operation.

In each relay, the designed parameter T_{proc} is measured by counting its own clock. However, another concern is that because the different each have a different clock, nodes with a fast clock will have T_{proc} expiring before the nodes with a slow clock. Let r_i be the ratio of the clock frequency of relay i normalized by the clock frequency of the source node. Without compensation of the discrepancy of the clock ratio, the actual amount of T_{proc} in each relay is given by

$$T_{proc,i} = T_{proc}/r_i. \tag{12}$$

Each relay can estimate the clock ratio r_i with the estimation error ϵ_i as

$$\hat{r}_i = r_i + \epsilon_i. \tag{13}$$

After compensation of the clock ratio based on the estimate $\hat{r}_i$, the processing time in (9) can be rewritten as

$$\begin{aligned} \rho_i &= T_{proc,i} \cdot \hat{r}_i = T_{proc,i}(\hat{r} + \epsilon_i) \\ &= T_{proc} + T_{proc} \cdot (\frac{\epsilon_i}{r_i}) \\ &= T_{proc} + \xi_i, \end{aligned} \tag{14}$$

where $\xi_i = T_{proc} \cdot (\epsilon_i/r_i)$. Therefore, the variance of the retransmission time $\tilde{t}_{T,i}$ that is scheduled by T_{proc} in (10) can be expressed as

$$s^2_{\tilde{t}_T} = \sigma^2_\tau + \sigma^2_\omega + \sigma^2_\xi. \tag{15}$$

It is pointed out that the variance of the clock ratio estimation error σ^2_ξ is much less than the variance of the processing time σ^2_ρ.

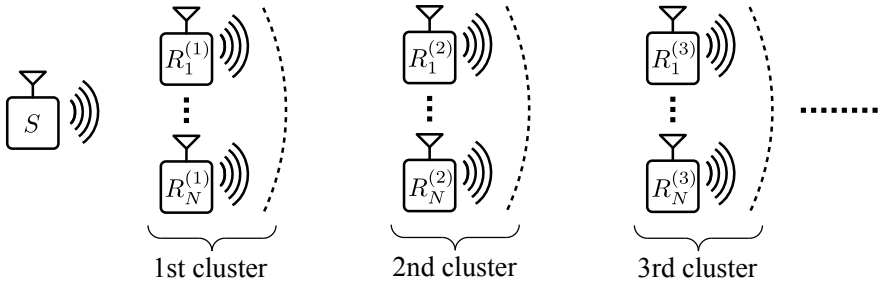


Figure 13: Illustration of consecutive CCT in a multi-hop network.

4.4 Multi-hop Concurrent Cooperative Transmission

In this section, consecutive CCT is considered, in which groups of multiple radios are formed as "clusters" to build a multi-hop network as shown in Figure 13. In this scenario, the clusters can be predetermined [37] or determined on-the-fly [38]. It is assumed that, in this topology, the second and third cluster cannot listen the source message. Therefore, the reference signal for the synchronization for each of those clusters must be the message transmitted by the previous cluster. The estimation error of SOP time at each relay in a sequence of relay clusters is analyzed where each cluster has N relays and does CCT.

The following assumptions are made for this objective.

- A packet is originally transmitted by the single-antenna source (S) at time T, as shown in Figure 14 for $N = 2$.
- The nodes in each cluster are co-located (not shown this way in the figure) and the clusters are arranged on a line with equal spacing. This assumption implies that the propagation distances between nodes in different clusters are equal and deterministic.
- Each relay in a cluster transmits a preamble that is orthogonal to the other $N - 1$ preambles being transmitted in the cluster. These orthogonal preambles define the N diversity channels for synchronization, and are transmitted simultaneously.
- Each receiver has a bank of N correlators, with one correlator for each diversity channel.

- All diversity channels exhibit independent Rayleigh fading.

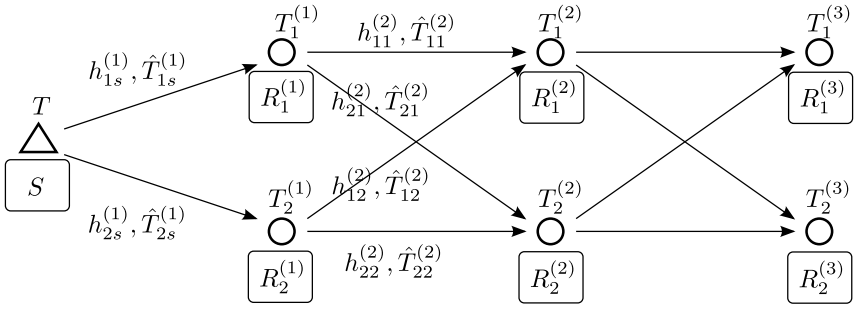


Figure 14: System model for consecutive CCT.

For all terms used in this section, a superscript and subscript represent the cluster and node index, respectively. For example, $T_k^{(j)}$ is a time of transmission of Relay k in cluster j. For example of two subscripts case, $h_{21}^{(j)}$ is a channel between Relay 1 in cluster $j-1$ and Relay 2 in cluster j. $\hat{T}_{kn}^{(j)}$ is an estimate of $T_n^{(j-1)}$ based only on correlation in channel $h_{kn}^{(j)}$ at the Node k in the cluster j. $\omega_{kn}^{(j)}$ is the "correlator estimation error" of $\hat{T}_{kn}^{(j)}$ transmitted from the Node n of previous cluster. $\hat{T}_k^{(j)}$ is referred to as a combined SOP time that is a combination of $\hat{T}_{kn}^{(j)}$ for $n = 1, \ldots, N$. The strategy to combine N SOP times will be discussed in the sequel section. $\xi_k^{(j)}$ is a sampling clock compensation error of the Node k in the cluster j.

The k^{th} relay node in the j^{th} cluster autonomously estimates the SOP time $\hat{T}_{kn}^{(j)}$ for N transmitters from the $(j-1)^{th}$ cluster and calculates the combined SOP time $\hat{T}_k^{(j)}$. Next, *ideally*, each relay determines its transmit time by adding an universally known constant, T_{proc}, to $\hat{T}_k^{(j)}$, to allow enough time for the signal processing on all relays in a cluster to be completed. As discussed in Section 4.2, the deviation in T_{proc} expiration on Node k of cluster j from an imaginary universal reference clock is designated as the "clock error," $\xi_k^{(j)}$. The sampling clock error $\xi_k^{(j)}$ is assumed to be zero mean and Gaussian random variable with a variance σ_ξ^2. Therefore, the transmit time of the k^{th} relay in the j^{th} cluster is given by

$$T_k^{(j)} = \hat{T}_k^{(j)} + T_{proc} + \xi_k^{(j)}. \tag{16}$$

4.4.1 Approximated Best Linear Unbiased Estimation

In this subsection, we design an approximated best linear unbiased estimator (ABLUE) of the combined SOP-time $\hat{T}_k^{(j)}$. The performance of time synchronization for CCT can be assessed by a covariance of transmit time $T_k^{(j)}$ for $k = 1, \ldots, N$. From (16), the variance of the retransmission time $T_k^{(j)}$ is sum of the variance of the combined SOP-time estimation error $\hat{T}_k^{(j)}$ and the variance of the clock error. Assuming that the statistical characteristic of the sampling clock error is the same across all the nodes, the sample variance of the retransmission time error can be minimized by minimizing the variance of the combined SOP-time estimation error.

To compute the combined SOP-time estimate $\hat{T}_k^{(j)}$ that is the estimate of the SOP time at the k^{th} node in the j^{th} cluster, the node first creates N estimates, $\hat{T}_{kn}^{(j)}$ for $n = 1, 2, \ldots, N$, of the SOP time in each of its N diversity channels. Since only one of the previous cluster nodes transmits in a given diversity channel, $\hat{T}_{kn}^{(j)}$ is actually an estimate of the SOP of the packet received from $R_n^{(j-1)}$ as

$$\hat{T}_{kn}^{(j)} = T_n^{(j-1)} + \omega_{kn}^{(j)}. \tag{17}$$

Then the combining rule is designed in such a way that $\hat{T}_k^{(j)}$ is a linear combination of $\hat{T}_{kn}^{(j)}$ for $n = 1, \ldots, N$ as

$$\hat{T}_k^{(j)} = a_{k1}^{(j)} \cdot \hat{T}_{k1}^{(j)} + a_{k2}^{(j)} \cdot \hat{T}_{k2}^{(j)} + \ldots + a_{kN}^{(j)} \cdot \hat{T}_{kN}^{(j)} \tag{18}$$

where $a_{kn}^{(j)}$ is a coefficient of the combiner. By BLUE, and under the assumption that the average transmission-time errors of previous cluster nodes are same, the coefficient vector becomes

$$\boldsymbol{a}_k^{(j)} = [a_{k1}^{(j)}, a_{k1}^{(j)}, \ldots, a_{kN}^{(j)}]^T = \frac{\mathbf{V}^{-1}\mathbf{1}}{\mathbf{1}^T\mathbf{V}^{-1}\mathbf{1}}, \tag{19}$$

where $\mathbf{V}$ is a covariance matrix of $\hat{\boldsymbol{T}}_k^{(j)} = [\hat{T}_{k1}^{(j)}, \hat{T}_{k2}^{(j)}, \ldots, \hat{T}_{kN}^{(j)}]^T$ and $\mathbf{1}$ is the column vector of N ones. Since single correlation errors $\omega_{kn}^{(j)}$ are uncorrelated over index n, the covariance matrix becomes a diagonal matrix $\mathbf{V} = diag\{\sigma_{\omega_1}^2, \sigma_{\omega_2}^2, \ldots, \sigma_{\omega_N}^2\}$.

It is pointed out that the error variance $\sigma^2_{\omega_k}$ is a system-dependent parameter that depends on a method to estimate an individual SOP-time $\hat{T}^{(j)}_{kn}$. Under the assumption that the error variance satisfies

$$\sigma^2_{\omega^{(j)}_{kn}} \propto \frac{1}{\mathrm{SNR}^{(j)}_{kn}}. \tag{20}$$

Assuming that the noise variances over all the nodes in the network are identical, then the estimator coefficient becomes

$$a^{(j)}_{kn} = \frac{|h^{(j)}_{kn}|^2}{\left(\sum_{i=1}^{N} |h^{(j)}_{ki}|^2\right)}. \tag{21}$$

In the following section, it will be shown that the above combiner makes the multi-hop CCT statistically convergent over hops. Furthermore, in the following chapters, the SOP-time estimation algorithms will be designed to satisfy the above condition in narrowband and wide-band systems.

To simplify analysis, it is assumed that all deterministic times that include the propagation time and T_{proc} are equal to zero. This assumption enables focusing only on the behavior of time estimation *errors*. Since all the errors are zero mean, all estimators are trying to estimate the same thing, T, which is the original transmit time of the source (S).

4.4.2 Markov Process Model

Let the vector $\boldsymbol{T}^{(j)} = [T^{(j)}_1, T^{(j)}_2, \ldots, T^{(j)}_N]^T$ represent the transmit times of each relay in cluster j (or "hop j"). From the arguments above, each element of $\mathbf{T}^{(j)}$ is an unbiased estimate of the transmit time of the original packet, or $E\{\mathbf{T}^{(j)}\} = T \cdot \mathbf{1}$. Let the estimation error vector of $\boldsymbol{T}^{(j)}$ be denoted as $\boldsymbol{e}^{(j)} = [e^{(j)}_1, e^{(j)}_2, \ldots, e^{(j)}_N]^T$. The objective is to get a recursion for the covariance of transmit-time error $\mathbf{C}^{(j)} = E\{\mathbf{e}^{(j)}\mathbf{e}^{(j)^T}\}$, from which the sample variance is derived.

Because the coefficients of the estimator at cluster j do not depend on the multi-path channels of previous hops, it can be seen that the statistics of the errors at cluster j are independent of the errors of past hops, assuming $\boldsymbol{T}^{(j-1)}$ is given. In other words, the

conditional joint probability density function (PDF) of $\boldsymbol{T}^{(j)}$, given the entire past transmit times $\{\boldsymbol{T}^{(j-1)}, \boldsymbol{T}^{(j-2)}, \ldots, \boldsymbol{T}^{(1)}\}$, is the same as the conditional joint probability density function (PDF) of $\boldsymbol{T}^{(j)}$, given only the latest vector of transmit times $\boldsymbol{T}^{(j-1)}$. This fact makes $\boldsymbol{T}^{(j)},\ j = 1, 2, \ldots, N$, a vector-valued, continuous-state, discrete-time *Markov Process*. $\boldsymbol{e}_k^{(j)}$ with substitution of $\hat{T}_k^{(j)}$ and $\hat{T}_{kn}^{(j)}$ can be written as

$$\begin{aligned}\boldsymbol{e}_k^{(j)} &= \hat{T}_k^{(j)} + \xi_k^{(j)} - T = \sum_{n=1}^{N} (a_{kn}^{(j)} \hat{T}_{kn}^{(j)}) - T + \xi_k^{(j)} \\ &= \sum_{n=1}^{N} (a_{kn}^{(j)} (T_n^{(j-1)} - T + \omega_{kn}^{(j)})) + \xi_k^{(j)} \\ &= \sum_{n=1}^{N} a_{kn}^{(j)} e_n^{(j-1)} + \sum_{n=1}^{N} a_{kn}^{(j)} \omega_{kn}^{(j)} + \xi_k^{(j)}, \end{aligned} \tag{22}$$

noting that $\sum_{n=1}^{N} a_{kn}^{(j)} = 1$ for all j and k. Let $\mathbf{A}^{(j)}$ and $\boldsymbol{\omega}^{(j)}$ be an estimator coefficient and correlator error matrix whose elements of k^{th} column and n^{th} row are $a_{kn}^{(j)}$ and $\omega_{kn}^{(j)}$ respectively, and $\boldsymbol{\xi}^{(j)} = [\xi_1^{(j)}, \xi_2^{(j)}, \ldots, \xi_N^{(j)}]^T$. The covariance matrix $\mathbf{C}^{(j)}$ has an iterative form as

$$\begin{aligned}\mathbf{C}^{(j)} &= E\{\boldsymbol{e}^{(j)} \boldsymbol{e}^{(j)^T}\} = E\{\mathbf{A}^{(j)} \mathbf{C}^{(j-1)} \mathbf{A}^{(j)^T}\} \\ &+ E\left\{[[\mathbf{A}^{(j)} \circ \boldsymbol{\omega}^{(j)}]\mathbf{1} + \boldsymbol{\xi}^{(j)}][[\mathbf{A}^{(j)} \circ \boldsymbol{\omega}^{(j)}]\mathbf{1} + \boldsymbol{\xi}^{(j)}]^T\right\}, \end{aligned} \tag{23}$$

where $\circ$ denotes an element-wise product operator. In the following sections, it is shown that the expected value of the sample variance of estimation error $\boldsymbol{e}_k^{(j)}$ is convergent even though the absolute covariance matrix is not.

4.4.3 Convergence Property of Transmit-time Pre-synchronization

In this subsection, the asymptotic property of the Markov model in (22) is investigated. The performance of transmit time pre-synchronization error can be assessed by relative transmit time error of relay nodes. The relative transmit-time error of j^{th} cluster's relay nodes is equal to the sample variance of estimation error $\boldsymbol{e}^{(j)}$. To simplify analysis of a convergence property of the covariance matrix $\mathbf{C}^{(j)}$, it is assumed that the statistics of all channels, clocks and noises do not vary with relay index within a cluster. These assumptions yield

that the variance and the covariance of the estimation error are not functions of a node index. Therefore, the diagonal terms of the matrix $\mathbf{C}^{(j)}$, denoted as $C_d^{(j)}$, are identical. Also, the off-diagonal terms, denoted as $C_o^{(j)}$, are identical. In Appendix A, it is showed that $C_d^{(j)}$ and $C_o^{(j)}$ can be written as

$$C_d^{(j)} = \frac{2}{N+1} C_d^{(j-1)} + \frac{N-1}{N+1} C_o^{(j-1)} + H + N\sigma_\xi^2 \tag{24}$$

and

$$C_o^{(j)} = \frac{1}{N} C_d^{(j-1)} + \frac{N-1}{N} C_o^{(j-1)} \tag{25}$$

where

$$H = \sum_{n=1}^{N} E\{a_{gn}^{(j)} \omega_{gn}^{(j)}\}. \tag{26}$$

In [49], the expected value of sample variance in which the samples are not mutually independent and pairwise covariance is constant γ, is given by $E\{\sigma_s^2\} = \sigma^2 - \gamma$ where s^2 is the sample variance and σ^2 is the variance. From this relationship, the linear difference equation form of the expected value of the sample variance at j^{th} cluster can be obtained as

$$E\{\sigma_s^{2(j)}\} = \frac{N-1}{N(N+1)} E\{\sigma_s^{2(j-1)}\} + H + N\sigma_\xi^2.$$

The expected value of the sample variance converges to

$$E\{\sigma_s^2\} = \frac{N(N+1)}{N^2+1}(H + N\sigma_\xi^2) \text{ as } j \to \infty. \tag{27}$$

4.5 Simulation Study of Transmit-time Pre-synchronization

Monte Carlo simulation is used to check the convergence property of the transmit-time pre-synchronization when the combining rule that is proposed in Section 4.4.1 is used. In (27), it is theoretically proved that the expected value of the sample variance for the transmit-time error converges, while the variance and covariance for the error diverge as hop increases. In order to confirm the theoretical convergence property, the transmit-time error in (22) is calculated iteratively using Monte Carlo simulation. The following parameters are used for the simulation.

Table 2: Parameters for the Monte Carlo simulation.

Parameters	Values
N	2,4,8 and 16
SNR	10 dB
T	0 sec
T_s	1 us
K	2.446×10^{-11}
σ_{ξ}^2	10^{-12}

Figure 15 shows the divergence of $C_d^{(j)}$ and $C_o^{(j)}$ regardless of the number of cooperators N. Even though the slope of $C_d^{(j)}$ and $C_o^{(j)}$ decreases as hop increases, the variance and the covariance are still monotonically increasing. Figure 16 shows the convergence of the sample variance of the transmit-time error. As shown in the figure, the sample variance converges after the third hop. It can be observed that the sample variance keeps fluctuating as hop increases. This observation can be explained by the fact that the *expected value* of the sample variance converges as proved in (27). In other word, the sample variance is statistically convergent over hops. This convergence will also be demonstrated experimentally in the next chapter.

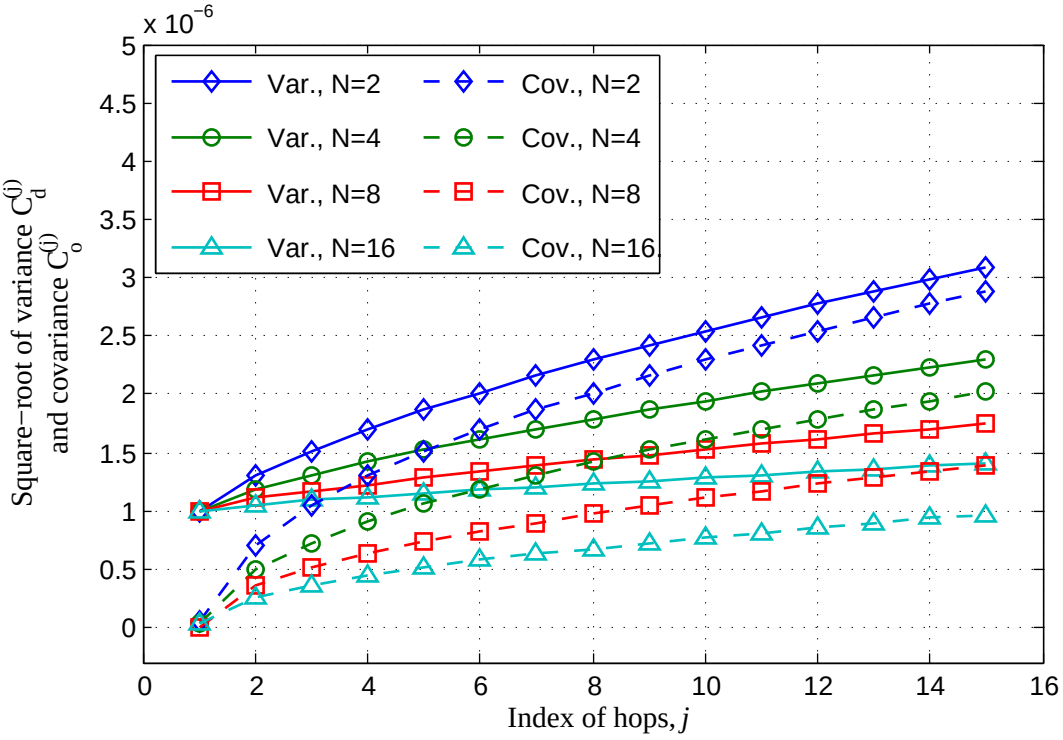


Figure 15: Square root of the variance $C_d^{(j)}$ and covariance $C_o^{(j)}$ of the transmit-time error in the Monte Carlo simulation when N=2, 4, 8 and 16.

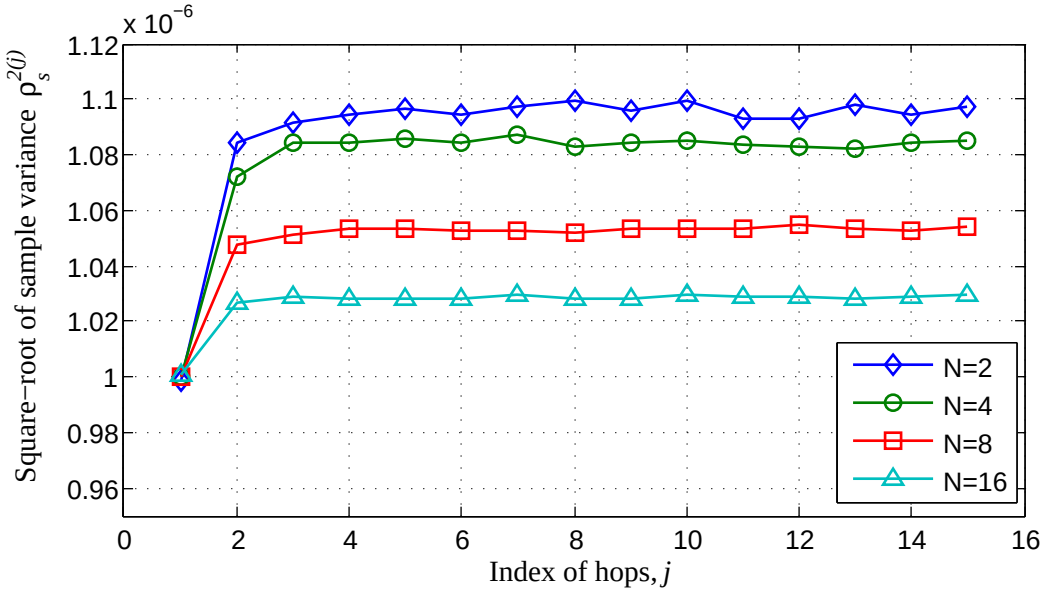


Figure 16: Square root of the sample variance σ_s^2 of the transmit-time error in the Monte Carlo simulation when N=2, 4, 8 and 16.

4.6 Summary

In this chapter, we designed the pre-synchronization algorithm for CCT based on SOP-time estimation and a universal constant T_{proc}. Instead of using a global clock, such as a global positioning system (GPS) receiver or a network time protocol (NTP) to provide reference signals for the synchronization, the reference for pre-synchronization comes from a packet, simultaneously received by all the relays, in which the packet is encoded with a form of transmit diversity. Also the combining multiple SOP-time estimates based on ABLUE is proposed to minimize the error variance of transmission time. It was theoretically shown that transmission time error in each cluster is statistically convergent over multi-hop CCT. The convergence property of the combining rule also has been verified by Monte-Carlo simulation. It is noted that, without loss of generality, the proposed ABLUE can be applied to pre-synchronize the transmit frequency as well for multi-hop CCT, as long as the CFO estimation error satisfies (20).

CHAPTER 5

TIME ESTIMATION FOR NARROW-BAND CONCURRENT COOPERATIVE TRANSMISSION

5.1 Overview

In this chapter, we develop a SOP-time estimation algorithm for a narrowband system using non-coherent binary frequency-shift keying (NCBFSK), which will approximate the analysis of the previous chapter. FSK modulation is known to enable a power efficient transmitter and a simple, low-cost receiver. In addition, a non-coherent design allows the receiver to estimate the SOP-time without having information of channel impulse response and CFO. Orthogonality for CCT is achieved in the frequency domain, by having different cooperating radios transmit on different orthogonal carriers. The center-of-mass (COM) estimation method is proposed for the robust and simple transceiver design in the presence of CFO causes distortion of the preamble correlation output. It is shown that the COM estimation of the combined correlation output approximates the combined SOP-time estimation proposed in Chapter 4 under the high-SNR assumption. The proposed design is evaluated though a computer simulation as well as experimental studies.

The remainder of this chapter is organized as follows. In Section 5.2, we design the SOP-time estimation algorithm using NCBFSK and the COM estimation method for the simple transceiver. Section 5.3 presents the initial experimental result of CCT where the cooperative diversity is used as a performance metric to show the SNR advantage of CCT. In Section 5.4, the SNR advantage of CCT is evaluated from range extension point of view. The convergence of multi-hop CCT is demonstrated through a "ping-pong" experiment that is designed to emulate a long CCT network or a long CCT route within a network in Section 5.5, and we summarize this chapter in Section 5.6.

5.2 Start-of-packet Time Estimation

Under ideal conditions, which implies non-CT, perfect carrier synchronization, and a static, flat-fading (or no-fading) channel, the estimation of SOP-time can be realized by cross-correlating the received signal with a known preamble or auto-correlating the consecutive preambles, and locating the peak of the correlation output. The baseband signal of BFSK in the k^{th} orthogonal channel with frequency separation Δf can be written as

$$s_{k,m}(n) = \sqrt{\frac{2\mathcal{E}}{T}} e^{j\pi\left\{\left(m(1/2)+k\right)\Delta f\right\}nT_s} \quad , n = 0, \ldots, L \tag{28}$$

where m denotes the symbol $\in \{-1, 1\}$, and $\mathcal{E}$ is transmit symbol energy. L is the number of samples in a symbol, and it is given by $L = T/T_s$ where T and T_s are symbol duration and sampling duration respectively. It is noted that m defines frequency separation between two symbols while k defines frequency separation between orthogonal sub-channels. Suppose that $\mathcal{E}$ is set to satisfy the unit energy as $\sum |s_k(n)|^2 dt = 1$, and all K relays transmit the same symbol so that the index m is taken out. The received signal $r(n)$ is a superimposed signal of K transmissions and has a frequency offset Δf_k which is introduced by two different carrier frequencies between the k^{th} transmitter and receiver. Assuming a quasi-static fading channel with a channel coefficient h_k by

$$r(n) = \sum_{k=1}^{K} \left\{ h_k \cdot s_{k,m}(n - mL - \tau_k) \cdot e^{j2\pi\Delta f n T_s} \right\} + w(n), \tag{29}$$

where τ_k and $w(n)$ are respectively a propagation delay normalized to the sampling duration T_s and zero-mean complex additive white Gaussian noise (AWGN) with a power spectral density of N_0. Since the preambles are transmitted through the orthogonal channels, it is noted that each of the transmitted preambles is orthogonal as

$$\sum_{n=1}^{N} s^*_{i,x}(n) \cdot s_{j,y}(n) = 0 \quad \text{if } i \neq j \text{ or } x \neq y \tag{30}$$

where N is the length of the preamble.

At the receiver, the received signal $r(n)$ passes through K pairs of BFSK envelope detectors which consists of matched filter banks and squarers as shown in Figure 21. $\tilde{\phi}^*_{(1,k)}$ and

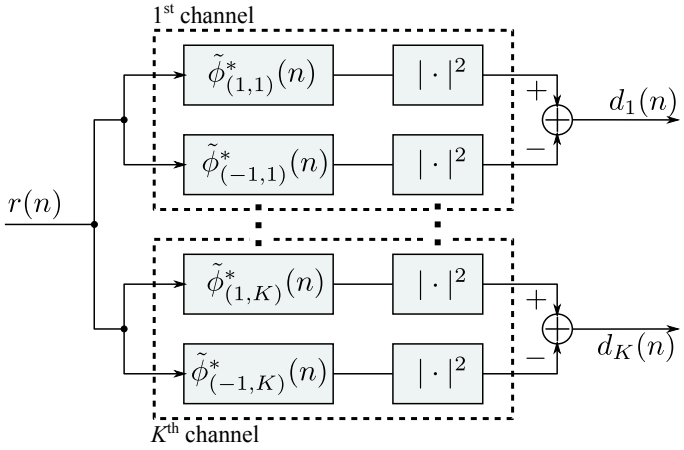


Figure 17: Non-coherent BFSK demodulator with orthogonal frequency diversity.

$\tilde{\phi}^*_{(-1,k)}$ are matched to symbol $\{1,-1\}$ for the k^{th} diversity channel, and combined to produce a bipolar, soft-valued output $d_k(n)$.

Mathematically, the soft-valued output for the m^{th} symbol in the k^{th} diversity channel can be written as

$$d_k(n,\tau,\Delta f_k) = \left|\sum_{l=0}^{L-1}\left\{r(n+l)\cdot s^*_{k,1}(l)\right\}\right|^2 - \left|\sum_{l=0}^{L-1}\left\{r(n+l)\cdot s^*_{k,-1}(l)\right\}\right|^2$$

for $n = (m-1)L,\ldots,mL-1$ where L is the number of samples in a symbol $L = T/T_s$. If the preamble is known a priori at the receiver, the beginning of the k^{th} preamble can be found by correlating the preamble sequence with the k^{th} soft-valued output $d_k(n)$. Let $\mathbf{p} = \{p_1,\ldots,p_P\}$ denote the preamble sequence where $p \in \{1,-1\}$. The correlation output of the k^{th} diversity channel can be written as

$$\Lambda_k(n) = \sum_{i=0}^{P-1} d_k(n+iL)\cdot p_i \quad \text{for } n = 0,\ldots,L-1. \tag{31}$$

Under the assumption of high signal-to-noise ratio (SNR) and $\Delta f_k = 0$ [50], the estimation of the SOP time for the k^{th} can be achieved by choosing $\hat{T}_k$ to maximize $\Lambda_k(n)$ as

$$\hat{T}_k = T_s \cdot \arg\max_n\{\Lambda_k(n)\}. \tag{32}$$

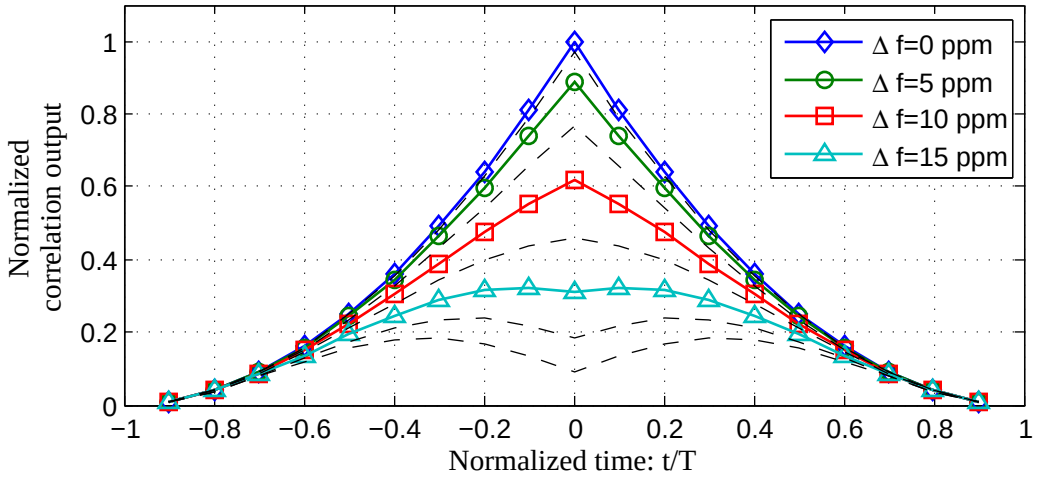


Figure 18: Normalized correlation output with different carrier frequency offsets.

5.2.1 Center-of-Mass Estimation

The accuracy of correlation rule is proportional to the SNR and side-lobe suppression of the correlation output [51]. Assuming that there is a CFO and it is not compensated prior to the SOP-time estimation, however, the peak detection suffers performance degradation because the CFO makes the filter mismatched. Example filter output shown in Figure 18, which is the decision function $\Lambda(n)$ over $(-T, T)$ where $h = 1$, $P = 1$, $\tau = 0$, and noise-free. It is shown that signal strength diminishes as CFO increases. Furthermore, the correlation output $\Lambda(n)$ becomes blunt and finally it becomes bimodal, which is not desirable for the peak detection.

Instead of searching for a peak of correlation output to determine a SOP time, a COM estimation is proposed in this subsection. Let $\Lambda_k(n)$ denote the preamble correlation output of k^{th} diversity channel. Without loss of generality, it is assumed that the propagation delay $\tau_k = 0$ for all k which implies that the truth time of SOP is 0. The SOP time of the k^{th} relay can be estimated by finding a mean of $\Lambda_k(n)$ as

$$\hat{T}_k = T_s \frac{\sum_{n=-L}^{L} n \cdot \Lambda_k(n)}{\sum_{n=-L}^{L} \Lambda_k(n)}. \tag{33}$$

In Appendix B, it is shown that the estimation error ω in (33) has an approximately

zero mean Gaussian distribution in high SNR and its variance σ_ω^2 can be approximated by

$$\sigma^2_{\hat{T}_k} = \frac{N_0 K_k T_s^2}{|h_k|^2} \tag{34}$$

where

$$K_k = \frac{2\sum_i \sum_j \left\{ i \cdot j \cdot \left| q_k(i, \Delta f_k) \right| \cdot \left| q_k(j, \Delta f_k) \right| \cdot (s \star s)^2 (i - j) \right\}}{\left(\sum_i q_k^2(i, \Delta f_k) \right)^2} \tag{35}$$

and

$$q_k(n, f) = \sum_{l=0}^{L-1} s_{k,m}(l) \cdot s^*_{k,m}(n + l) \cdot e^{j2\pi f n T_s}. \tag{36}$$

Table 3: System parameters for the comparison of theoretical and simulated error variance.

Parameters	Values
Modulation	BFSK
Sampling rate	1 Ms/s
Symbol length	16 μs
Shaping pulse	Rectangular pulse
Preamble length (P)	1
Carrier frequency offsets	0, 5, 10, 15 (ppm) at 2.4 GHz

The theoretical error variance of the proposed COM estimator is verified by comparing with a Monte-Carlo simulation. For the variance calculation and the simulation, BFSK modulation is used with the rectangular shaping pulse and the number of samples per symbol $L = 16$. It is also assumed that the single-bit preamble $P = 1$ is used. It is noted that when the longer preamble is used, SNR would be increased by coding gain. The other system parameters are presented in Table 3. As a performance metric, root-mean-square error (RMSE) has been used, that is the square root of the variance for an unbiased estimator. The error is calculated by taking the difference between the estimated SOP-time and the true time that is zero. The comparison is shown in Figure 19. In the figure, the theoretical RMSE of the COM estimator is illustrated as a dotted line while the simulated RMSE is shown as a solid line. It is shown that the simulated RMSE curves are well matched to the theoretical RMSE curves for all of CFOs in high-SNR region. The mismatch at a low-SNR

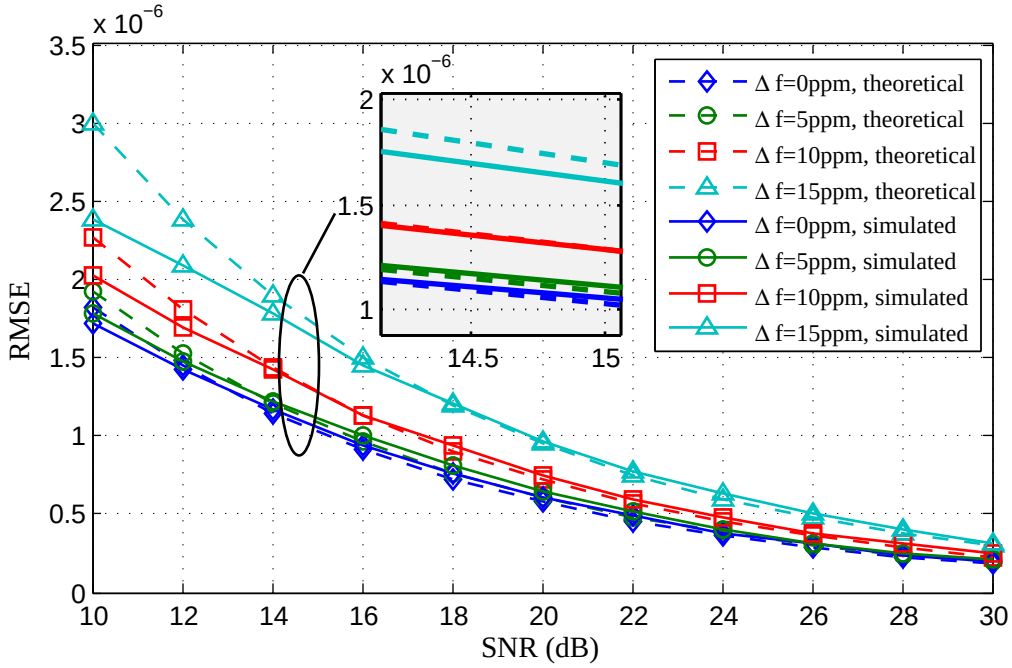


Figure 19: Comparison between the theoretical RMSE performance of the proposed mean estimator and simulated RMSE performance as a function of SNR.

region is mainly due to the high-SNR approximation in (32). The amount of mismatch tends to increase as the amount of CFO increases. It is noted that the RMSE in this figure is an order or microsecond. The overall RMSE can be further reduced by using a longer preamble as $P >> 1$.

Through a Monte-Carlo simulation, the performance of the proposed COM estimator is compared with a peak-finding algorithm that is typically used with a correlation-based estimator. The comparison is presented in Figure 20. In the figure, the RMSE values from the COM estimator are shown in solid lines while the values from the peak-finding estimator are shown in dotted lines. While peak-finding estimation outperforms COM estimation in the absence of or relatively small CFO, the mean estimation gives better performance in the presence of relatively large CFO $\Delta f > 10$ ppm that is typical in a commercial wireless device. Intriguingly, it is shown that the RMSE of the peak-finding estimator at 15 ppm CFO rises rapidly because the correlation output $\Lambda(t)$ becomes blunt and bimodal as CFO increases.

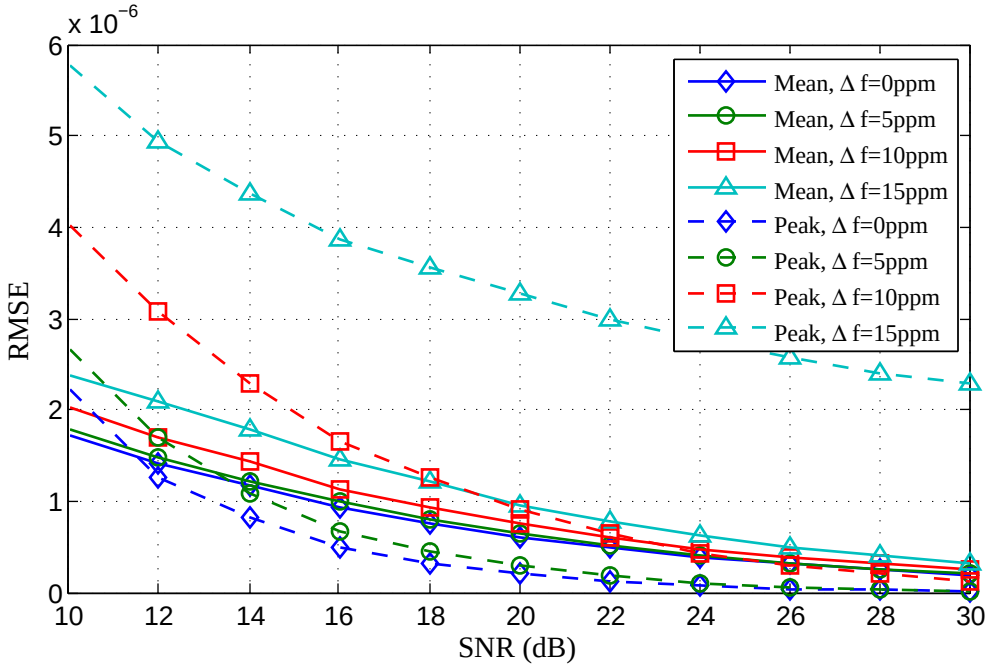


Figure 20: Comparison between the proposed mean estimator and the peak detector as a function of SNR.

5.2.2 Combined Start-of-Preamble-Time Estimation

In Section 4.4.1, the ABLUE estimator is proposed under the assumption of (20), and it is shown that the multi-hop CCT is statistically convergent when the ABLUE estimator is used to calculate the combined SOP-time. Since the error variance in (34) satisfies the design criteria, the weighted average of the SOP-time estimates $\hat{T}_k$ for $k = 1, \ldots, K$ to the channel gains $|h_k|^2$ minimizes the variance of the transmit time error, and makes the multi-hop CCT statistically stable. The estimation of the combined SOP-time $\hat{T}$ for the NCBFSK system can be written as

$$\hat{T} = \sum_{k=1}^{K} \frac{|h_k|^2}{\sum_{i=1}^{K} |h_i|^2} \cdot \hat{T}_k \tag{37}$$

Substituting $\hat{T}_k$ for (33), the above equation can be written as

$$\hat{T} = T_s \cdot \sum_{k=1}^{K} \left\{ \frac{|h_k|^2}{\sum_{i=1}^{K} |h_i|^2} \cdot \frac{\sum n \cdot \Lambda_k(n)}{\sum \Lambda_k(n)} \right\}. \tag{38}$$

To calculate the combined SOP-time in the above equation, it is required that there are

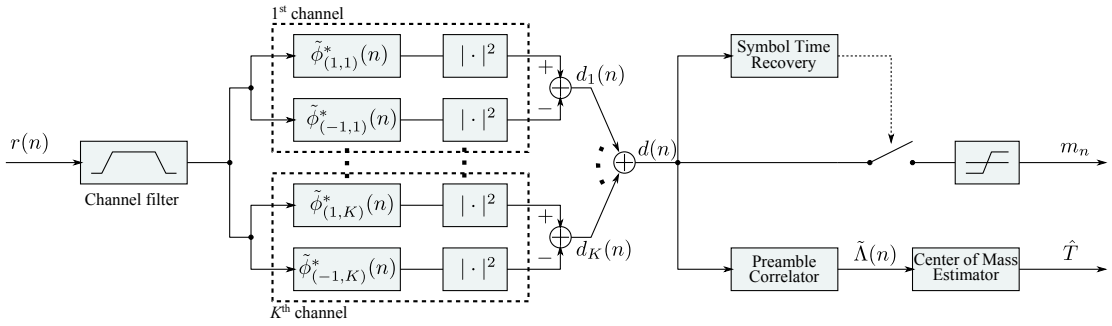


Figure 21: Non-coherent BFSK demodulator with orthogonal frequency diversity to estimate the combined SOP-time.

K preamble correlator, and the channel-gain estimation needs to be performed prior to the SOP-time estimation. Assuming high SNR where $|h_k|^2 >> |h_k|$, the denominator of (33) can be approximate with a constant Q as

$$\sum_{n=-PT_s}^{PT_s} \Lambda_k(n) \approx Q \cdot |h_k|^2. \tag{39}$$

Therefore (38) can be approximated as

$$\begin{aligned}
\hat{T} &= T_s \cdot \frac{\sum_{k=1}^{K} \left(\sum_n n \cdot \Lambda_k(n) \right)}{\sum_{k=1}^{K} \left(\sum_n \Lambda_k(n) \right)} \\
&= T_s \cdot \frac{\sum_n \left(n \cdot \sum_{k=1}^{K} \cdot \Lambda_k(n) \right)}{\sum_n \left(\sum_{k=1}^{K} \Lambda_k(n) \right)} \\
&= T_s \cdot \frac{\sum_n n \cdot \tilde{\Lambda}(n)}{\sum_n \tilde{\Lambda}(n)}
\end{aligned}$$

where

$$\tilde{\Lambda}_k(n) = \sum_{i=0}^{P-1} \left\{ \sum_{k=1}^{K} d_k(n + iL) \right\} \cdot p_i. \tag{40}$$

Therefore, the combined SOP-time can be approximated without having a channel-gain information. Furthermore, just a single preamble-correlator is needed where the sum of the outputs from K pairs of the matched filters is used for the preamble correlation.

Figure 21 shows how to implement the combined SOP-time estimation algorithm in a NCBFKS. The outputs from K pairs of the matched-filters are summed to produce the soft-valued output $d_k(n)$. The soft-valued output is used to estimate the combined SOP-time

through the preamble correlator, and to demodulate symbols.

5.3 Experimental Study of Cooperative Diversity

As discussed in Chapter 2, CCT provides spatial and cooperative diversity when cooperating nodes relay a source message through orthogonal channels. This experiment focuses on evaluating the spatial diversity of CT by measuring packet-delivery ratio (PDR), which is one minus the packet error rate (PER). Theoretically, diversity performance is indicated by the slope of the curve of BER vs. SNR [52]. However, in practical radio systems, it is easier to measure PDR or PER rather than BER. The CRC is used to determine if the received packet is corrupted. For the experiment, NCBFSK modulation and four diversity channels with equal-gain combining (EGC) are used. For the pre-synchronization of multiple relays, the SOP-time estimation method designed in Section 5.2 has been implemented on the SDR testbed. The other parameters used for the implementation are presented in Table 4.

Table 4: System parameters for the cooperative diversity experiment.

Parameters	Values
Modulation	BFSK
Sampling rate	1 Ms/s
Symbol length	16 μs
Bit rate	128 kb/s
Diversity channels	4
Carrier frequencies	2.482 GHz, 2.492 GHz
Packet length	100 bytes

The topology for the experiment is a two-hop network as shown in Figure 22. The source node transmits a packet to the nodes in Cluster A. The nodes in Cluster A relay the packet using CCT to the nodes in Cluster B. Then, the PDR is calculated at the nodes in Cluster B. The objective of this experiment is to measure the averaged PDR at Cluster B by changing the number of nodes in Cluster A. The PDR in each node is obtained by counting the number of correctly received packets, based on 1000 transmitted packets.

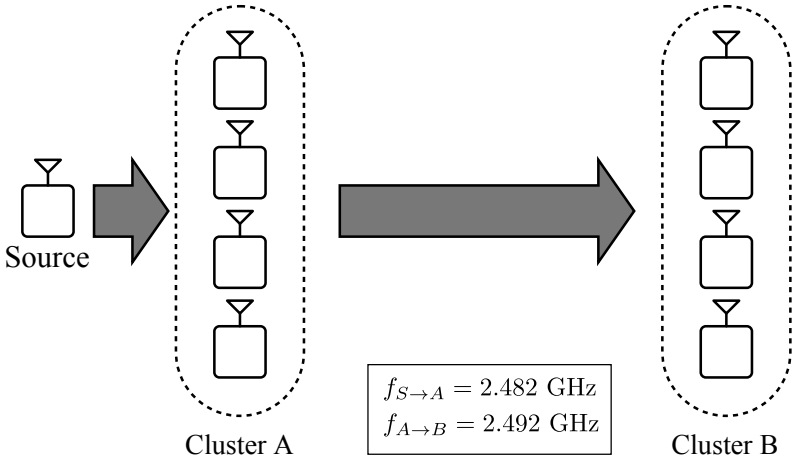
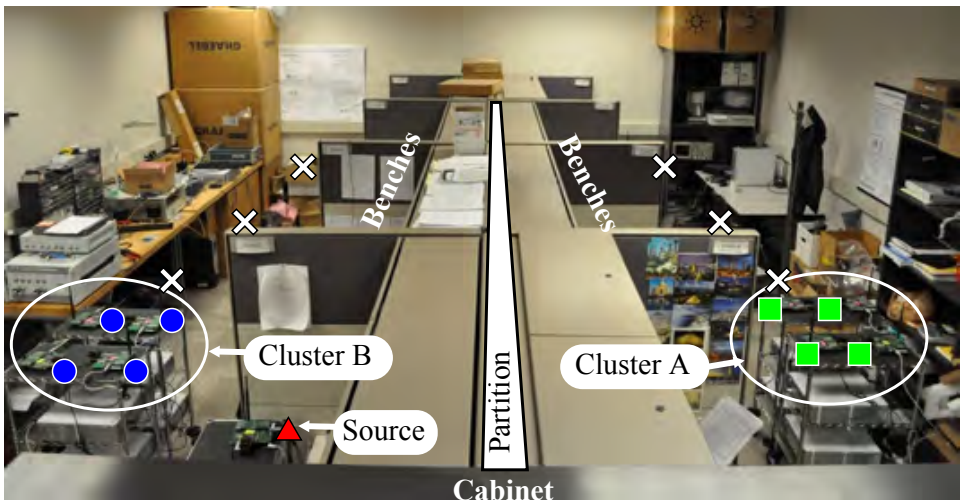


Figure 22: Network topology for the cooperative diversity experiment.



Bench	30"H	Engineered Wood
Cabinet	72"H	Metal
Partition	65"H	Foam-padded with metal frame

Source node
Node in Cluster A
Node in Cluster B
Location of Cluster

Figure 23: Node placement and layout of the Smart Antenna Research Laboratory.

The measurement is taken in the Smart Antenna Research Laboratory of Centergy Building, Georgia Institute of Technology. The layout of the lab and the wireless node placements are shown in Figure 23. In it noted that two different carrier frequencies f_A=2.482 Ghz and f_B=2.492 Ghz are used to isolate the source node and the nodes in Cluster B. In this experiment, different transmitter and receiver locations, keeping the distance constant, are used to realize channel variations.

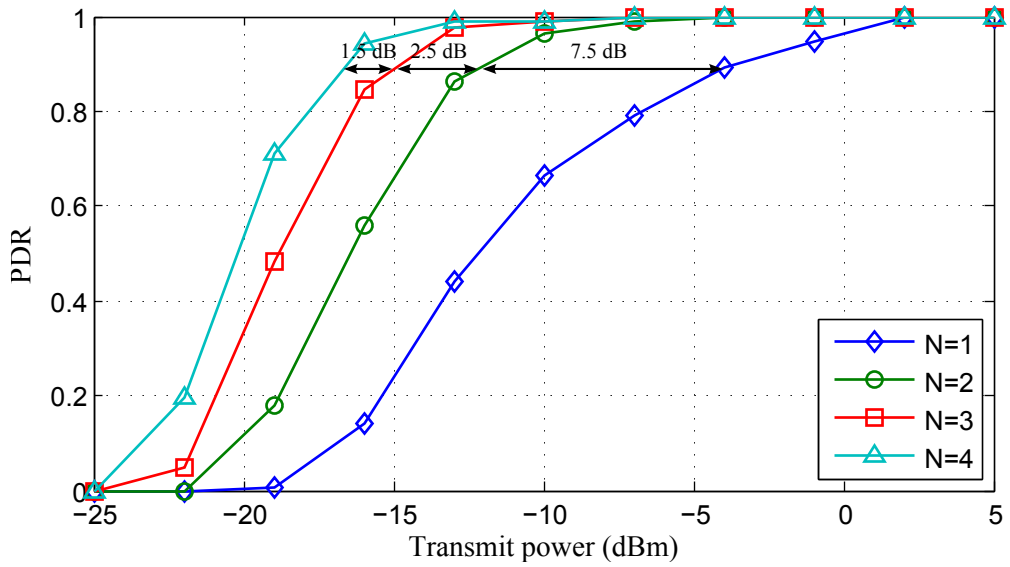


Figure 24: Measured PDR versus transmit power in each relay when $N = 1, 2, 3$ and 4.

Figure 24 shows the PDR versus transmit power at the nodes in Cluster A (i.e., relay nodes). The result shows that the averaged PDR is improved as the number of cooperators in Cluster A increases. The gain at PDR=0.9 is about 8 dB when $N = 2$ compared to $N = 1$. It is observed that the amount of gain decreases as the number of cooperators increases, which is a well-known characteristic of diversity gain [53]. It is also pointed out that even though each transmitter keeps the same transmit power regardless of the number of cooperators, the gain from the cooperative diversity is greater than the array gain that is 3 dB when $N = 2$. Therefore, it clearly shows that spatial diversity is exploited in a non-line-of-sight (NLOS) environment.

5.4 Experimental Study of Range Extension

In Section 5.3, the SNR advantage of CCT has been demonstrated through PER performance. The objective of this experiment is to compare the two-hop range of conventional SISO multi-hop with the two-hop range of 4-element CCT. The term "range" is defined to be the maximum distance between the relays and the destination, such that PER, when the effects of multipath fading are averaged out, is approximately ten percent (0.1). For the CCT cluster, a "maximally dispersed" topology is considered, which means that each

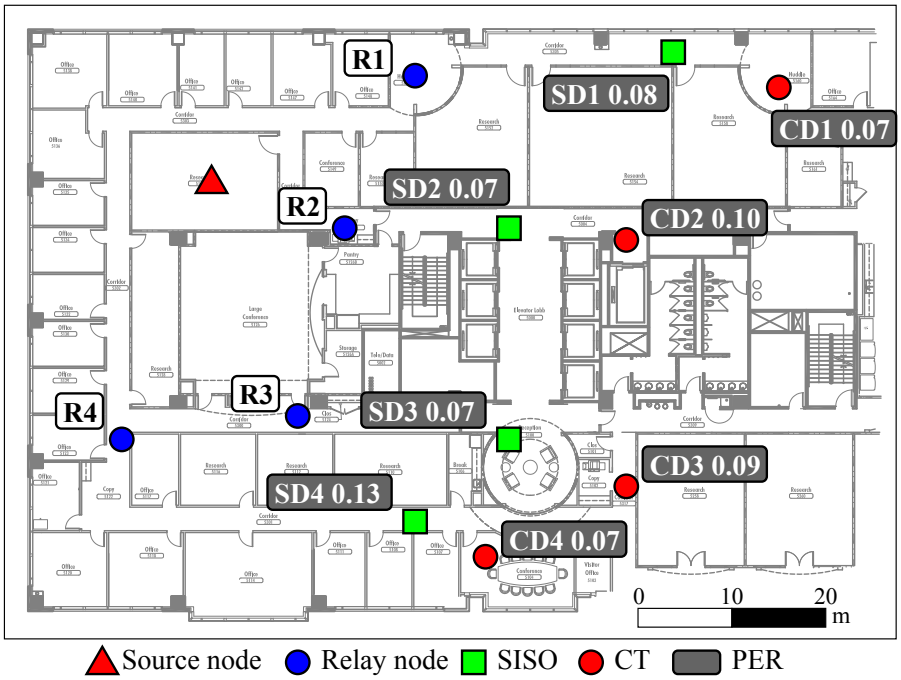


Figure 25: Node placement and floor map of the fifth floor of Centergy Building in Georgia Institute of Technology for the range-extension experiment.

cluster node is as far as possible from the source as well as being well separated from the other cluster nodes, while keeping an average PER of approximately 0.01. Figure 25 indicates, on the floor plan of the building, the source, relay and destination locations in which green squares indicate the reachable locations of a destination node when relays does not perform CCT while red circles indicate the ones for CCT. For each SISO destination, the previous-hop node is the nearest relay. For example, for **SD2**, the associated relay is **R2**. The CCT destinations are **CD1-CD4**, and they receive the CCT signal from all four relays.

For each destination and for each multipath realization, the PER at the destination is calculated based on 1000 transmitted packets. Next, the PER is averaged over 120 independent multipath channel realizations created by moving the terminals around in a local area, so that the shadowing effects are preserved and a 90 % confidence interval of about 0.06 is achieved. 15 independent channel realizations are obtained by having 15 different radios simultaneously receive the signal at the destination, as shown in Figure 26. For the CCT range measurement, moving the *relays* to eight different locations in a local area

Figure 26: Photographs of the destination cart for the range-extension experiment.

generates $8 \times 15 = 120$ different multipath channel realizations. For each SISO range measurement, we placed a small cart at the relay location; the cart held four GNU radios. A measurement was made for each radio on the cart, resulting in $4 \times 15 = 60$ channel realizations. Moving the small cart once created another 60 realizations for a total of 120.

We observe that the CCT destinations are further to the right than the SISO destinations, indicating range extension. However, the range extension appears to be only about 80 %, which is less than the predicted factors of 2 to 4. The shortfall can be explained in part by the non-homogeneity of the channels and the limitations of where we could put the measurement carts. For example, the top three SISO destinations are all within line-of-sight (LOS) or near LOS of one of the relays, while the CCT destinations are all non-LOS (NLOS). LOS gives a strong advantage over NLOS, especially in the center of the building where the cement walls surrounding the elevators and stairwells strongly attenuate the signal. Another factor contributing to the shortfall is the large distance between relays, which implied that, for any particular CCT destination, at most only two relays made significant contributions to the total power, so there was effectively only second order diversity from two relays. Therefore, we think that more range extension would be observed from (i) a denser distribution of relays, (ii) placing the SISO destinations also in NLOS locations, and (iii) performing the experiment in a part of the building away from the cement core.

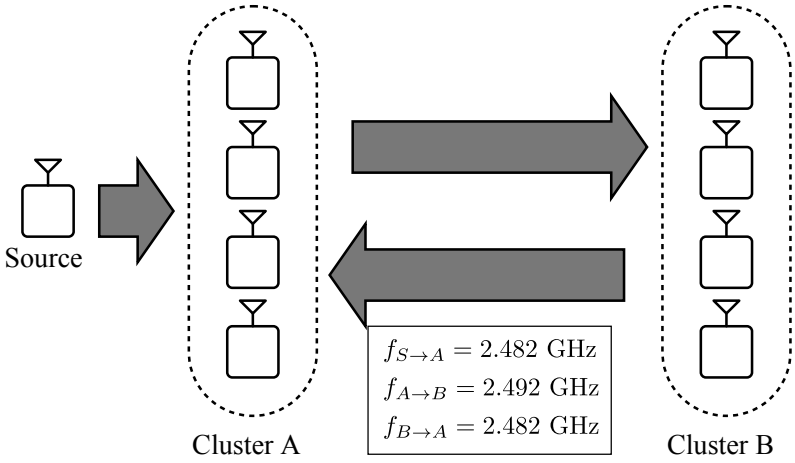


Figure 27: Network topology for the ping-pong experiment.

5.5 Experimental Study of Narrowband Multi-hop CCT

This experiment focuses on measuring the RMS transmit-time spread (RTTS) of cooperative nodes in a multi-hop network. In Section 4, it is theoretically proved that the RTTS over multiple CCT hops statistically converges then the proposed combining-rule is used to estimate the SOP time for CCT. To emulate multiple hops with the small number of nodes, the "ping-pong" experiment is designed, where two groups of cooperating nodes transmit the source message back and forth for up to 10 hops (or "CT"s). Figure 27 shows the network topology for the ping-pong experiment. The source node initiates a packet transmission through frequency f_A. This single packet is transmitted back and forth between Cluster A and Cluster B. The RTTS of each successive hop is measured and recorded. In order to measure the transmit time in each node in a cluster, the FPGA in USRP1 is modified to generate a square wave at the general purposed IO (GPIO) pin on the board. The square wave is generated in the way that its rising edge is coincided with the time when the packet is released for retransmission by the FPGA in the USRP1. This signal that is generated in each node in a cluster is monitored by the customized FPGA board to calculate the time differences between the transmissions. The time precision of the customized FPGA board is 15.6 ns.

Two different cluster topologies are considered, as shown in Figure 28; the left topology

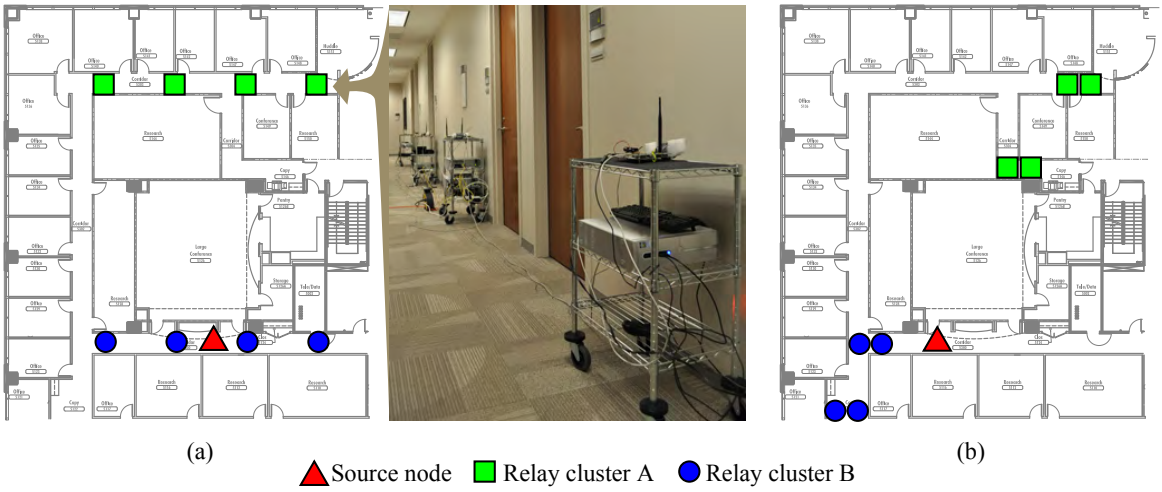


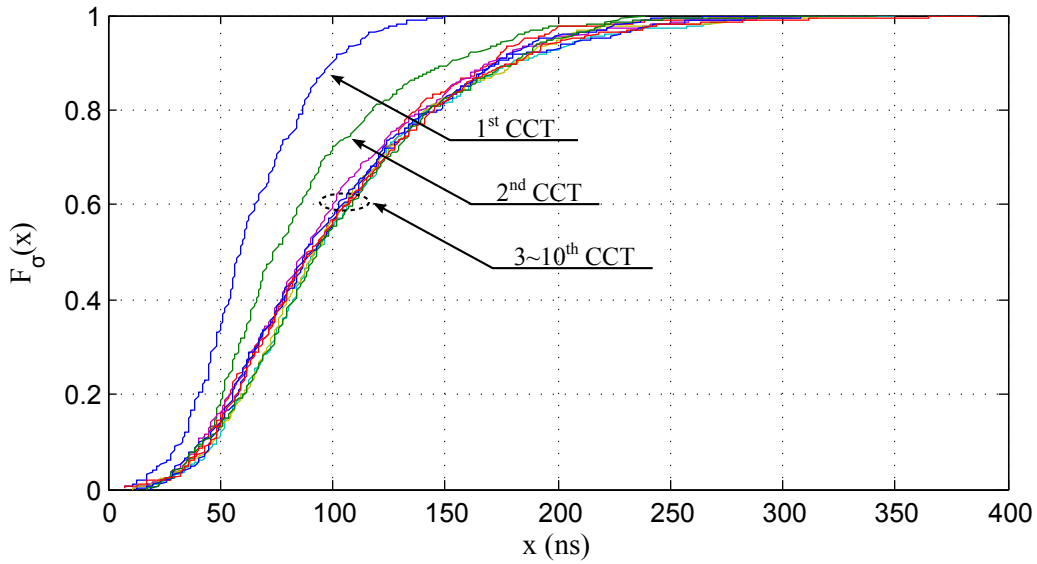
Figure 28: Floor map of the two topologies for the ping-pong experiment.

features parallel linear clusters, which might approximate OLA broadcasting [54] in a strip network. The photograph of one of the linear clusters is also shown in the figure. The right topology in Figure 28 is intended to be a worst case topology in terms of propagation delay differences [55]. The RTTS of cooperative relay nodes in l^{th} CT can be calculated by

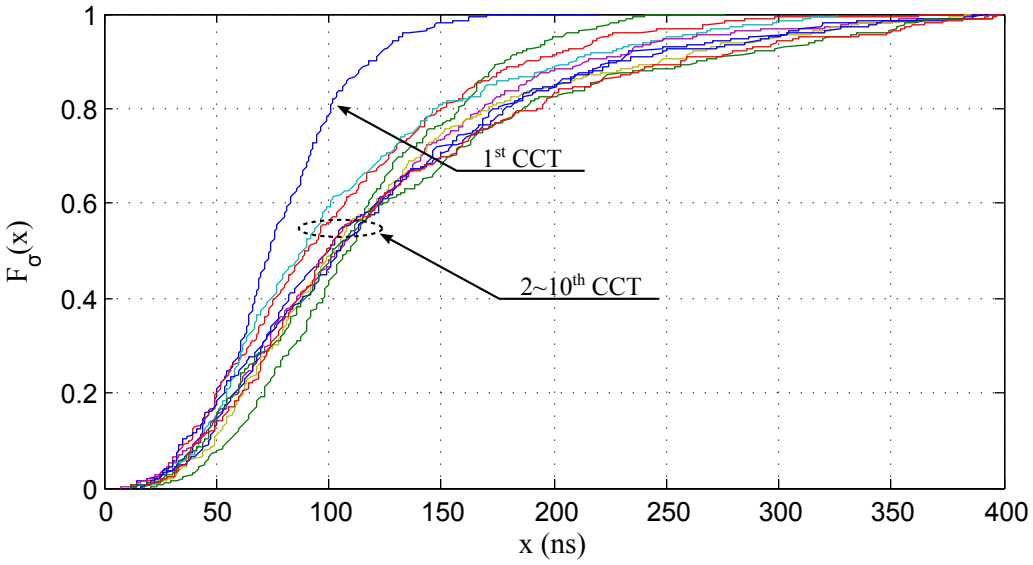
$$\sigma_\tau^l = \sqrt{\frac{\sum_i^N (t_{T,i}^l - \hat{t}_T^l)^2}{N-1}}, \tag{41}$$

where $t_{T,i}^l$ is a measured transmit time of i^{th} relay node in l^{th} CT and $\hat{t}_T^l$ is the sample mean of transmit time of all relay nodes. As shown in Figure 27 two groups of cooperative nodes transmit the source message back and forth for up to 10 hops (or "CT"s). The experiment was repeated 500 times to get 500 trials of σ_τ^l.

Figure 29 shows the empirical RTTS of cooperative nodes in the two topologies. Each curve represents an empirical cumulative density function (CDF) of RTTS of each CT. It is observed that the RTTS of the first CCT provides better performance than other CCTs in both topologies. This is because the timing reference of cooperative nodes in 1^{st} CT is a single source transmission in which there is no timing spread caused by multiple timing offsets from previous cluster. From the measurement, it is also observed that the CDFs of RTTS of CTs 3 through 10 essentially overlay each other, which implies the practicality of concurrent multi-hop CT. Moreover 90% of RTTS outcomes are less than 300 ns in both



(a) RTTS in the first topology



(b) RTTS in the second topology

Figure 29: Measured RTTS in the ping-pong experiment.

topologies, which indicates that CCT can support up to 300 kbits/s data rate in narrowband waveforms without ISI degradation [21]. Broadband waveforms, such as OFDM with a 0.8 μsec guard interval, could also be supported. These results experimentally support the fact that the RTTS over multiple CCT hops statistically converges as hop increases.

5.6 Summary

In this chapter, we developed the non-coherent SOP-time estimation algorithm and the COM estimator. The proposed COM estimator is designed to have low implementation complexity and satisfy the ABLUE criteria proposed in Chapter 4. In addition, the COM estimator is robust to CFO that causes a distortion of the preamble correlation output. The theoretical error variance of the proposed estimator is derived and compared with the Monte-Carlo simulation result. Using the proposed SOP-time estimator, the SNR advantage of CCT has been demonstrated in terms of PER as well as transmission range. Furthermore, the ping-pong experiment demonstrated that the statistics of transmit-time error are convergent as the number of hop increases, which implies that the CCT is practical to a multi-hop network.

CHAPTER 6

TIME AND FREQUENCY ESTIMATION FOR OFDM-BASED CONCURRENT COOPERATIVE TRANSMISSION

6.1 Overview

In this chapter, we design an orthogonal frequency division multiplexing (OFDM)-based CCT system for multi-hop DMIMO systems. OFDM is a method of encoding digital data on multiple subcarrier frequencies. By using OFDM, complicated channel equalization can be avoided since each subcarrier can be treated as a narrowband transmission with small enough subcarrier spacing. In addition, an OFDM system is robust to timing error (e.g., multipath delays or OFDM symbol timing error) by having a guard interval (GI) to avoid inter-symbol interference (ISI). On the other hand, OFDM requires very accurate frequency synchronization between the transmitter and the receiver. The offset of the carrier frequency makes the subcarriers no longer orthogonal, causing inter-carrier interference (ICI). In a MIMO-OFDM link, any CFOs between antennas also can induce the ICI at a receiver. In a conventional MIMO-OFDM link (also referred to as centralized MIMO-OFDM) where the single transmitter has multiple antennas and the single receiver has multiple antennas, there are no offsets between transmit antennas, because the antennas are connected by wire. The effect of the CFO between distributed relays in a space-time block code (STBC)-OFDM system has been studied theoretically [56, 57] and numerically [58]. The analysis in [58] shows that the symbol-error-rate (SER) performance has the 10^{-5} error floor when there exists 1% of CFO error between two quadrature phase-shift keying (QPSK) Alamouti Coded-OFDM antennas. Doppler shift can induce such an offset. In this dissertation, however, we assume a static network, and therefore the Doppler shift is zero. In an OFDM-based CCT link where transmitters are spatially separated, however, the frequency offsets between distributed transmitters must be effectively suppressed to be within the tolerance of the OFDM system. In addition, the timing spread caused by the pre-synchronization

error of the distributed transmitters will increase the length of GI, which also decreases the achievable data-rate.

Most of the previous work in synchronization for DMIMO-OFDM explicitly estimates each of the multiple offsets and then exploits those estimates to equalize or otherwise compensate the offsets in the destination receiver [31,32,59]. The estimation can be done using either a time-division multiplexing (TDM)-based training sequences [59] or an approach designed especially for CT by the cooperative transmitters [31, 32]. In the TDM-based approach, each relay transmits in a different time slot, causing excessive overhead when there are more than two cooperating transmitters, e.g., in an opportunistic large arrays [38] or in a wide cooperative route [33, 35].

Several authors have addressed explicit offsets estimation for simultaneous transmission [31, 32]. In [31], orthogonal training preambles are proposed to estimate multiple timing and frequency offsets for distributed STBC-OFDM systems. The method estimates the multiple offsets at the destination and feeds back the estimates to the relays so that the data can arrive in a synchronous manner. However, the feedback operation requires extensive resources in some applications. Furthermore, the feedback might not be possible if the cooperation creates range extension where the destination cannot reach back to the relay by itself. Also, although our proposed preambles are similar to [31] in that they are both orthogonal and OFDM-based, the preambles in [31] would not enable the low-complexity, single-estimate approach that we propose. Linearly independent training signals simultaneously transmitted from the relays are proposed in [32]. Even though they utilize the multiple estimated offsets for decoding in a two-hop OFDM-based cooperative system, they do not discuss about the pre-synchronization for a multi-hop DMIMO network. We note that explicit offset estimation usually also requires knowledge of the number of offsets, which is problematic in opportunistic large array-based approaches, such as [33, 35, 38], because the number of relays in any particular cluster is not known a priori.

All of the approaches that explicitly estimate each offset tend to be computationally

intense. In contrast, in our approach, the receiver directly computes just one offset estimate for each of time and frequency, with no explicit knowledge of channel gains, and with relatively low computational complexity. The proposed algorithm uses just a single OFDM preamble to estimate the time and frequency with a reasonable constraint that the CFO is bounded. In addition, because of the construction, the preamble OFDM symbol can be easily distinguished from data OFDM symbols, which can significantly reduce a false alarm probability or can avoid using additional resource to detect the existence of a packet. Furthermore, our estimate is of the form that is already known to produce convergent estimation error statistics as a function of number of hops in Chapter 4, and which produce offset spreads that are consistent with typical receiver tolerances in Chapter 5. The main contributions in this chapter are 1) a novel OFDM preamble design that makes possible this particular type of estimate, and 2) the low-complexity signal processing method in the receiver that computes the estimate, and 3) experimental demonstration.

This chapter is organized as follows. Section 6.2 describes the proposed preamble structure and system model. In Section 6.3, we develop the time and frequency estimation method for an OFDM-based CCT. The simulation studie of the proposed estimation method will be provided in Section 6.4. In Section 6.6and 6.7, experimental studies for two-hop and multi-hop CCT are provided. This chapter is summarized in Section 6.8.

6.2 Preamble Structure and System Model

The proposed preamble structure in time and frequency domain is illustrated in Figure 30. In frequency domain, all odd subcarriers are set to zero. This makes the inverse Fourier transform of the sequence is also repeated in time domain as same as [60]. The OFDM subcarriers are divided Z sectors $\{P_0, P_1, \cdots, P_{Z-1}\}$ excluding Q length of a guard band on both sides. The guard band is used to give safety margin to avoid aliasing in the presence of CFO. Each sector consists of $2K$ subcarriers where one even slot is assigned to each of

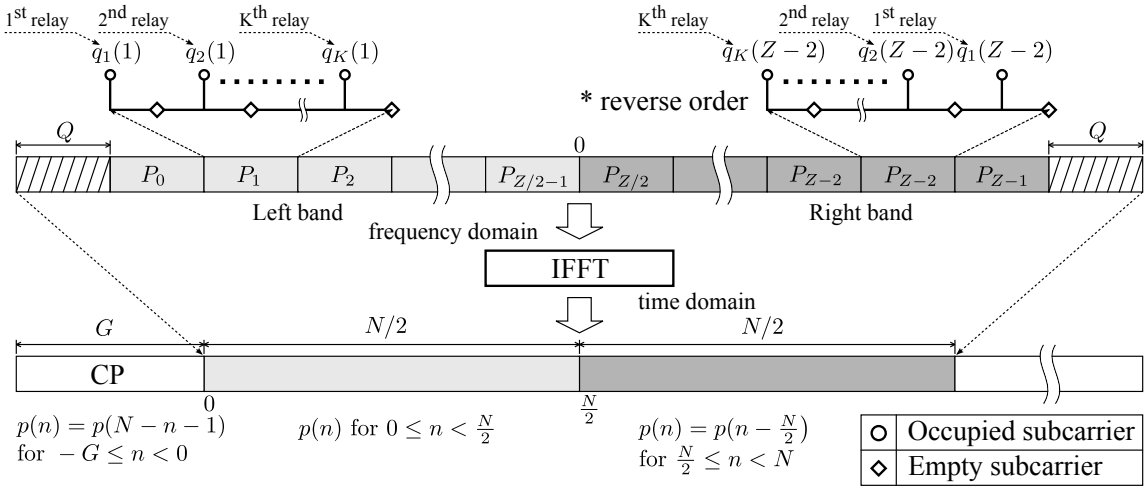


Figure 30: Illustration of the proposed preamble structure in time and frequency domain.

the K relays. Therefore, the number of sectors becomes

$$L = \frac{N - 2Q}{2K}.$$

As shown in the figure, the subcarriers for each relay are assigned in left-to-right in the left band, while they are assigned right-to-left in the right band. The symmetric preamble of the k^{th} relay can be written as

$$q_k(l) = q_k(L - l - 1), \quad \text{for } l = 0, 1, \ldots, \frac{Z}{2} - 1.$$

Z preamble symbols $[q_k(0), q_k(1), \ldots, q_k(Z-1)]$ for the k^{th} relay are inserted into corresponding location in each sector as

$$P_k(n) = \begin{cases} q_k(n - Q), & Q \le n < N - Q \\ 0, & \text{otherwise} \end{cases}$$

where $P_k(n)$ is the frequency domain preamble symbol of the k^{th} relay. The constructed preamble in frequency domain is converted into the time-domain signal through the inverse discrete Fourier transform (IDFT) and a cyclic prefix is attached to the time-domain preamble. For the sake of convenience, we assume that the time-domain preamble starts at discrete index zero and the attached cyclic prefix has negative time index as shown in

the figure. It is pointed out that the preambles transmitted from K relays are orthogonal in frequency since they choose different subcarriers.

Let $p_k[n]$ denote the time-domain preamble of the k^{th} relay. The transmission signal at the k^{th} relay with cyclic prefix G can be written as

$$s_k(n) = \begin{cases} p_k(n), & 0 \le n < N \\ p_k(n+N), & -G \le n < 0 \end{cases}.$$

As discussed above, the time domain preamble has repeated sequences in time domain as

$$p_k(n) = p_k(N/2+n), \quad \text{for } n = 0, 1, \ldots, \frac{N}{2} - 1.$$

In addition, since all transmissions are orthogonal for the preambles, we have

$$\sum_{n=0}^{N} p_i(n) \cdot p_j(n) = 0, \quad \text{for all } i, j \text{and } i \neq j$$

Assuming that the k^{th} relay transmits its preamble with normalized time offset ϵ_k to sampling period and normalized frequency offset ω_k to subcarrier spacing, the received signal from the k^{th} relay at the receiver can be expressed as

$$r(n) = \sum_{k=1}^{K} e^{j2\pi\omega_k \frac{n}{N}} \left\{ \sum_{l=0}^{L-1} h_k(n) s_k(n-l-\epsilon_k) \right\} + v(n)$$

where $\mathbf{h}_k = [h_k(0), \ldots, h_k(L-1)]^T$ and $v(n)$ are channel impulse response and an AWGN with variance σ_v^2 respectively.

Letting $\mathbf{r} = [r(0), \ldots, r(N-1)]^T$, $\mathbf{\Gamma}_k = diag\{1, e^{j2\pi\omega_k/N}\}, \ldots, e^{j2\pi\omega_k(N-1)/N}\}$, $[\mathbf{S}_k]_{mn} = s_k(m-n-\epsilon_k)$, where $m = 0, \ldots, N-1$ and $n = 0, \ldots, L-1$, the received signal for the k^{th} relay can be expressed as a matrix form as

$$\mathbf{r}_k = \mathbf{\Gamma}_k \cdot \mathbf{S}_k \cdot \mathbf{h}_k.$$

The superimposed signal from K relays can be expressed as a vector form as

$$\mathbf{r} = \mathbf{\Lambda} \cdot \mathbf{h} + \mathbf{v}$$

where $\mathbf{\Lambda} = [\mathbf{\Gamma}_1\mathbf{S}_1, \ldots, \mathbf{\Gamma}_K\mathbf{S}_K]$ and $\mathbf{h} = [\mathbf{h}_1^T, \ldots, \mathbf{h}_K^T]^T$ and $\mathbf{v} = [v(0), \ldots, v(N-1)]^T$.

Since the vector $\mathbf{v}$ in the above equation is a vector of AWGN variables with the same mean and variance, the joint ML estimator of the timing and CFO parameters can be expressed as

$$(\hat{\epsilon}, \hat{\omega}) = \arg\max_{\epsilon,\omega}\left\{\mathbf{r}^H\mathbf{\Lambda}(\mathbf{\Lambda}^H\mathbf{\Lambda})^{-1}\mathbf{\Lambda}^H\mathbf{r}\right\}. \tag{42}$$

Since the maximization in (42) requires a search over the $2K$ dimensional space spanned by (ϵ, ω), the ML estimation is not practical.

6.3 Time and Frequency Offset Estimation

In this section, the time and frequency estimation is developed using the proposed preamble structure. The time and frequency offset estimation is achieved in the following sequences.

- **Coarse timing offset estimation**

  The presence of the preamble is detected by delayed correlation in time domain. The coarse timing estimate from the delayed correlation is used to define a window for the following estimation processes.

- **Combined fractional frequency offset estimation**

  In this stage, the fractional part of the combined CFO is estimated and compensated. Assuming that relative frequency offsets between transmitters are small enough, the fractional part of the weighted average of the CFOs with respect to the subcarrier spacing is estimated instead estimating individual CFOs.

- **Integer part frequency offset estimation**

  Once the window is defined and the fractional part of the combined CFO is compensated, the integer part of the combined CFO is estimated using ML estimator in this stage.

- **Fine timing offset estimation**

  The fine timing offset of the SOP location is estimated in this stage by using frequency domain symmetric correlation. The timing metric is designed to cope with the phase rotation caused by the timing offset.

6.3.1 Coarse Timing Estimation

The coarse timing of the preamble location can be obtained by delayed auto-correlation of the two halves of the preamble in time domain [60]. From the construction of the preambles, the first $N/2$ half of the preamble is identical to the second $N/2$ half. The correlation function and the normalized function for the coarse timing estimation are calculated as

$$A_c(m) = \sum_{n=0}^{N/2-1} r^*(m+n)r(m+n+\frac{N}{2}) \tag{43}$$

$$B_c(m) = \frac{1}{2}\sum_{n=0}^{N-1} \left|r(m+n)\right|^2 \tag{44}$$

and the timing metric becomes

$$M_c(m) = \frac{|A_c(m)|}{B_c(m)}.$$

Assuming that all timing errors are zero $\epsilon_k = 0$ for all k and frequency-flat fading channel which yields $L = 1$, the correlation function can be written as

$$\begin{aligned} A_c(m) &= \sum_{n=0}^{N/2-1} \left\{ \sum_{k=1}^{K} e^{j2\pi\omega_k(n+m)/N} h_k s_k(n+m) \right\}^* \\ &\quad \cdot \left\{ \sum_{k=1}^{K} e^{j2\pi\omega_k(n+m+\frac{N}{2})/N} h_k s_k(n+m+\frac{N}{2}) \right\} \\ &= \sum_{n=0}^{N/2-1} \left\{ \sum_{k=1}^{K} \left|h_k\right|^2 e^{j\pi\omega_k} \left|s_k(n+m)\right|^2 \right. \\ &\quad \left. + \sum_{\substack{x=1 \\ x\neq y}}^{K} \sum_{y=1}^{K} h_x^* h_y e^{j\theta_{xy}} s_x^*(n+m) s_y(n+m) \right\}. \end{aligned} \tag{45}$$

Within a guard interval of the preamble where $-G \leq m < 0$, the second term of (45)

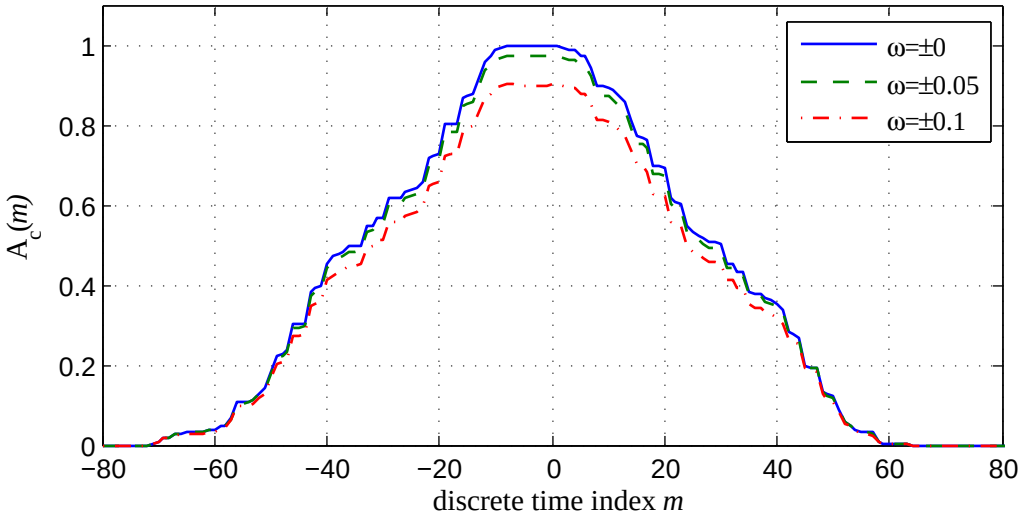


Figure 31: Example of coarse timing metric for AWGN channel and $K = 2$ with different amount of normalized CFOs.

becomes zero because of the orthogonal property. Therefore, the correlation function becomes

$$A_c(m) = \sum_{n=0}^{N/2-1} \left\{ \sum_{k=1}^{K} \left|h_k\right|^2 e^{j\pi\omega_k} \left|s_k(n+m)\right|^2 \right\}. \tag{46}$$

$B_c(m)$ can be similarly simplified. Since $s_k(n) = s_k(n + N/2)$, the coarse timing metric within the GI becomes $M_c(m) = 1$. By Cauchy Schwarz inequality, it is readily shown that $A_c(m) \leq B_c(m)$ for all m. Therefore, the coarse timing metric around the preamble location becomes

$$M_c(m) \begin{cases} = 1, & 0 \leq n < N/2 \\ < 1, & \text{otherwise.} \end{cases}$$

Figure 31 shows an example of the coarse timing metric for the AWGN channel and $K = 2$ with 128 subcarriers and the guard interval 8. With different amount of CFOs between two transmitters, the maximum value of the timing metric decreases. In addition, the timing metric reaches a plateau within the GI. It is noted that for the decoding purpose the start of the preamble can be taken any point within this plateau. This is because the timing offset within the GI only causes the phase rotation across subcarriers. The phase rotation can be compensated by the channel equalization process since the channel training sequence

also experiences the same phase rotation. On the other hand, from the pre-synchronization point of view, the timing error will be accumulated to the channel delay spread when they retransmit the message to the next cluster. This leads to have a large guard interval for the nodes in the next cluster. Therefore, more accurate timing estimation is further required for the pre-synchronization.

Once the timing metric $M_c(m)$ exceeds a pre-defined threshold, the coarse timing can be found by searching

$$\tilde{m}_c = \arg\max_{m \in \mathcal{M}} \{M_c(m)\}$$

where $\mathcal{M}$ is a set of adjacent sample points that exceed the threshold.

6.3.2 Fractional Combined-CFO Estimation

In Chapter 4, we show that the weighted average of multiple SOP times and CFOs, weighted by the channel gain, minimizes the error variance of the retransmission time and frequency. In this section, we show how this type of estimate can be obtained for the fractional CFO, under the assumption that the CFOs from the K relays differ by less than half of the sub-carrier spacing as $|w_x - w_y| < 0.5, \quad$ for all x and y.

When $K = 1$, (46) can be written as

$$A_c(m) = \sum_{n=0}^{N/2-1} |h|^2 \cdot e^{j\pi\omega} \cdot \left|s(m+n)\right|^2.$$

Referring to [60], at the coarse timing point $\tilde{m}_c$ the CFO can be estimated as

$$\hat{\omega} = \frac{\angle A_c(\tilde{m}_c)}{\pi}.$$

The phase angle operator $\angle(\cdot)$ wraps the angle to the interval $[-\pi, \pi]$ corresponding to the normalized frequency interval $[-1, 1]$. Therefore, $\hat{\omega}$ is considered as a fractional part of the CFO. The residual CFO η, referred to as an integer part of CFO, will be treated in the next subsection.

For $K > 1$, the combined CFO can be written as

$$\omega_c = \frac{\sum_{k=1}^{K} |h_k|^2 \omega_k}{\sum_{k=1}^{K} |h_k|^2}. \tag{47}$$

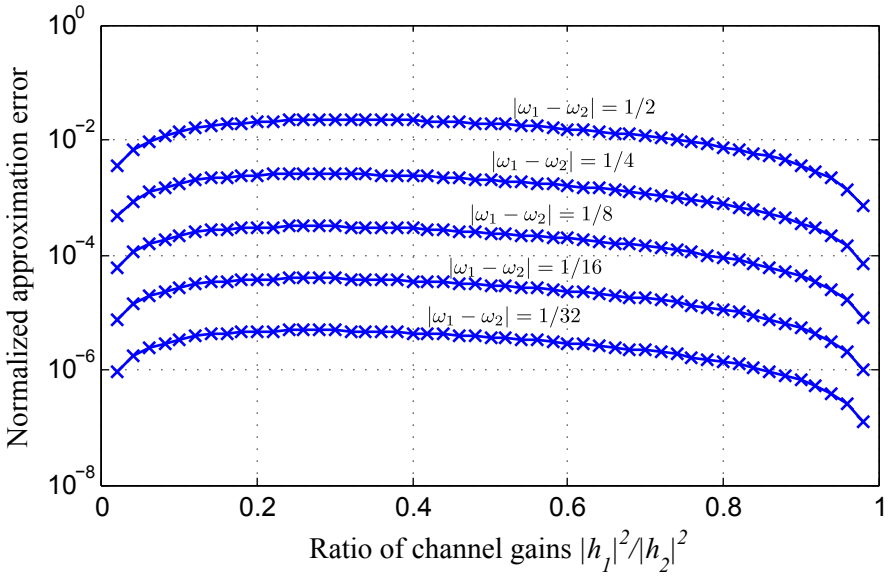


Figure 32: Approximation error when two angles are combined.

In Appendix C, assuming that the relative CFO errors are small enough, we show (47) can be approximated by simply taking the phase angle of $A_c(\tilde{m}_c)$ as

$$\hat{\omega}_c \approx \frac{\angle A_c(\tilde{m}_c)}{\pi}. \tag{48}$$

It is pointed out that this operation is the same as $K = 1$ so that the fractional CFO estimation method can be used regardless of the number of relays.

Figure 32 shows the approximation error of the combined fractional CFO estimation method when $K = 2$. For this plot, the actual combined CFO is calculated from (48) as

$$\omega_c = \frac{|h_1|^2}{|h_1|^2 + |h_2|^2} \cdot \omega_1 + \frac{|h_2|^2}{|h_1|^2 + |h_2|^2} \cdot \omega_2 \tag{49}$$

while the approximated combined CFO is calculated as

$$\hat{\omega}_c = \angle\left\{|h_1|^2 \cdot e^{j\pi\omega_1} + |h_2|^2 \cdot e^{j\pi\omega_2}\right\}. \tag{50}$$

The x-axis is the ratio of two channel gains and the y-axis represents the normalized approximation error. From the plot, it can be observed that the reduction of the relative CFO error by half decreases the approximation error by a factor of 10. For example, when the normalized relative CFO error is 0.125, the approximation error of the normalized combined CFO is less than 10^{-3}. It is pointed out that the approximation error is zero when the

ratio of the channel gain is zero or one. Trivially, the approximation error becomes zero regardless of the ratio of the channel gain when the relative error is zero.

6.3.3 Integer CFO Estimation

Once the coarse timing $\tilde{m}_c$ of the SOP is estimated, it can be used to define the searching window for the following estimation processes. From the property of the coarse timing estimation, $\tilde{m}_c$ is within the guard interval. To reduce the searching dimension and computational complexity, we set the searching window around the coarse timing location as shown in Figure 33. The backward and forward window sizes are user-defined parameters, which are denoted as W_B and W_F respectively in the figure. N samples $r(m)$ for $m = m_s, \ldots, m_s + N - 1$ where $\tilde{m}_c - W_B \leq m_s < \tilde{m}_c + W_F$ are converted into frequency domain samples through the sliding DFT window. Let $R_m(n)$ denote the DFT output of the samples $\{r(m), \ldots, r(m + N - 1)$ for $n = 0, \ldots, N - 1$, and let the actual preamble start at $m = 0$. In this section, we assume that the combined fractional CFO is already compensated. Since the residual CFO error η is integer, it makes the output from the DFT window shifted by η. The DFT output of the received preamble from the the k^{th} relay that is windowed starting at m can be written as

$$R_{k,m}(n) = \psi_k(n) \cdot e^{j2\pi \frac{m+n}{N}} \cdot P_k(n + \eta) \tag{51}$$

where $\psi_k(n)$ and $z(n)$ are the channel frequency response of the n^{th} subcarrier from the k^{th} relay node and AWGN with variance σ_z^2 respectively.

In this section, we propose two estimators: 1) a maximum-likelihood estimator and 2) a phase-difference correlation-based estimator. The former estimator is designed under the assumptions that there is no channel status information at a receiver. For the latter estimator, it is assumed that the channel phases are constant during two consecutive preamble sequences in the frequency domain.

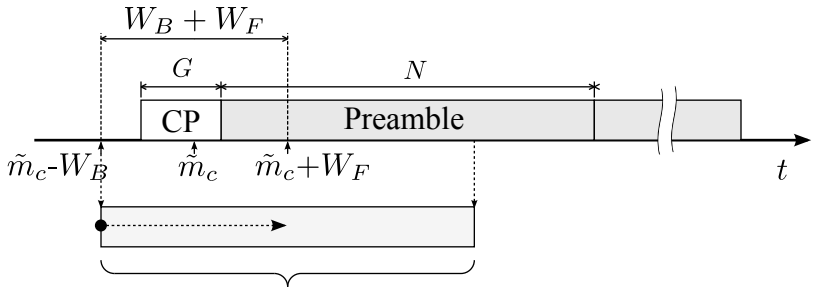


Figure 33: Illustration of defining the searching window.

6.3.3.1 Maximum-Likelihood Estimator

The vector form of (51) can be written as

$$\mathbf{R}_{k,m} = \mathbf{\Psi}_k \cdot \mathbf{\Phi}_m \cdot \mathbf{P}_{k,\eta}$$

where

$$\mathbf{\Psi}_k = \text{diag}\{\psi_k(0), \ldots, \psi_k(N-1)\}$$

$$\mathbf{\Phi}_m = \text{diag}\{e^{j2\pi\frac{m}{N}}, \ldots, e^{j2\pi\frac{m+N-1}{N}}\}$$

$$\mathbf{P}_{k,\eta} = [P_k(\eta), \ldots, P_k(N-1+\eta))]^T.$$

Then the superimposed preamble in frequency domain transmitted from K relays can be written in vector form as

$$\mathbf{R}_m = \mathbf{\Phi}_m \cdot \mathbf{P}_\eta \cdot \mathbf{\Psi} + \mathbf{z}$$

where $\mathbf{P}_\eta = [\mathbf{P}_{1,\eta}, \ldots, \mathbf{P}_{K,\eta}]$, $\mathbf{\Psi} = [\mathbf{\Psi}_1, \ldots, \mathbf{\Psi}_K]^T$, and $\mathbf{z} = [z(0), \ldots, z(N-1)]$.

The vector $\mathbf{z}$ in the above equation is a vector of AWGN variables with the same mean and variance. Let us assume that the integer part of the CFO is within $[-Q, Q]$; this makes the DFT output of the received preamble not aliased regardless of the CFO amount. Without having exact knowledge of the channel impulse response $\mathbf{\Psi}$, the ML estimator of the integer CFO η at given starting point m can be readily expressed as

$$\hat{\eta}_m = \underset{\eta\in[-Q,Q]}{\arg\max}\left\{\left\|\mathbf{\Theta}(m,\eta)\right\|^2\right\}$$

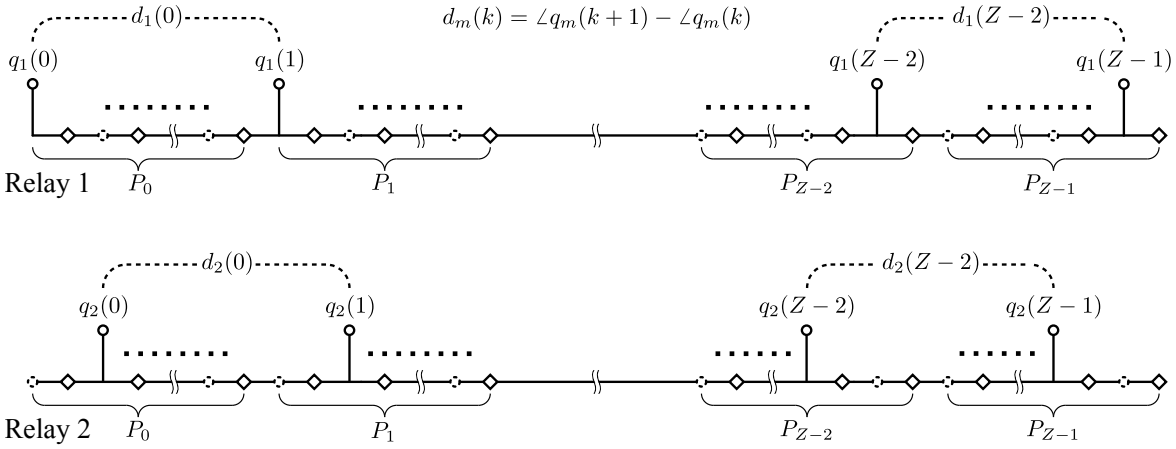


Figure 34: Illustration of the phase difference of the preamble sequences of two relays.

where

$$\mathbf{\Theta}(m, \eta) = \mathbf{R}_m^H \cdot \mathbf{P}_\eta.$$

Within the searching window, the integer part of CFO can be found by searching all possible m to maximize $\mathbf{\Theta}(m, \eta)$ as

$$\hat{\eta} = \max\left(\left\{\mathbf{\Theta}(m, \hat{\eta}_m) : m = [\tilde{m}_c - W_B, \tilde{m}_c + W_F]\right\}\right).$$

6.3.3.2 Phase-difference-correlation-based Estimator

If the channel phases are constant over multiple preamble sequences, the phase difference between the consecutive preamble sequences is still hold. Therefore, the integer CFO can be estimated more accurately under this assumption. Figure 34 shows the illustration of the phase difference of the designed preamble sequence in two relays.

Let $d_m(k)$ denote the designed phase difference between $q_m(k+1)$ and $q_m(k)$. Let also $\tilde{d}_m(k)$ denote the phase difference of the received signal at the same subcarriers that might be shifted by η in the frequency domain due to CFO. For the m^{th} relay, the CFO can be estimate by minimizing

$$\hat{\eta}_m = \underset{\eta \in [-Q, Q]}{\arg\max} \left\{ \sum_{k=1}^{N} \left| \tilde{d}_m(k+\eta) - d_m(k) \right|^2 \right\}.$$

Like the maximum-likelihood estimator, the integer part of CFO can be found within the searching window by searching all possible m to maximize the phase-difference correlation

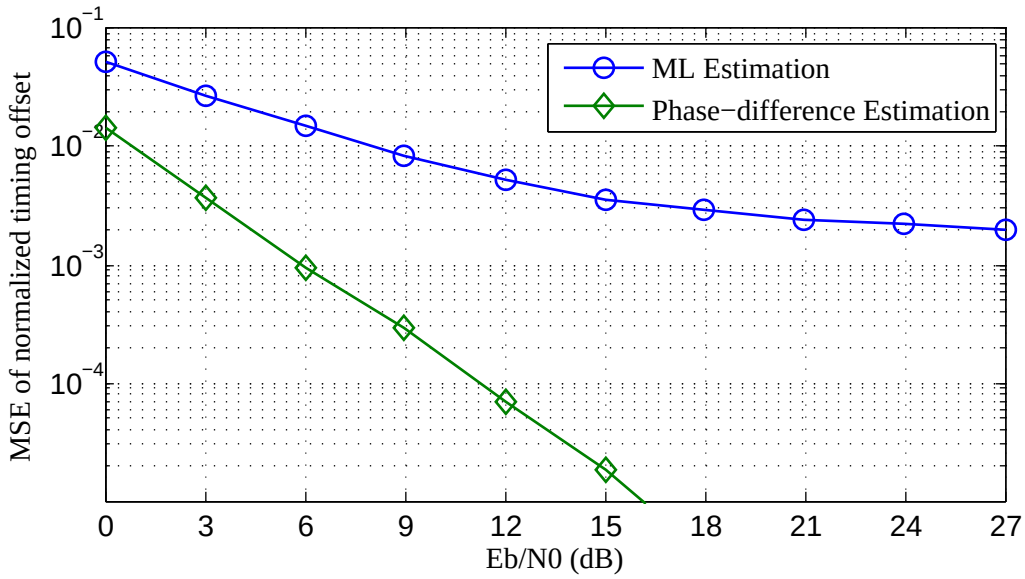


Figure 35: Simulation result of the mean-square error of the two proposed integer CFO estimators in the frequency-flat fading channel.

output as

$$\hat{\eta} = \max\left(\left\{ \sum_{k=1}^{N} \left|\tilde{d}_m(k+\hat{\eta}_m) - d_m(k)\right|^2 : m = [\tilde{m}_c - W_B, \tilde{m}_c + W_F] \right\}\right).$$

Figure 35 shows the mean-square error of the two proposed integer CFO estimators in the frequency-flat fading channel. In the frequency-flat fading channel, where the channel phases are constant over all subcarriers, the phase-difference estimator outperforms the ML estimator. The performance gap might be reduced by that the channel is getting frequency-selective.

6.3.4 Fine timing estimation for each relay

By the shifting property of DFT, the time offset from the actual preamble location causes phase rotation across subcarriers of the DFT output. Assuming that the combined fractional and integer part of CFO is already estimated compensated, we propose a frequency-domain symmetric correlation (FSC) to estimate the fine preamble timing within the searching window for each relay. The proposed timing metric for the k^{th} relay using FSC is defined

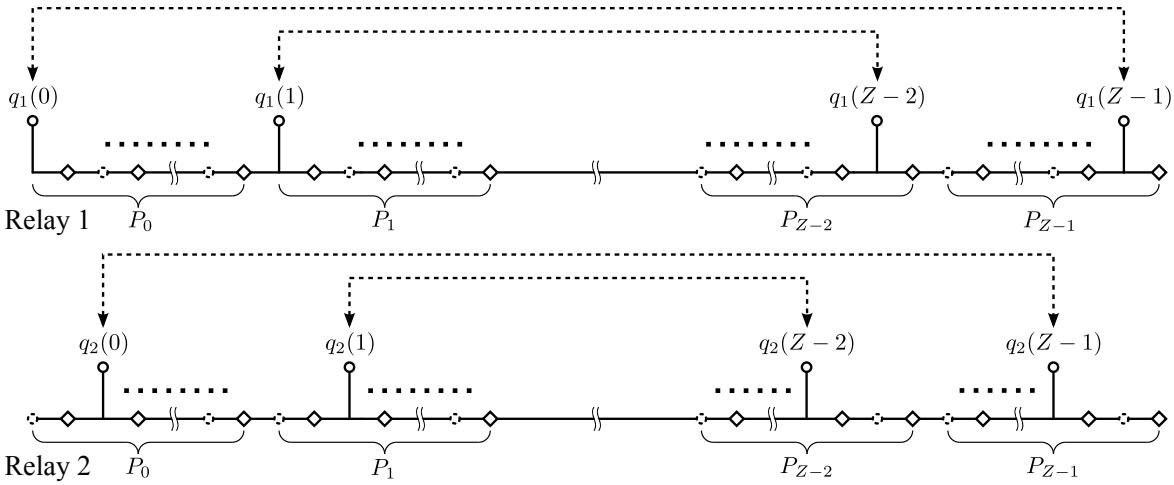


Figure 36: Illustration of the frequency-domain symmetric correlation of two relays out of K.

as

$$A_k(m) = \Bigg| \sum_{l=0}^{L/2-1} R_m^*(Q + 2(Kl + k)) \cdot R_m(N - Q - 2(Kl + k) - 1) \Bigg|. \tag{52}$$

For the k^{th} relay, the fractional index of SOP timing can be estimated using quadratic interpolation as

$$\tilde{m}_k = \frac{1}{2} \frac{A_k(\bar{m}_k - 1) - A_k(\bar{m}_k + 1)}{A_k(\bar{m}_k - 1) - 2A_k(\bar{m}_k) + A_k(\bar{m}_k + 1)},$$

where

$$\bar{m}_k = \arg\max_m \{A_k(m)\}.$$

The interpolated peak value of $A_k(m)$ becomes

$$\tilde{A}_k(\tilde{m}_k) = A_k(\bar{m}_k) - \frac{1}{4}(A_k(\bar{m}_k - 1) - A_k(\bar{m}_k + 1))\tilde{m}_k.$$

From (51) and (52), the timing metric $A_k(m)$ at the actual SOP-time location, that is zero, becomes

$$A_k(0) = \Bigg| \sum_{l=0}^{L/2-1} \Big\{ \Psi_k^*(Q + 2(Kl + k)) \cdot \Psi_k(N - Q - 2(Kl + k) - 1) \cdot \left|q(l)\right|^2 \Big\} \Bigg|.$$

By constructing $|q(l)|^2 = 1$ for all l, $A_k(0)$ can be rewritten as

$$A_k(0) = \left| \sum_{l=0}^{L/2-1} \left\{ \Psi_k^*(Q + 2(Kl + k)) \cdot \Psi_k(N - Q - 2(Kl + k) - 1) \right\} \right|.$$

Assuming the frequency-flat fading channel, $A_k(0)$ becomes $A_k(0) = \frac{L}{2} \cdot |h_k|^2$. Therefore, the combined SOP-time can be obtained by

$$\hat{m} = \frac{\sum_{k=1}^{K} \tilde{A}_k(\tilde{m}_k)\tilde{m}_k}{\sum_{k=1}^{K} \tilde{A}_k(\tilde{m}_k)}.$$

6.4 Simulation Study of Time and Frequency Estimation

The MSE performance of the proposed method when $K = 2$ is evaluated by the computer simulation. The OFDM system with 256 subcarriers and frequency-selective channel with the exponential power delay profile model [61] are used. The GI length normalized to the sampling duration, is eight. To evaluate the performance of the proposed algorithm, we consider two scenarios that represent a conventional MIMO-OFDM system where two transmitters are perfectly aligned in time and frequency, and a DMIMO-OFDM system where there exist time and frequency errors. For the first scenario, Therefore, in the first scenario, the time offsets between two transmit antennas, and CFOs for both antennas are set to zero. For the second scenario, the transmission time of each relay is a zero-mean Gaussian random variable with variance σ_ϵ^2=1. In addition, the normalized CFOs of the first transmitters are deterministically set to 0.05 and -0.05, respectively. The detailed simulation parameters are presented in Table 6.

The proposed method is compared with Schmidl&Cox's [60] and Park's [62] methods, which are labeled as 'S&C' and 'Park' in the simulation result, respectively. Representing the simple extension of the single-antenna-based estimation methods, these methods assign the same preamble to the two transmitters keeping the same transmit power per node. On the other hand, Ding's method [63] is designed to exploit transmit diversity by using an orthogonal preamble structure.

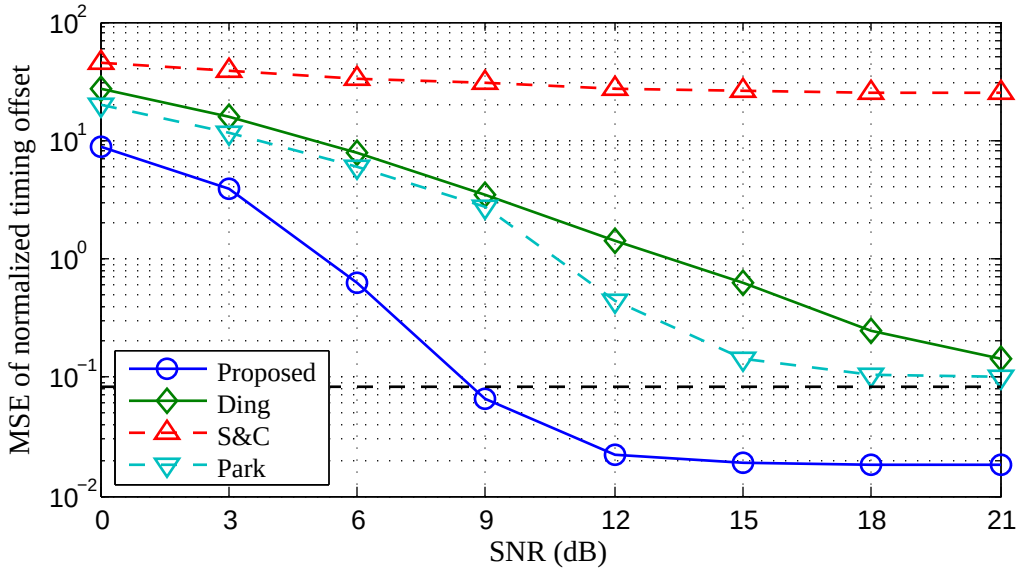
Table 5: Simulation parameters for OFDM-based CCT.

Parameters	Value
FFT size	256
Guard interval	8
FFT sampling frequency	10 Mhz
Subcarrier spacing	37.8 KHz
PN sequence	M-sequence

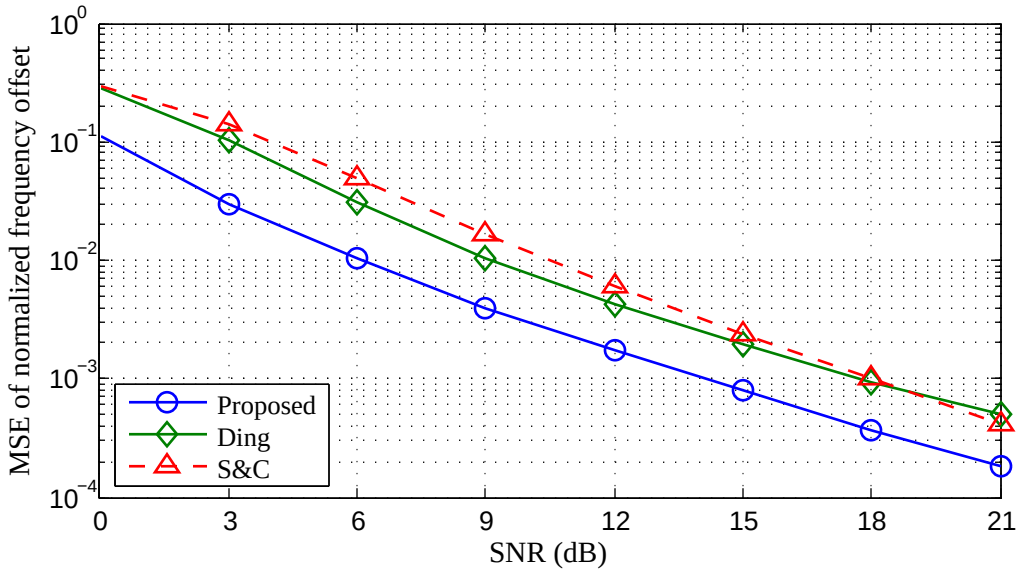
The simulated MSE performances of the time and frequency estimation are shown in Figure 37a and 37b for the conventional MIMO-OFDM scenario, and in Figure 38a and Figure 38b for the DMIMO-OFDM scenario. The timing error in Figure 37a and Figure 38a are normalized by the sampling period. The result shows that the proposed method outperforms the others in both scenarios. Even in the conventional MIMO-OFDM case, the construction of orthogonal preambles in the proposed method exploits spatial diversity to estimate the correct timing. We also observe that the S&C method has high MSE compared to the others. While this error does not affect the OFDM decoding-performance as long as it less than the length of the GI, it would create a large delay spread when multiple relays rely on this estimate for DMIMO. In addition, Park's method has better MSE performance than Ding's, despite that Ding's method is designed to exploit transmit diversity. It can be also observed that all the methods have an error floor. The error floor, denoted as a black dashed line in the figure, is due to the discrete sampling. The error floor of the proposed method, however, is less than the error floor of the others because the timing between two samples is estimated by the quadratic interpolation.

Figure 37b and Figure 38b show the MSE performance of the frequency estimation methods. For this simulation, the Park's method is excluded because its frequency estimation performance is identical to the S&C's method. The simulation result clearly shows that the proposed method is superior than the others. In Section 6.3.2, we claimed that the estimation of fractional CFO in the proposed method is equivalent to the S&C method. Despite this fact, the proposed method has 3dB gain than the S&C method. This gain comes

from that the preambles in the S&C method are not orthogonal so that the second term in (45) is remained.

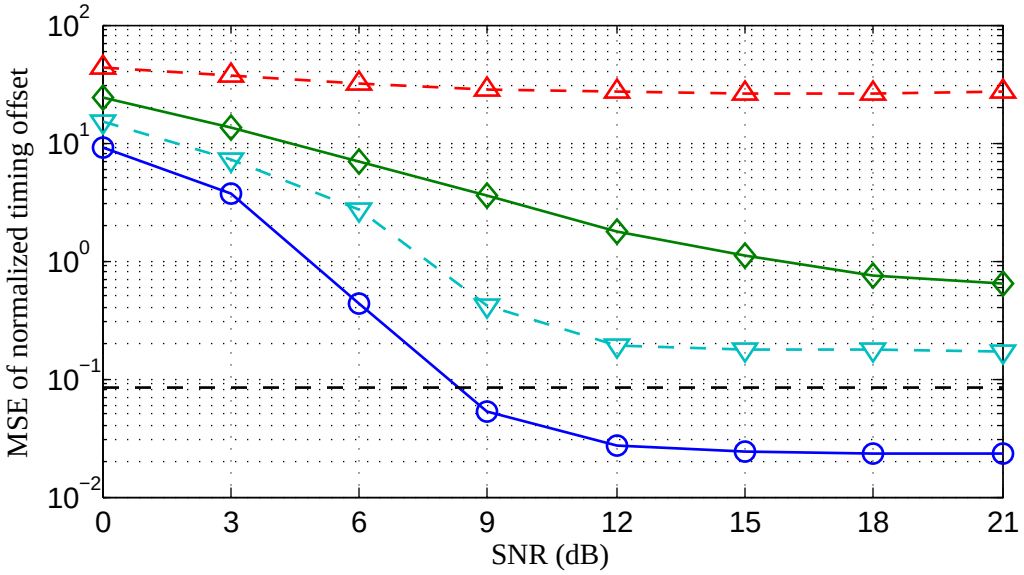


(a)

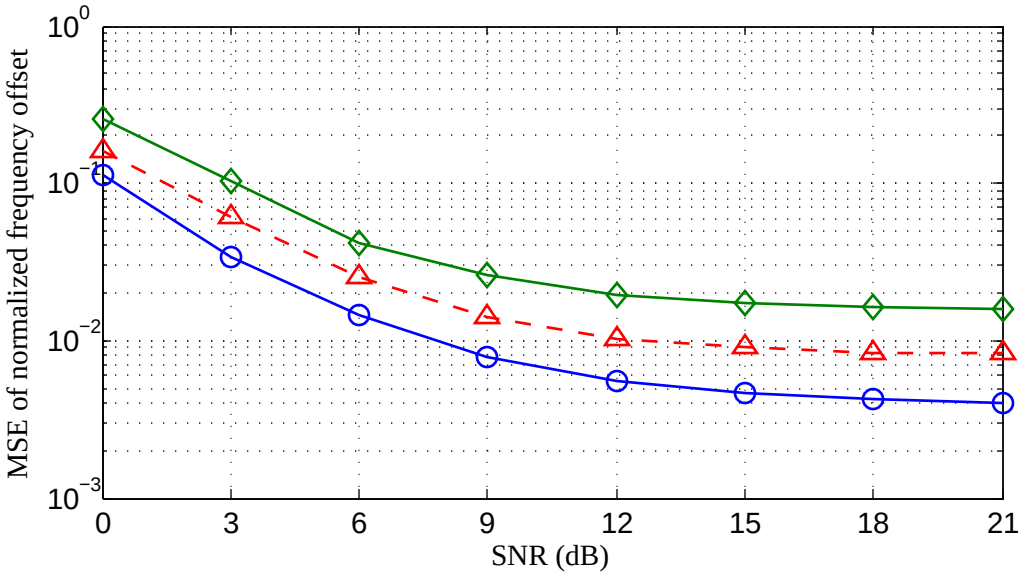


(b)

Figure 37: Simulation result of the proposed time estimation method with $\omega_1 = 0$, $\omega_2 = 0$, and $\sigma_\epsilon^2 = 0$ in the frequency-selective fading channel.



(a)



(b)

Figure 38: Simulation result of the proposed time estimation method with $\omega_1 = 0.05$, $\omega_2 = -0.05$, and $\sigma_\epsilon^2 = 1$ in the frequency-selective fading channel.

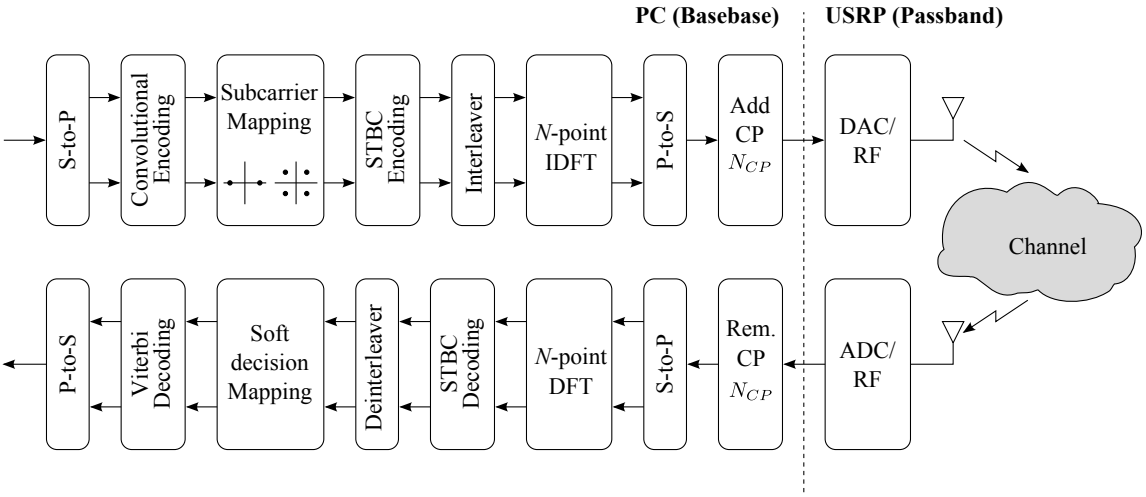


Figure 39: Illustration of the OFDM transmitter and receiver.

6.5 Implementation of the DMIMO-OFDM system

In this section, we first describe the rest of the DMIMO-OFDM system, which are adopted from the existing techniques to construct the SDR testbed. Then, later in this section, we propose two frame structures that are designed for the DF-based multi-hop CCT system for spatial diversity and the AF-based multi-hop CCT system for spatial multiplexing. The DMIMO-OFDM system is designed not only to combat the frequency-selective fading environment taking advantage of OFDM, but also to be spectrally efficient compared to the BFSK-based CCT system in Chapter 5. To exploit the bandwidth efficient spatial diversity across distributed transmitters, a distributed space-time block code (DSTBC) is used in the implementation. In addition, forward error correction with interleaving is combined with the DSTBC-OFDM.

6.5.1 OFDM System and Subcarrier Structure

It is well known that the modulation and demodulation of OFDM can be realized in the discrete-domain by using the DFT and IDFT. The overall block diagram for the data transmission is illustrated in Figure 39. The stream of data bits are de-multiplexed into P parallel bits. The length of the parallel bits is determined by the modulation order and the convolutional code, which will be discussed in the following subsection. Depending upon the

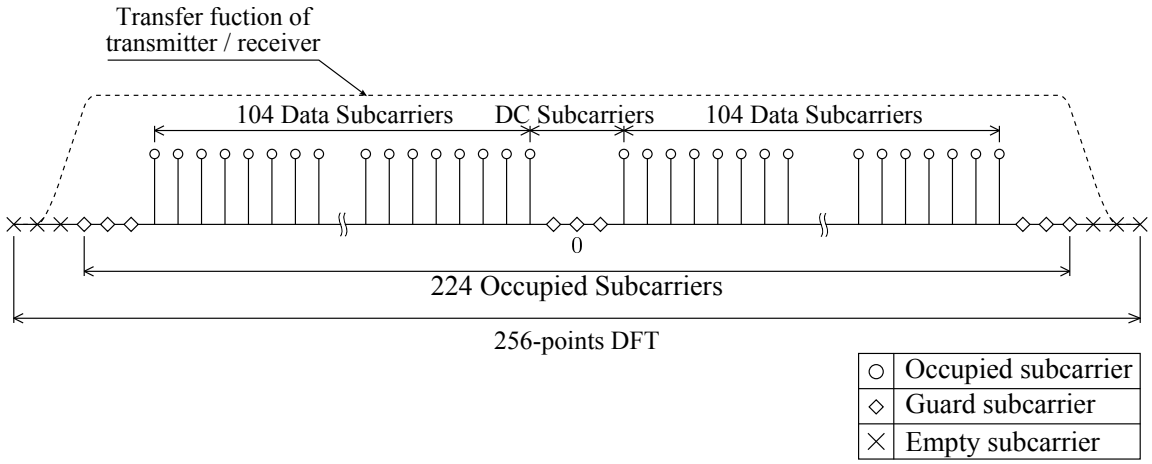


Figure 40: Illustration of the OFDM structure in the implementation.

modulation order, the parallel bits are packed with $\log_2(M)$ bits in the M-ary PSK modulation, and mapped to the complex values in the signal constellation producing one OFDM symbol. In the STBC encoder block, a number of OFDM symbols are stored in memory to generate the space-time coded OFDM symbols. Then the frequency-domain signal is converted into the time-domain signal, and a CP is added to each OFDM symbol. The constructed time-domain signal is sent to the USRP board to be transmitted over the air.

In the receiver, the USRP converts the passband to the baseband signal. Assuming the OFDM symbol timing is already estimated at the receiver, $N + N_{CP}$ time-domain samples are captured from the received baseband signal. After removing the CP, the N samples are converted to the frequency-domain signals through the DFT block producing the frequency-domain OFDM symbol. In the STBC decoding block, a number of frequency-domain OFDM symbols are stored and combined to exploit the spatial diversity. The output of the STBC decoder block are de-interleaved and mapped to the soft-decision values for the soft-decision Viterbi decoding. It is noted that the symbol de-mapper is not required in this system because the Viterbi decoder decodes and demodulates the soft-values symbols simultaneously.

The design of the OFDM subcarriers is depicted in Figure 40. To reduce the computational complexity in the baseband signal processing, the received signal is sampled close to

the Nyquist frequency at the USRP board. The subcarriers close to the Nyquist frequency will be attenuated by the DUC and DAC at the transmitter, and by the DDC and ADC at the receiver [64]. Thus, the subcarrier close to the boundary cannot be used for data transmission (see the 'x'-marks in the figure). In addition, as we discussed in Section 6.2, the CFO between a transmitter and a receiver shifts the received OFDM subcarriers in the frequency domain. Therefore, the guard bands are added to the both sides of the subcarriers to embrace the frequency shift. Furthermore, the USRP board suffers from direct current (DC) offset, which distorts the subcarriers around the frequency zero. Therefore, we do not use some subcarriers around DC to avoid the DC offset.

Though the flexibility of the SDR platform allows us to use any arbitrary FFT length, occupied and data subcarriers, it is noted that the OFDM parameters presented in Table 6 are successfully tested in a generic Dual-core 2.6 GHz PC to run the proposed transceiver in real time with 2 Mhz sampling rate.

Table 6: OFDM parameters for the experimental testbed.

Parameters	Values
Modulation	BPSK, QPSK
FFT Length	256
Occupied subcarriers	224
Used subcarriers	208
CP length	8
Subcarrier spacing	3.8 Khz, 7.6 KHz
Sampling rate	1 Msps, 2 Msps

6.5.2 Distributed Space-Time Block Coding for Spatial Diversity

In a MIMO system, Space-time trellis coding (STTC) was introduced in [65] as an effective transmit diversity technique to combat fading. Since the decoding complexity of

STTC increases exponentially with the transmission rate, STBC [17, 66] was proposed as an alternative to its trellis counterpart with a lower decoding complexity. To combat frequency-selective fading, OFDM has been applied to STTC in [67] and to STBC in [68].

In a distributed MIMO system, the distributed space-time block code (DSTBC) was proposed to achieve full spatial diversity [18]. Combining OFDM with DSTBC has been considered to exploit spatial diversity for cooperative systems in a frequency-selective fading environment [69, 70]. For the testbed, we implemented the unit-rate Alamouti STBC [17] and 3/4-rate orthogonal STBC [71] for $K = 2$ and $K = 4$ cases respectively.

The coding matrices for the Alamouti STBC and the 3/4 OSTBC code in a MIMO system are

$$C_2 = \begin{bmatrix} c_1 & c_2 \\ -c_2^* & c_1^* \end{bmatrix} \tag{53}$$

and

$$C_4 = \begin{bmatrix} c_1 & c_2 & c_3 & 0 \\ -c_2^* & c_1^* & 0 & c_3 \\ -c_3^* & 0 & c_1^* & -c_2 \\ 0 & -c_3^* & c_2^* & c_1 \end{bmatrix} \tag{54}$$

where each column is assigned to each transmit antenna while each row represents the time-slot. It is readily shown that the Alamouti code is a rate-1 code because two time-slots are taken to transmit two symbols, while the 3/4 OSTBC takes four time-slots to transmit three symbols. For a DSTBC system, one of the columns of the STBC can used in the corresponding node [18], or a random combination of the columns can used to avoid the column coordination [72]. These two methods can be realized by using the combining vector, which represents the weighting coefficients of the STBC columns. The combining vector for the k^{th} node is defined as

$$\mathbf{g}_k = [g_{k,1}, g_{k,2}, \ldots, g_{k,K}]^T \tag{55}$$

where

$$\sum_{j=1}^{K} g_{k,j} = 1. \tag{56}$$

The transmitted symbols for the k^{th} node are generated by

$$\mathbf{x}_k = C_K \cdot \mathbf{g}_k. \tag{57}$$

The elements $g_{k,j}$ of the combining vector determines the weights of the STBC columns depending upon the column selection methods. For example, the first and the second node in a distributed Alamouti STBC system can choose $\mathbf{g}_1 = [1, 0]^T$ and $\mathbf{g}_2 = [0, 1]^T$ respectively.

6.5.3 Forward Error Correcting for Frequency Diversity

OFDM is typically used in conjunction with channel coding, so called FEC, with interleaving [73, 74]. In a frequency-selective fading environment, subcarrier interleaving ensures that the bit errors that result from the subcarriers in the faded part of the bandwidth are spread out rather than being concentrated. Then, the spread bit errors are corrected by the error correction decoder.

For the testbed in this chapter, the three-stage $1/2$-rate $(5, 7)$ and five-stage $1/2$-rate $(23, 35)$ convolutional encoders are implemented for FEC, which are shown in Figure 41. The FEC is applied to the information bits in an OFDM symbol with a helical interleaving. The amount of information bits required for an OFDM symbol is calculated in Table 7.

Table 7: Information bits per OFDM symbol.

	3-stage Convolutional code	5-stage Convolutional code
BPSK	102	104
QPSK	202	204

Based on the modulation order and the convolutional code, the required information bits are multiplexed from the data bits and encoded with the convolutional encoder. The

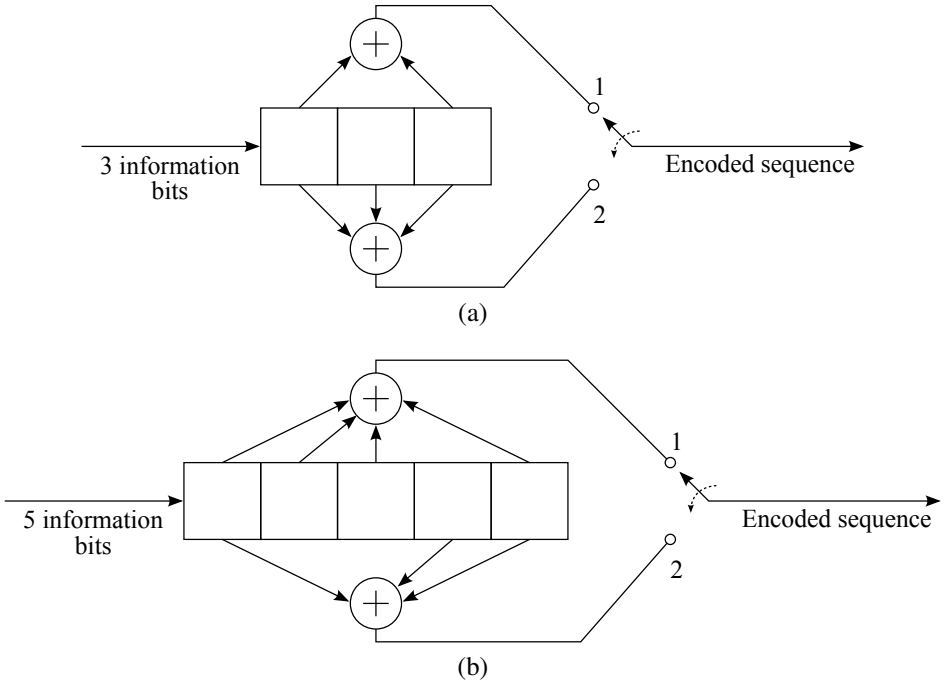


Figure 41: Three-stage and five-stage convolutional encoder.

encoded bits are mapped to the corresponding constellation points, and interleaved across subcarriers. The interleaved symbols are coded with the STBC and converted into the time-domain signal through the IDFT.

At the receiver, the soft-decision Viterbi decoder is used for convolutional decoding. Assuming the timing synchronization is already achieved, an OFDM symbol is captured and converted to the frequency domain signal. We also assume that the complex signal in each subcarrier is divided by the corresponding frequency-domain channel gain so that the AWGN is remained in the constellation. The complex signal in data subcarriers are de-interleaved and de-mapped to the 3-bit soft-decision output based on the modulation order. The 3-bit quantization for a BPSK signal is illustrated in Figure 42. Let $Q(\cdot)$ denote the 3-bit quantization function. The soft-decision value V_n for the n^{th} subcarrier signal S_n can be obtained as

$$V_n = Q\Big(\mathrm{Re}\{S_n\}\Big). \tag{58}$$

For $\frac{\pi}{4}$-QPSK, the Gray mapping is used to simplify the soft-decision of the received

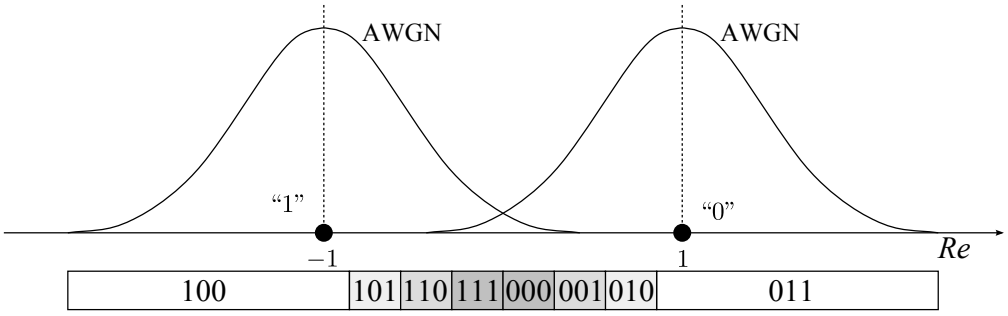


Figure 42: Quantization zones for a 3-bit soft decision for BPSK signal.

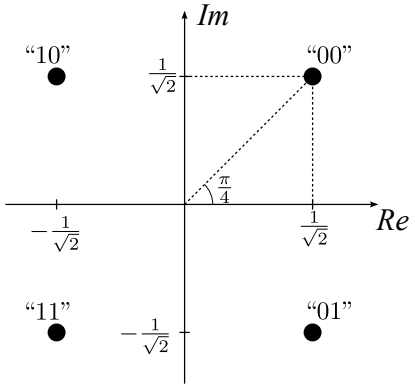


Figure 43: $\frac{\pi}{4}$-QPSK constellation map.

signal as shown in Figure 43. Two convolutionally coded bits are mapped to the corresponding constellation point at the transmitter. From the Gray mapping, the detection of the first bit and the second bit can be separated by taking the real and imaginary part of the constellation point as shown in Figure 43. Therefore, at the receiver, the soft-decision values for the first and the second bit can be obtained by

$$V_{n,1} = Q\Big(\operatorname{Re}\{S_n\}\Big)$$
$$V_{n,2} = Q\Big(\operatorname{Im}\{S_n\}\Big).$$

6.5.4 Frame Structure

In this section, we propose two examples of the OFDM frame structure for a DF-based DMIMO-OFDM system and an AF-based DMIMO-OFDM system. Figure 44 depicts the two multi-hop CCT systems we consider in this section.

The upper figure illustrates the DMIMO-OFDM DF relaying strategy, which exploits

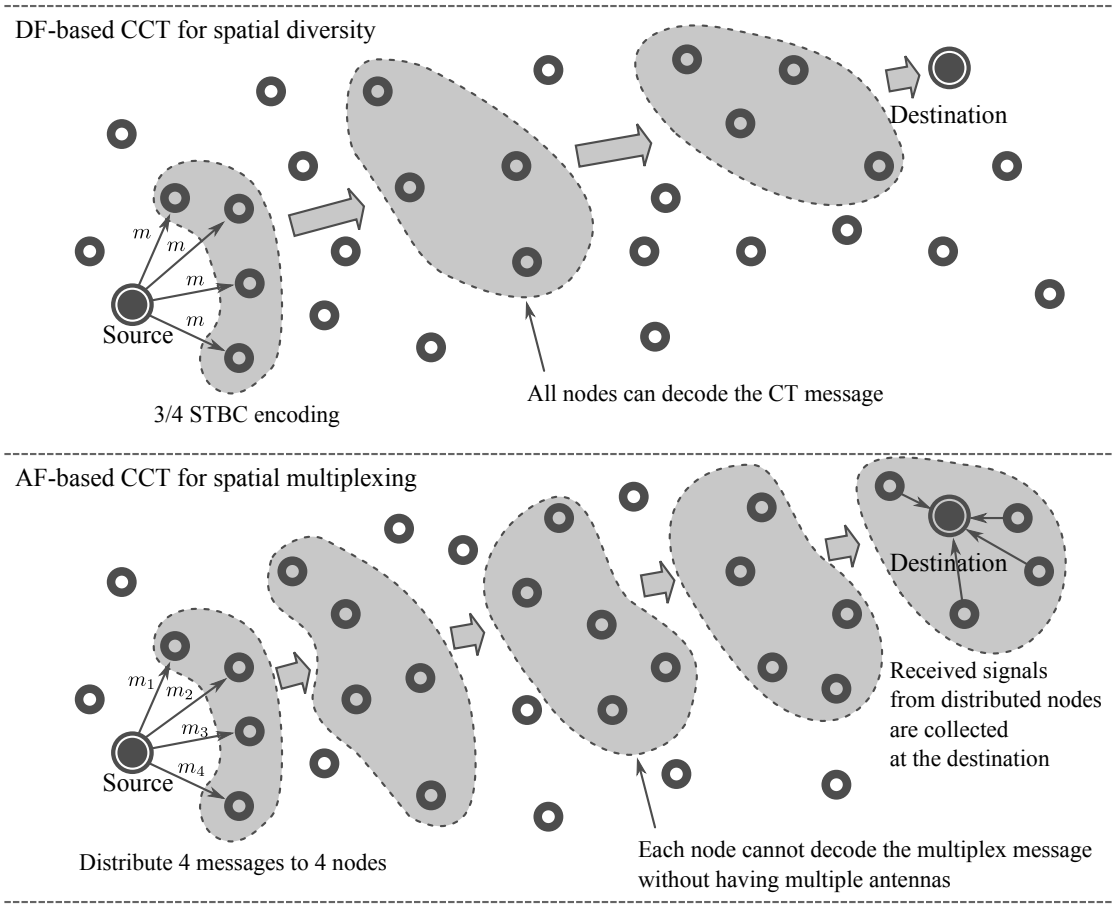
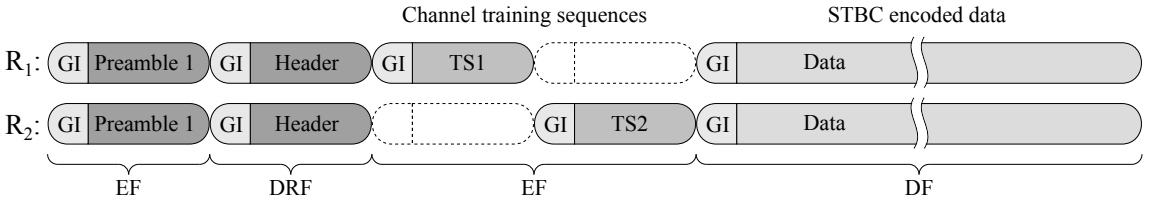


Figure 44: DF-based multi-hop CCT system to exploit spatial diversity vs. AF-based multi-hop CCT system to exploit spatial multiplexing.

the spatial diversity. In the first hop, the source broadcasts the same message m to each relay in the first cluster. As we discussed in Chapter 2, the clusters are formed opportunistically based on ability to decode correctly. This system can support the lower-layer functionalities for the OLA-based broadcasting and routing protocols [35, 38]. It is noted that the cooperative message transmitted from a cluster in this system can be decoded in any node in the next cluster as long as the received SNR exceeds the decoding threshold because the simultaneously transmitted messages are diverse versions of the same message. The frame structure for this system is illustrated in Figure 45. The figure illustrates only the example of two relays with the Alamouti STBC, but without loss of generality, the source node can choose any of the two frame structures to initiate the transmission.

Let us suppose R_1 and R_2 already received the source message and decoded it correctly.

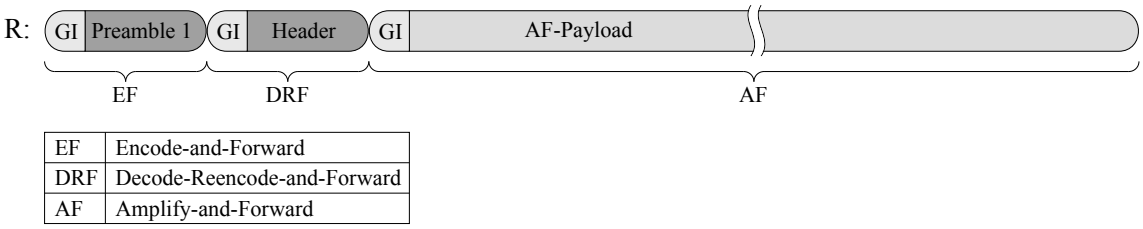


EF	Encode-and-Forward
DRF	Decode-Reencode-and-Forward
DF	Decode-and-Forward

Figure 45: OFDM frame structure for the DMIMO-OFDM AF system.

Since the preamble structures are commonly shared across all the nodes in the network, the preamble is simply encoded in each relay. The header symbol contains 1) the modulation and encoding information of the payload 2) the hop count information, and 3) the random sequence number generated from the source node. Usually the hop count information is a part of the network layer packet. We claim that the hop count information in the physical layer header has an advantage of faster relaying operation to avoid that the message is processed in the higher layer in intermediate relays. The hop count information should be updated as hop goes. Therefore, each relay decodes the header and updates the hop count information to create the retransmit message. Furthermore, the hop count information is used to stop the retransmission if the hop count exceeds the network limit threshold. The random sequence originated from the source node is used to avoid the broadcasting storm in the multi-hop network. Each node records the sequence number whenever the node relays a message. Because of the broadcasting nature of the wireless system, a cluster will overhear when the next cluster relays the packet. Each node, then, can reject the overheard message by comparing the recorded sequence number with the decoded sequence number in the header.

In order to avoid the unnecessary decoding operation that consumes energy, the header needs to be decoded prior to the following procedures. Since there is no channel information up to this stage, the header is non-coherently encoded with the differential BPSK (DBPSK). Once the modulation and encoding information is decoded in the header, the



EF	Encode-and-Forward
DRF	Decode-Reencode-and-Forward
AF	Amplify-and-Forward

Figure 46: OFDM frame structure for the DMIMO-OFDM DF system.

node has the knowledge about how the rest of the packet is encoded. The header information also includes the order of the diversity, which determines the number of training sequences and the STBC type. We assume that all the nodes in the network use the same training sequences, therefore the training sequences are also encoded in each node and forwarded. Trivially, the data is decoded based on the modulation and encoding information from the header, and re-encoded to be forwarded.

The lower part of Figure 44 depicts the other multi-hop CCT system, where the AF relaying strategy is used to exploit distributed spatial multiplexing (DSM). DSM is considered as a means for high-throughput in an ad hoc network [36]. In the figure, the source distributes the four different messages $\{m_1, m_2, m_3, m_4\}$ to the four node in the first clusters. The distribution can be accomplished by transmitting four different packets in turns, or in such a way that each relay in the cluster just takes the corresponding part of the source packet partially. Each node in the first cluster, simply re-encodes its own message similar to V-BLAST [75], and forwards to the next cluster. Since the cooperative message from the first cluster is a superimposed signal of four different messages, no node in the second cluster can decode the message without having multiple antennas. Therefore, each node simply performs AF on the received message. In the last cluster, the destination node collects the recording of the received signal from its neighbors. In this phase, the collection has to be done orthogonally in time or frequency to utilize the neighbors as if they were multiple antennas in a conventional MIMO system. Then, the destination node de-multiplexes the source message.

Figure 46 shows the frame structure proposed for this system. Even though the AF

strategy is assumed to be a simple relaying scheme, which avoids energy-consuming decoding processes, we claim that it involves significant decoding processes in the real-world implementation. First, the AF packet needs to be detected in a relay node. It is unrealistic that relays always amplify and forward no matter what they receive, specifically in a half-duplex system. The packet detection can be realized by the proposed method in this chapter. In addition, as same as the DF example, the hop count information and the sequence number are required to avoid the aforementioned problems. Therefore, the header has to be decoded and re-encoded in the relay nodes. It is pointed out that the timing estimation and CFO correction has to be performed to decode the header even in the AF relaying system.

On both of the aforementioned systems, the pre-synchronization of time and frequency has to be employed for multi-hop CCT. Since the proposed method only requires a single OFDM preamble, it can be applied to both of the DF- and AF-based CCT systems.

6.6 Experimental Study of Pre-synchronization for OFDM-based CCT

The performance of the proposed method is evaluated on the experimental testbed on GNU Radio and USRPs. The same OFDM parameters as in the simulation have been used. For the performance metrics, we define the RMS transmit time spread (RTTS) and RMS transmit frequency spread (RTFS) as

$$\sigma_\epsilon = \sqrt{\frac{\sum_k^K (\epsilon_k - \hat{\epsilon})^2}{K-1}} \tag{59}$$

where ϵ_k is an estimated quantity of time or frequency at the k^{th} relay node and $\hat{\epsilon}$ is the sample mean of the K time or frequency estimates. The measurement was taken in the Smart Antenna Research Laboratory of Centergy Building, Georgia Institute of Technology. The source, the observer and the two relay nodes, denoted as 'S', 'O', 'R_1', and 'R_2' respectively, are deployed in the lab as shown in Fig. 47. Therefore, $K = 2$ in (59). A common clock is provided to the source and observer and another common clock is provided to the two relays, for measuring relative error only. In other words, the common clocks were

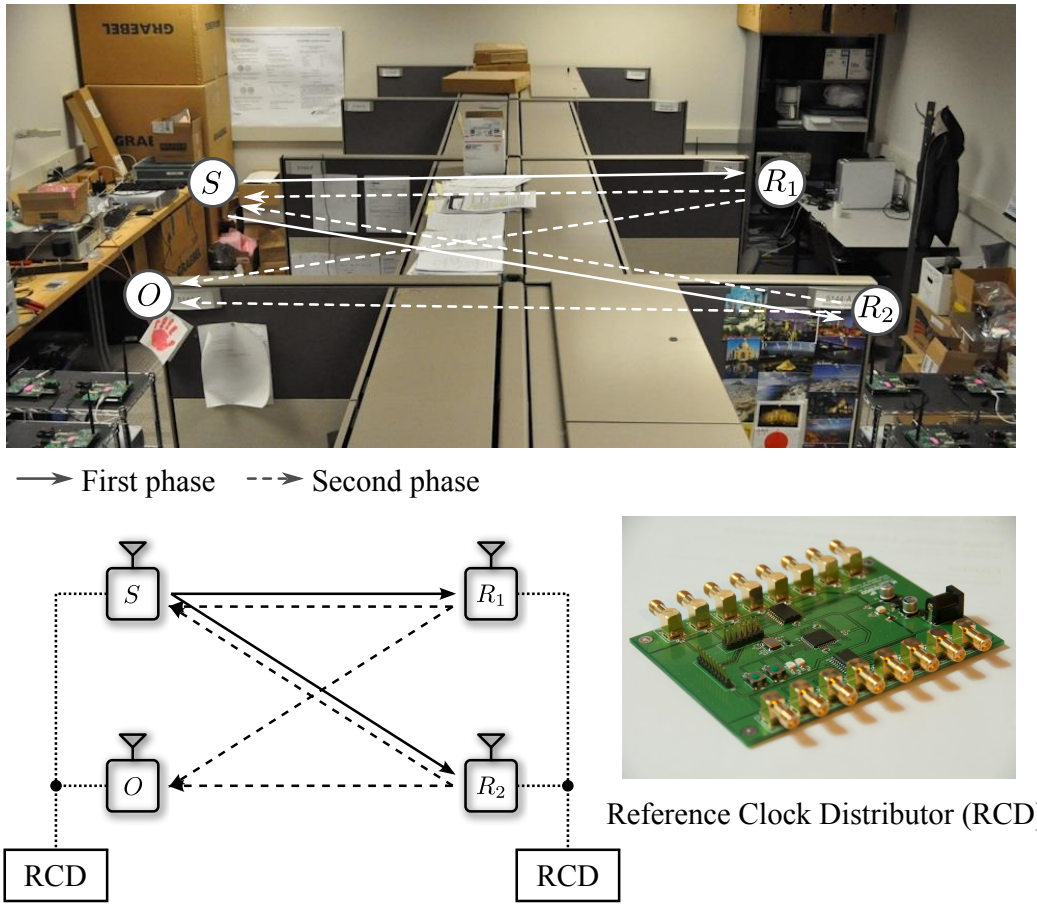
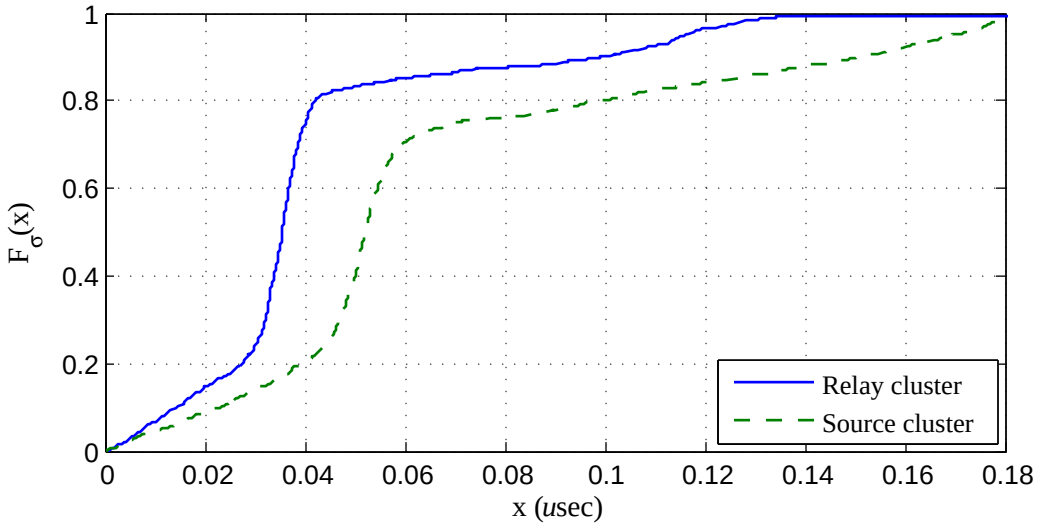


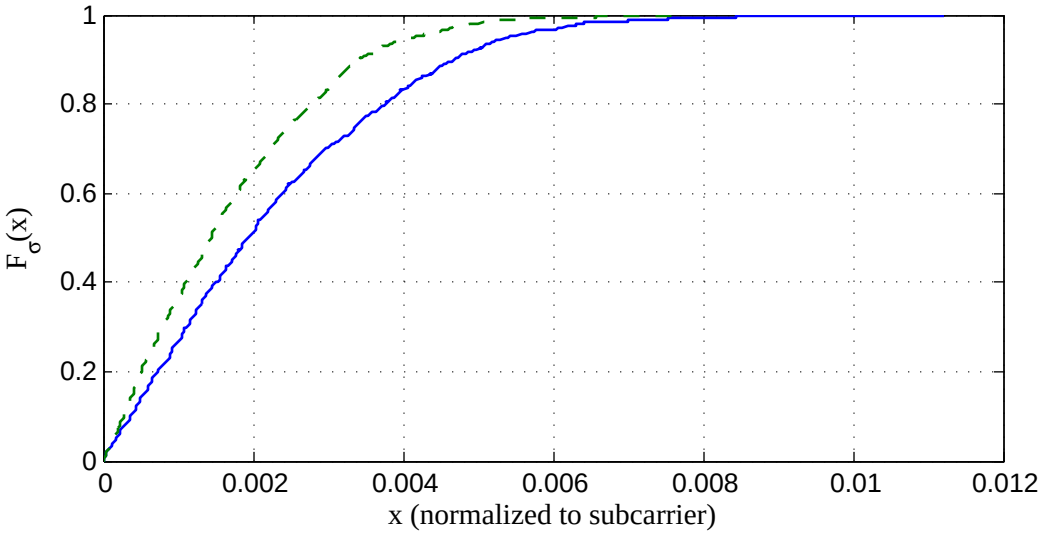
Figure 47: Empirical CDF of relative CFO between R1 and R2.

not used in the synchronization process, but only to measure synchronization performance. The transmit power in each node is set to 5 dBm. The source transmits a packet with the proposed preamble to the two relays. Each relay autonomously estimates the time and frequency from the received preamble, and retransmits the packet with the proposed pre-synchronization algorithm. Specifically, R_1 and R_2 choose different subcarrier slots in the proposed preamble and retransmit the source packet with the preamble. Next, the source and observer nodes estimate their time and frequency pre-synchronization parameters using the proposed method, as though they would be relaying. This round-trip transmission was repeated 1000 times and the estimated time and frequency data was collected in each cluster.

Fig. 48a shows the empirical cumulative distributed function (CDF) of the RTTS in each



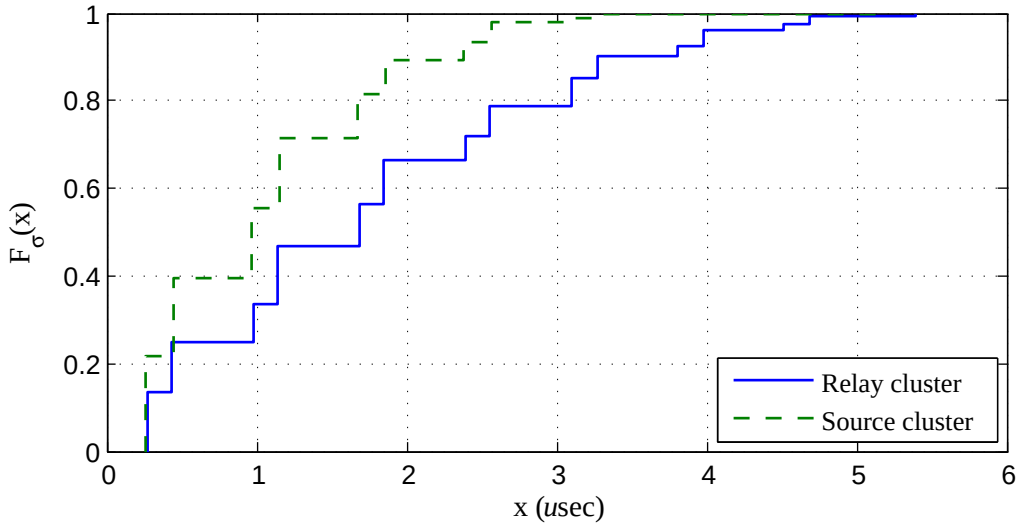
(a) RTTS



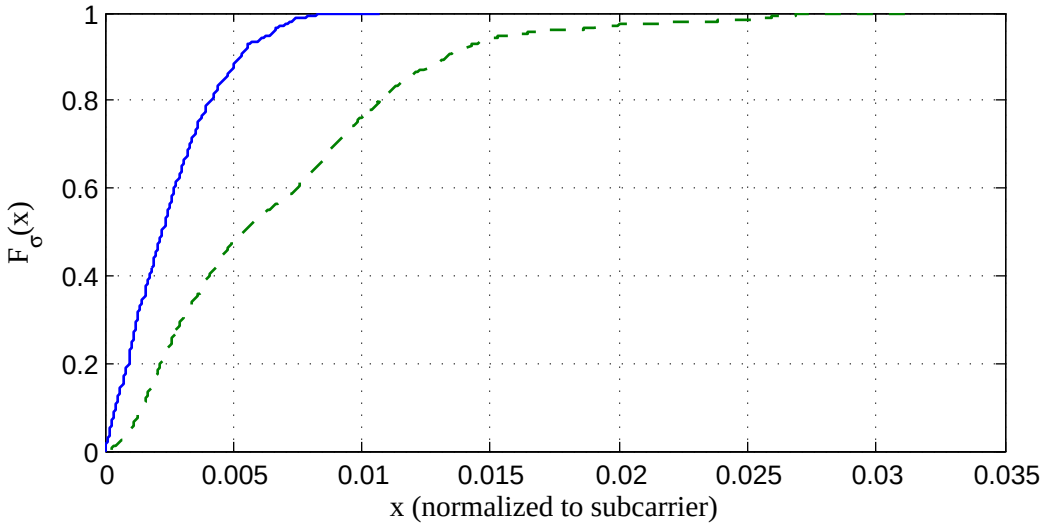
(b) RTFS

Figure 48: Empirical CDF of the (a) RTTS and (b) RTFS of the proposed method in Relay and Source cluster.

cluster. The measurement shows that the RTTS in the relay cluster is less than 0.13 μsec in 90% of the cases, which is relatively small compared to the sampling duration that is 1 μsec. It is noted that the bumpy shape of the RTTS curve is caused by a bimodal distribution of quadratic interpolation. The RTTS in the source cluster is slightly larger than the one in the relays cluster. This is because the reference for the synchronization in the source cluster's nodes is the superimposed received signals from the relay cluster, while the synchronization



(a) RTTS



(b) RTFS

Figure 49: Empirical CDF of the (a) RTTS and (b) RTFS of the Schmidl&Cox method in Relay and Source cluster.

in the relay cluster is based on a single transmission. The measurement shows that the RTTS is still less than 0.13 μsec in 90% of the cases. The RTFS of the source cluster is larger than the RTFS of the relay cluster by three times at the 90% level. As with RTTS, the errors at the source cluster are higher than the relay cluster because the synchronization is based on a superposition rather than just a single transmission. However, the RTFS of the source cluster is still less the 1.5% of the subcarrier spacing, at the 90% level. It is noted

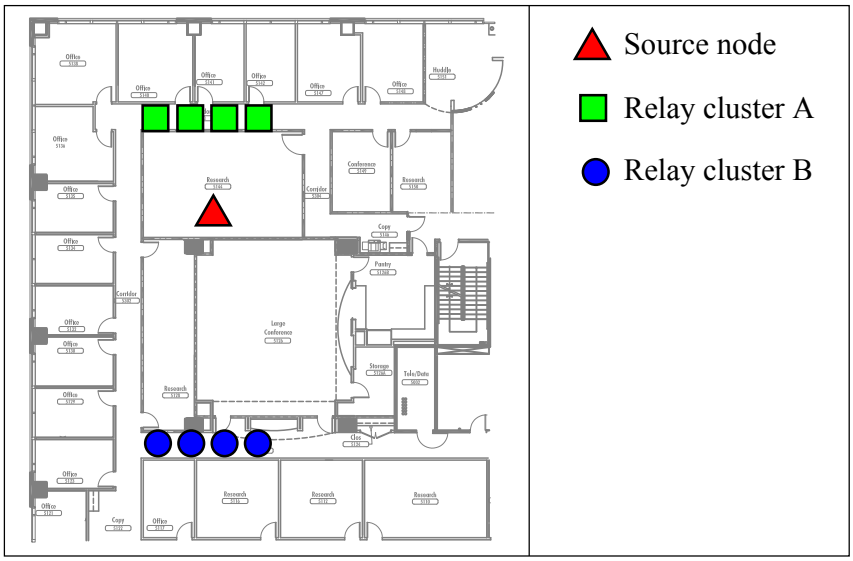


Figure 50: Network topology for the OFDM-based ping-pong experiment.

that the average packet delivery ratios (APDRs) in both clusters are one.

For the comparison, S&C method has been implemented and tested in Figure 49a and Figure 49b. As we learn from the simulation result, S&C method provides fair enough time estimation performance for decoding. In the relay cluster RTTS in Figure 49a, the timing error is still less than 6 μsec that is within the GI. Therefore, it was measured that the APDR in the relay cluster. However, it was measured that the APDR in the source cluster is 0.9. The packet error comes from the large spread of the transmit time in the relay cluster. It is pointed out that the RTTS in the source cluster looks better than the relay cluster in the figure because the RTTS data can be measured only when the data is successfully decoded.

6.7 Experimental Study of OFDM-based Multi-hop CCT

This experiment focuses on evaluating the proposed time and frequency pre-synchronization method for the OFDM-based CCT system over a multi-hop network. As we demonstrated in Section 5.5, the RTTS over multiple CCT hops statistically converges when the multiple SOP-time estimates are combined after being weighted by the channel gains. In this section, we design the "ping-pong" experiment to emulate multiple hops, similarly in Section 5.5. Figure 27 shows the network topology for the ping-pong experiment. It is

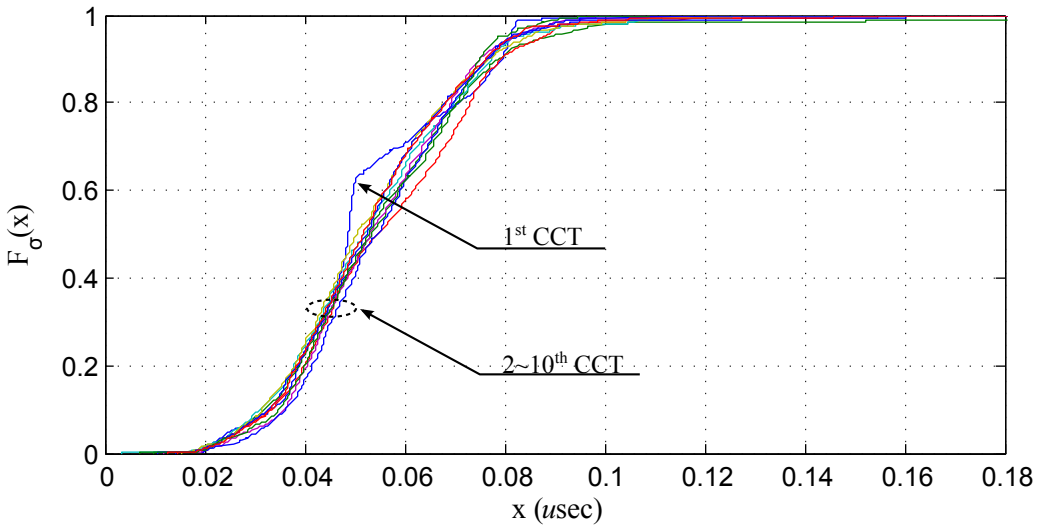


Figure 51: RTTS of ping-pong experiment for the proposed method.

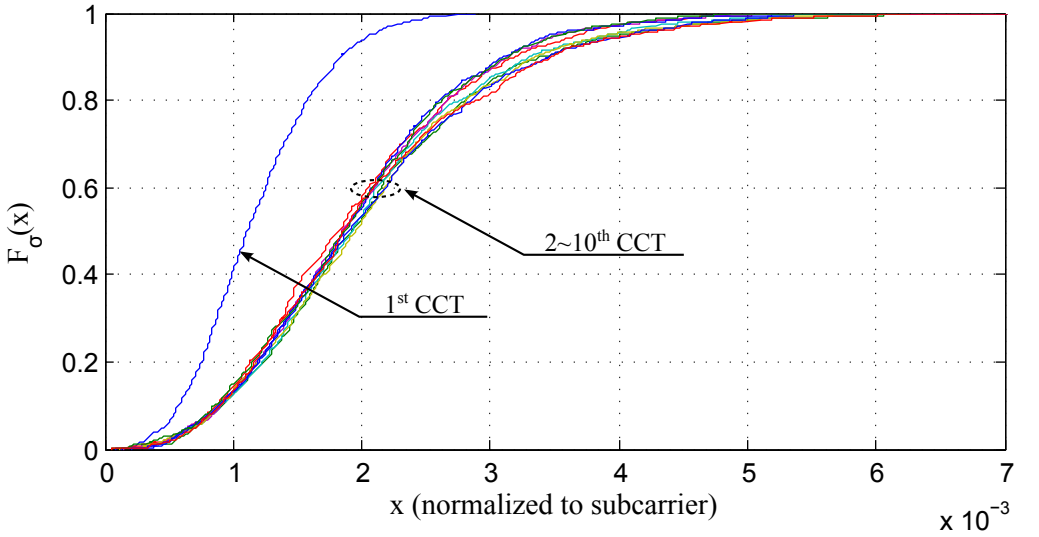


Figure 52: RTFS of ping-pong experiment for the proposed method.

noted that the inter-node distance in this experiment is smaller than the "ping-pong" experiment for the NCBFSK-based testbed in Section 5.5 because the measurement device cannot reach further than two meters.

The source node initiates a packet transmission through frequency 900 Mhz with 8 dBm transmit power. This single packet is transmitted back and forth between Cluster A and Cluster B up to the 10^{th} hop with per-node transmit power 5 dBm through the 904 Mhz and 908 Mhz center frequencies. The 10-hop experiment is repeated 1000 times. The RTTS

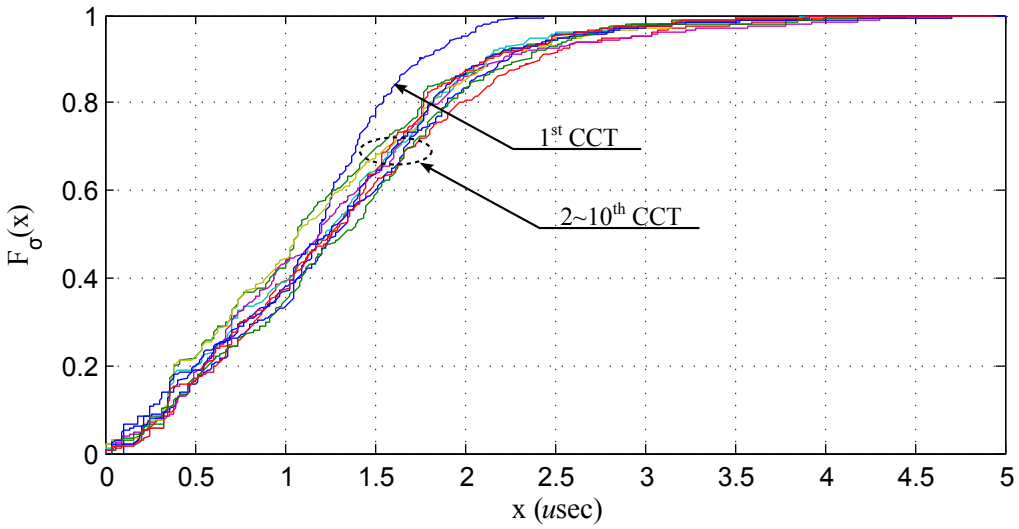


Figure 53: RTTS of ping-pong experiment for the Schmidl&Cox method.

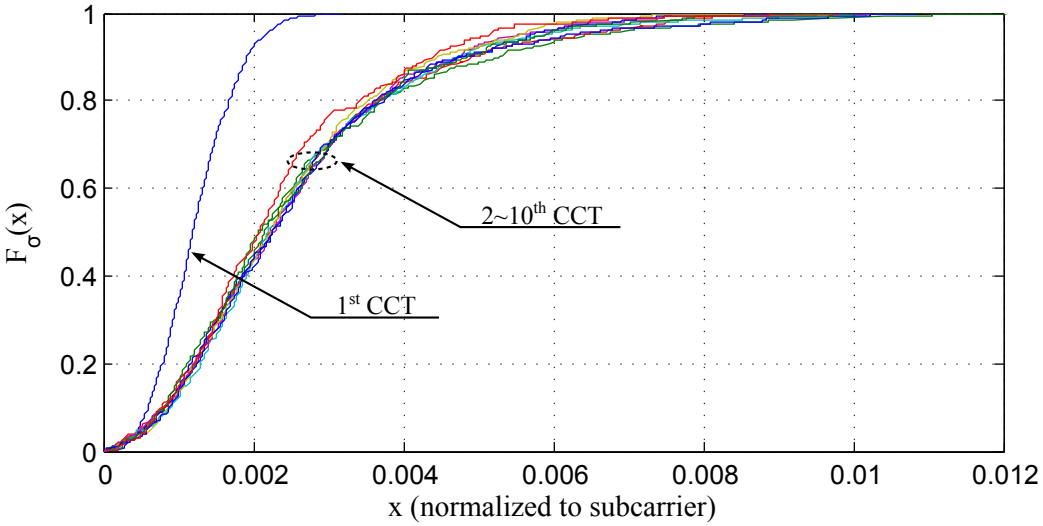


Figure 54: RTFS of ping-pong experiment for the Schmidl&Cox method.

and RTFS of each successive hop is measured and recorded. In addition, each node in the clusters records the detection results of the received packet to evaluate the packet delivery performance. The RTTS, RTFS, and packet delivery result are recorded in each node only when the packet is successfully detected and decoded without error. To investigate the effect of the pre-synchronization, the S&C method is also evaluated in the same topology and configuration. For this experiment, the BPSK modulation and the 5-stage convolutional code were used.

Figure 51 shows the empirical RTTS of each cluster for the proposed pre-synchronization. Each curve represents an empirical CDF of RTTS of each CT. As we demonstrated in Section 5.5, it is also observed that the CDFs of RTTS of all CTs essentially overlay each other, which shows the proposed pre-synchronization method can support the OFDM-based multi-hop CCT effectively. 90% of RTTS outcomes are less than 80 ns, which is 40% of the NCBFSK case.

Figure 52 shows the empirical RTFS of each cluster. In this measurement, it is observed that the first CDF curve, that represents the first cluster, shows better performance than the others do. This is because the source is close to the Cluster A so that the received SNR in the first cluster is relatively higher than the other ping-pong messages. As for RTTS, it is observed that the CDFs of RTFS of CTs 2 through 10 overlay each other. 90% of RTFS outcomes are less than 3.5×10^{-3}.

Figure 53 and Figure 54 show the empirical RTTS and RTFS of the S&C method. First, it is observed that the RTTS is order of μ sec for all hops. Since the RTTS and RTFS are recorded only when the received message is successfully decoded, the RTTS result of the S&C method implies that the timing errors of an order of μsec do not hinder the decoding of the OFDM signal. However, when the RTTS error is larger than the tolerance, the packet cannot be decoded and the RTTS result is not shown in the plot. The packet loss caused by large synchronization errors will be discussed later in this section. The RTFS result of the S&C method is 30% worse than the proposed method. This result is consistent with the simulation results in Section 6.4.

The packet loss caused by the synchronization error is shown in Figure 55. The figure shows the average number of the relays that can decode the ping-pong message successfully in each hop, as referred to as the 'active' relay in the plot. Let $A_n^{(i)}$ denote the number of active nodes in the n^{th} hop at the i^{th} trial. The average number of the active relays can be calculated as

$$E[A_n] = \frac{1}{\#Trials} \sum_{i=1}^{\#Trials} A_n^{(i)}.$$

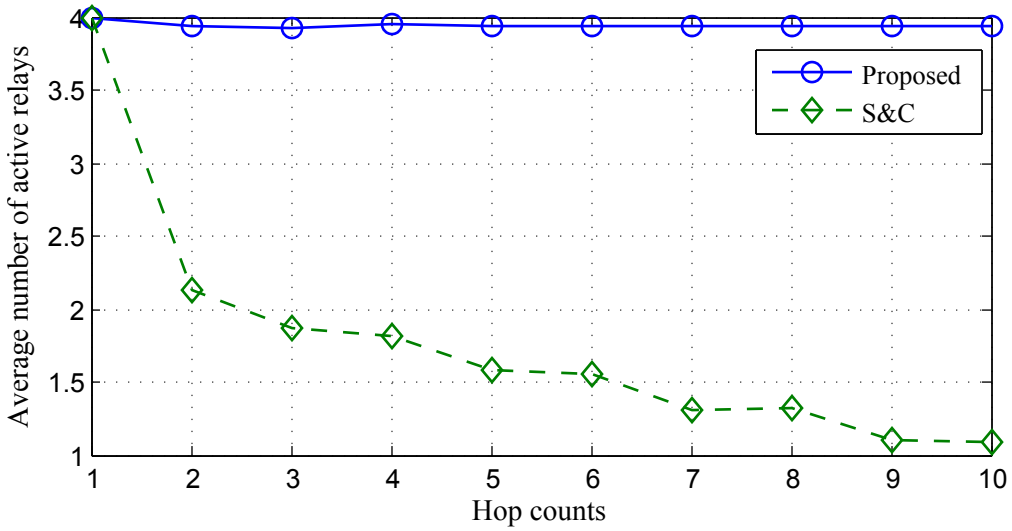


Figure 55: Average number of relay nodes that successfully decode the ping-pong message in each hop.

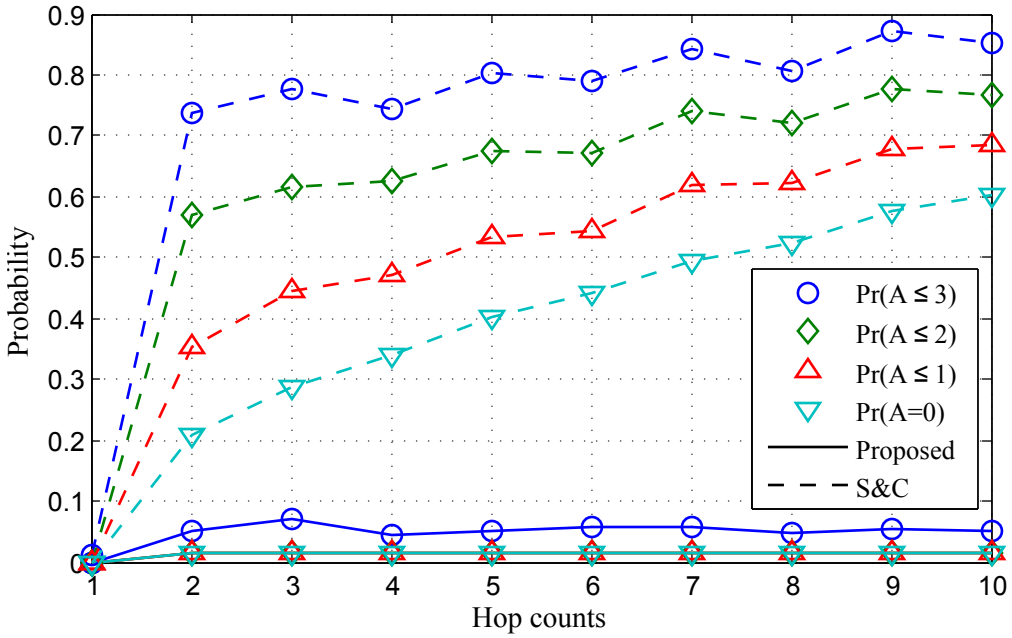


Figure 56: Outage probability of each hop with different outage criteria.

As shown in the result, the average number of the active relays in the proposed method is close to four over all hops. This implies that the CCT message initiated from the source can reach up to the last hop without loss. In contrast, the average number of the active relays in the S&C method degrades after the second hop. In the first hop, the message initiated from the source can be always decoded successfully in the first cluster. However, the cooperative

message from the first cluster can be decoded successfully by 50% of the relays in the second cluster. Since we keep the same transmit power, diversity and encoding scheme for the two methods, it is thought that the packet loss comes from the large synchronization error.

The performance of the multi-hop CCT also can be evaluated by the outage event. In this experiment, we define the outage of the multi-hop CCT as the number of active relays is less than or equal to the certain threshold $Pr\{A_n \leq O\}$ where O is the outage threshold. Figure 56 shows the outage probability in each hop with the different thresholds. As the result from Figure 55, the proposed method outperforms the S&C method for all thresholds. In addition, the outage performance of the propose method is consistent over hops for all thresholds. The experimental result in this figure shows that OFDM-based CCT with the proposed pre-synchronization is practical for the indoor environment.

6.8 Summary

In this chapter, a method for time and frequency pre-synchronization for OFDM-based DMIMO was proposed. The proposed method is designed to estimate the combined time and frequency offsets weighted by the channel gain to minimize the variance of the retransmission error. The performance of the proposed algorithm was evaluated through computer simulation, and demonstrated on as experimental testbed on GNU Radio and USRPs. The simulation and experimental results show that the OFDM-based DMIMO is practical, in that the transmit time and frequency offsets can be effectively suppressed by the proposed pre-synchronization method in a multi-hop CCT network.

CHAPTER 7

IMPLEMENTATION OF COOPERATIVE ANALOG-AND-DIGITAL PROTOCOL

7.1 Overview

Wireless sensor networks (WSNs) need synchronized clocks over the whole network to determine the duty cycle and schedule tasks. In addition, many applications of WSNs require accurate time synchronization (TS) at each node so that data measured in different areas of the network can be properly time-tagged and later processed (e.g., correlated) at a central location. Some of these applications such as object tracking [76] and global structural health monitoring [77], require TS functionality to extract meaningful information from collected data, and involve large coverage areas.

Although WSNs are often distributed over large areas, the sensor radios have only a short range. Multi-hop communication can reduce the cost of such a network by not requiring that every sensor be within one hop of the higher-functioning and more expensive sink nodes. Multi-hop communication presents additional challenges to the typical wireless communication protocol. For example, synchronizing the clocks on all nodes in the network, when there is no external reference, is more difficult than for a star topology network [78, 79]. While GPS can provide high-quality TS in some applications, the GPS signal is often not available at sensor locations, such as indoors, or costs too much energy.

In this chapter, inspired by the high quality of CCT time synchronization demonstrated in Chapter 5 and 6, and by the low and stable synchronization error statistics in cascaded (multi-hop) CCT, we propose a network-time synchronization protocol, called the Cooperative Analog and Digital (CANDI) TS protocol, which exploits analog and digital forms of CT at different stages of the synchronization process. CANDI promises significantly shorter protocol time and smaller time errors in multi-hop networks. The first sweep of the network uses CCT, wherein cooperating nodes simultaneously transmit the same digital

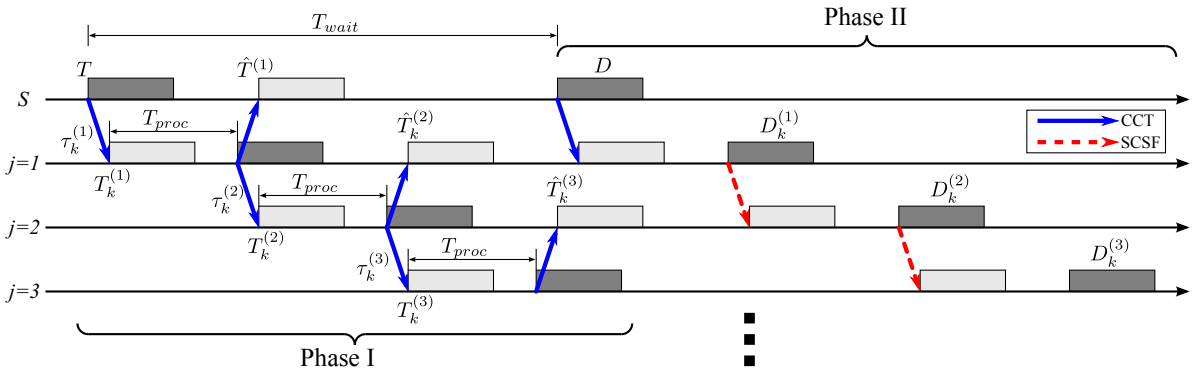


Figure 57: Illustration of CANDI network time synchronization protocol.

message in orthogonal channels that experience independent multi-path fading. Each receiver is capable of combining the differently faded copies, thereby achieving a significant SNR advantage, through array and diversity gains. In the analog stage, the cooperating nodes simultaneously transmit their individual estimates of the time, encoded across orthogonal dimensions. Nodes receiving this signal combat fading and reduce estimation error in one-step through the averaging inherent in diversity combining. The performance of the proposed protocol is compared to the performance of the Timing-sync Protocol for Sensor Networks (TPSN), which is a well-known time synchronization protocol for a sensor network.

This chapter is organized as follows. Section 7.2 describes the proposed protocol with the system topology. In Section 7.3, the implementation details to realize the proposed protocol are provided. The performance of the proposed protocol is compared to the Timing-sync Protocol for Sensor Networks (TPSN), which is a well-known time synchronization protocol for a sensor network in Section 7.4. This chapter is summarized in Section 7.5.

7.2 Protocol Description

In this section, we use a superscript and a subscript to express the cluster and node index, respectively. For example, $R_k^{(j)}$ is Node k in the j^{th} cluster. As an example of a two-subscripts case, $h_{21}^{(j)}$ is the channel between Node 1 in the $(j-1)^{th}$ cluster and Node 2 in the

j^{th} cluster. Similarly, $T_{21}^{(j)}$ is the time of that the head of a packet from Node 1 in the $(j-1)^{th}$ cluster reaches Node 2 in the j^{th} cluster. This time is referred to as the start-of-packet (SOP) time of the packet throughout this chapter. $\hat{T}_{21}^{(j)}$ represents the estimated value of the SOP time $T_{21}^{(j)}$.

We analyze the CANDI protocol using the multi-hop network topology. It is assumed that by operating OLA transmission in the first phase, $N^{(j)}$ nodes form a cluster in the j^{th} hop as shown in the figure. In Chapter 4 it was shown that by using the center-of-mass method for SOP time estimation, the weighted average of estimated SOP times by channel gain is an ABLUE, which effectively minimizes the estimated time errors across the nodes in the receiver clusters, if the propagation time differences between nodes in the cluster are negligible. Therefore at Node k in the j^{th} cluster the combining coefficient $a_{kn}^{(j)}$ for estimated SOP time $\hat{T}_{kn}^{(j)}$ becomes

$$a_{kn}^{(j)} = \frac{|h_{kn}^{(j)}|^2}{\sum_{i=1}^{N^{(j-1)}} |h_{ki}^{(j)}|^2}. \tag{60}$$

CANDI consists of two phases: 1) the root time broadcast and 2) the propagation time correction, as shown in Figure 57, in which the horizontal axis indicates time and the vertical axis indicates a node or cluster index. In the first phase, the source node (or the root) broadcasts its timestamp to the network using OLA broadcasting. In the meantime, the nodes in each cluster estimate and record a propagation time, e.g., $\hat{T}^{(1)}$, to the next cluster or node by overhearing the ongoing broadcast message. The source initiates the second phase after waiting some amount of time, T_{wait}, to avoid intra-flow interference [80] of the first phase broadcasting. The source broadcasts its estimation value of propagation delay to the first cluster's nodes. Once the nodes in the first cluster receive the source's estimated propagation delay information, they adjust their time based on the source timestamp and estimated propagation delay from the first and second phase respectively. The nodes in the first decoding level, then, rebroadcast their accumulated propagation delay to the next decoding level using Semi-Cooperative Spectrum Fusion (SCSF) [81], which is an analog cooperative transmission method. Next, we explain the phases in more detail.

7.2.1 Phase I: Root time broadcast

The source initiates the OLA broadcast with its timestamp (root time) embedded into the broadcast message at time T as shown in Figure 57. The nodes in the first cluster estimate the SOP time of the source's packet. In this case, the transmitted signal is a transmission by just one radio, so there is no combining. The nodes in the first cluster decode the source packet and reset their local clock to match the SOP time to the embedded source's timestamp. Now the nodes schedule their rebroadcasting by adding a fixed processing time T_{proc} to the estimated SOP-time $\hat{T}_k^{(1)}$. The time-to-live (TTL) field in the rebroadcasting message should be decremented by one. The rebroadcasting of Node k in the first cluster occurs at $\hat{T}_k^{(1)} + T_{proc}$. The source node overhears and estimates SOP time of this rebroadcasting message and calculates the propagation time from the source to the first decoding level by $\tilde{\tau}^{(1)} = (\tilde{T}^{(1)} - T - T_{proc})/2$. The source node records its estimated propagation delay for the second phase.

Node k in the j^{th} cluster receives $N^{(j-1)}$ copies of the source message from the previous cluster. The node estimates SOP times $\hat{T}_{kn}^{(j)}$ where $1 \leq n \leq N^{(j-1)}$, effectively using (60) to produce a combined SOP-time. All nodes in the j^{th} cluster also rebroadcast the source message with decremented TTL using CCT. The following re-broadcasted messages from the $(j+1)^{th}$ cluster will be overheard by the nodes in the j^{th} cluster. For calculating the propagation delay, they also combine multiple SOP-times using (60) to determine a combined SOP-time. In this case, the estimated propagation delay at Node k in the j^{th} cluster becomes a weighted average of multiple propagation delays because of the combining method in (60). The broadcasting of the source's timestamp will continue until the TTL field in the broadcasting packet becomes zero.

7.2.2 Phase II: Propagation time correction

The source starts the second phase at $T + T_{wait}$, where T_{wait} is a pre-determined value, which should be at least $3 \cdot (T_{proc} + max(\tau))$ to avoid the intra-flow interference [82]. The source node digitally encodes its estimated propagation delay $\hat{\tau}^{(1)}$ into the second phase's message.

The nodes in the first cluster receive the source message and adjust their clock offsets by adding $\hat{\tau}^{(1)}$, which is decoded from the message.

Each node in the first cluster encodes the accumulated propagation delay, $D_k^{(1)} = \hat{\tau}^{(1)} + \hat{\tau}_k^{(2)}$, where $\hat{\tau}^{(1)}$ is the received propagation delay information from the source and $\hat{\tau}_k^{(2)}$ is its own estimated propagation delay by overhearing the next cluster's rebroadcast messages. Because each node has a slightly different message to transmit, that is, each node has a slightly different notion of accumulated two-hop propagation delay, CCT cannot be used. Therefore, we use SCSF. In our demonstration, which will be described in Section 7.4, we use frequency to create orthogonality in SCSF. The k^{th} node encodes its $D_k^{(1)}$ using linear frequency modulation (FM), i.e., the k^{th} node transmits a single tone that is offset from the estimate of the received carrier frequency according to an offset $f_k^{(1)} = f_o + \alpha D_k^{(1)}$. Each node in the receiving cluster, which is Cluster 2 ($j = 2$), estimates the received spectrum, and uses the center-of-mass of that spectral estimate as the estimate of $f_k^{(1)}$. In general, Node k in the j^{th} cluster can decode the SCSF message from the $(j-1)^{th}$ cluster as

$$\tilde{D}_k^{(j-1)} = \frac{1}{\alpha}\left(\frac{\sum_{n=1}^{N_{win}} v_n |\mathcal{F}_k(n)|^2}{\sum_{n=1}^{N_{win}} |\mathcal{F}_k(n)|^2}\right), \tag{61}$$

where N_{win} is the DFT size, v_n is the n^{th} DFT frequency, $\mathcal{F}_k(n)$ is n^{th} DFT value of the received baseband SCSF signal [81]. The node adjusts its local clock by adding the decoded accumulated propagation delay $\tilde{D}_k^{(j-1)}$. The node next retransmits the re-encoded SCSF message, $D_k^{(j)} = \tilde{D}_k^{(j-1)} + \hat{\tau}_k^{(j)}$, to the next cluster.

In the second phase, the nodes in each cluster stop their synchronization operation right after transmitting an accumulated propagation delay to the next cluster. Therefore, overhearing operation does not necessarily occur in the second phase. The transmission will also continue until the embedded TTL field becomes zero.

7.3 Implementation Details

We implement our proposed algorithm using USRP1 and GNU Radio SDR. For the first phase of CANDI, we used the same physical layer and the protocol stack, as that discussed in Chapter 3.

7.3.1 SCSF Implementation

Unlike the SCSF application in [81], which is aerial reading of a sensor field in a LOS channel, we consider a fading channel environment. Also, different from [81], we consider imperfection of frequency and timing recovery in the SCSF decoding process. Each transmitter converts its desired data into frequency using a modulation index. The receiver first detects the start of SCSF-encoded sub-frame using BFSK-encoded preamble. The sub-frame passes through the DFT and mag-square blocks for non-coherent decoding, in which the mag-square output of DFT is the periodogram. The SCSF receiver finally calculates the center-of-mass of the periodogram to estimate the average of transmitted data. The SCSF algorithms can be realized as follows.

7.3.2 Modulation of Data

Node k converts its desired data s_k scaled by the modulation index ζ into frequency. The modulation index is chosen in such that $|s_k \cdot \zeta| < 1/N_{win}$ for all s_k. Discrete baseband representation of the transmitted SCSF signal from node k can be written as

$$x_k[n] = e^{j2\pi s_k \zeta n}, \tag{62}$$

where $n = 0, \ldots, N_{win} - 1$.

7.3.3 Imperfection of Carrier Frequency Recovery

Since SCSF modulates its data into frequency, the discrepancy of the receiver local clock oscillator, which provides a reference clock to the carrier frequency mixer, causes a misreading of SCSF encoded data. At the receiver, the superimposed signal from N transmitters with carrier frequency offset $\triangle_k$ can be written as

$$\begin{aligned} y[n] &= \sum_{k=1}^{N} h_k \cdot e^{j2\pi s_n \zeta n} \cdot e^{j2\pi \frac{\triangle_n}{f_s} n} + \omega[n] \\ &= \sum_{k=1}^{N} h_k \cdot e^{j2\pi (s_n \zeta + \frac{\triangle_k}{f_s}) n} + \omega[n], \end{aligned} \tag{63}$$

where f_s is the sampling frequency. The DFT of $y[k]$ can be calculated by

$$\begin{aligned} \mathcal{F}[m] &= \sum_{l=0}^{N_{win}-1} \left\{ \left(\sum_{k=1}^{N} h_k e^{j2\pi (s_n \zeta + \frac{\triangle_k}{f_s}) l} \right) + \omega[l] \right\} e^{-j2\pi \frac{m}{M} l} \\ &= \sum_{l=0}^{N_{win}-1} \left(\sum_{k=1}^{N} h_k e^{j2\pi (s_n \zeta - \frac{k}{M} + \frac{\triangle_n}{f_s}) l} \right) + \Omega[m], \end{aligned} \tag{64}$$

where $\Omega[m] = \sum_{l=0}^{N_{win}-1} \omega[l] e^{-j2\pi \frac{m}{M} l}$. If we assume the unit channels $h_k = 1$ for all k and the noise free channel, the estimated SCSF value of (64) will indicate the average value of the net or superimposed data, which is the sum of designated data and frequency offset normalized by $f_s \zeta$,

$$\tilde{D}_{net} \approx \frac{1}{N} \sum_{k=1}^{N} \left(s_k + \frac{\triangle_k}{f_s \zeta} \right). \tag{65}$$

If we set the transmit data s_k to zero for all index k, only the frequency offset term in (65) is left. Therefore, by transmitting the reference SCSF frame that encodes zero as a data along with the actual SCSF frame, the SCSF-encoded data with a frequency offset can be estimated by simply taking the subtraction the reference from the net as

$$\tilde{D} = \tilde{D}_{net} - \tilde{D}_{ref}. \tag{66}$$

Fig. 58 shows the SCSF encoding/decoding process for implementation. Each transmitter converts its desired data into frequency using a modulation index. It has to be within a DFT bandwidth, which can be guaranteed through the limiter in the figure. The receiver

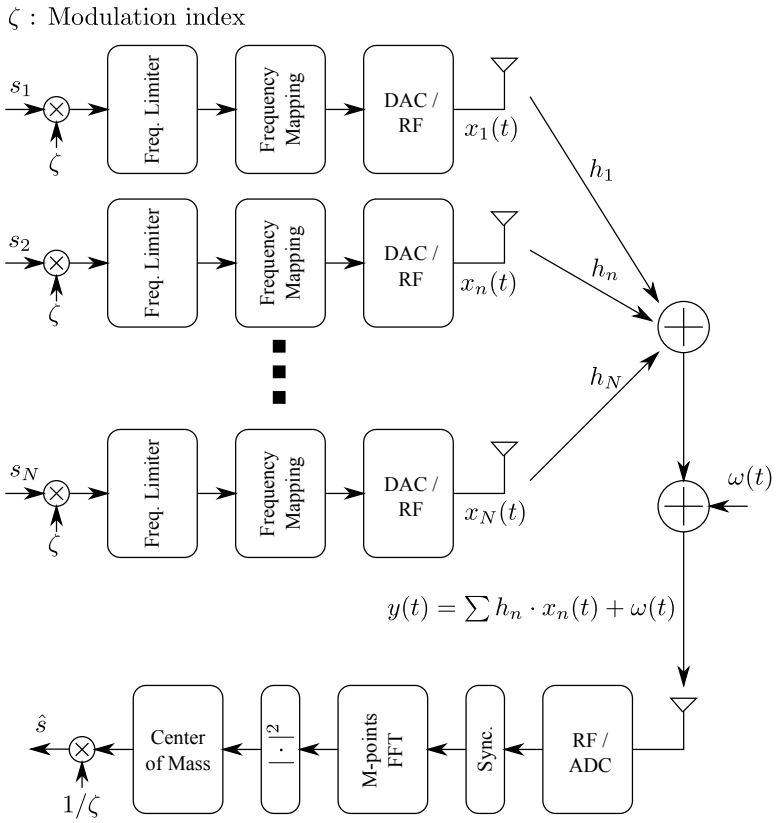


Figure 58: GNU Radio block diagram for SCSF implementation.

first detects the start of SCSF-encoded sub-frame using BFSK-encoded preamble. The sub-frame passes through the DFT and mag-square blocks for non-coherent decoding, in which the mag-square output of the DFT is called periodogram. The SCSF receiver finally calculates the center-of-mass of the periodogram to estimate the average of transmitted data.

Fig. 59 shows the frame structures for the first and second phase of CANDI. The frame structure in the first phase is a typical frame structure for a digital communication with a preamble. In contrast, the second phase of CANDI has two sets of SCSF sub-frames, in which the first sub-frame is the reference SCSF sub-frame to send a zero-encoded data while the second sub-frame sends the actual data (i.e., $D_k^{(i)}$).

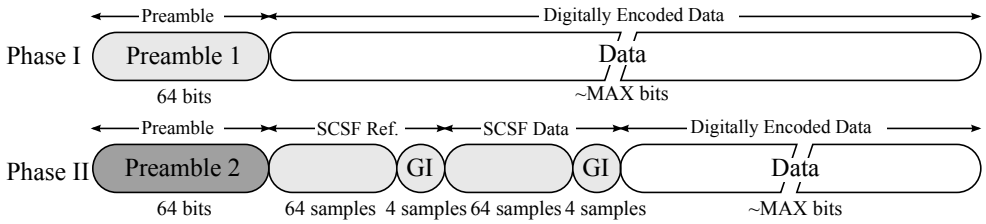


Figure 59: Frame structure for CANDI implementation.

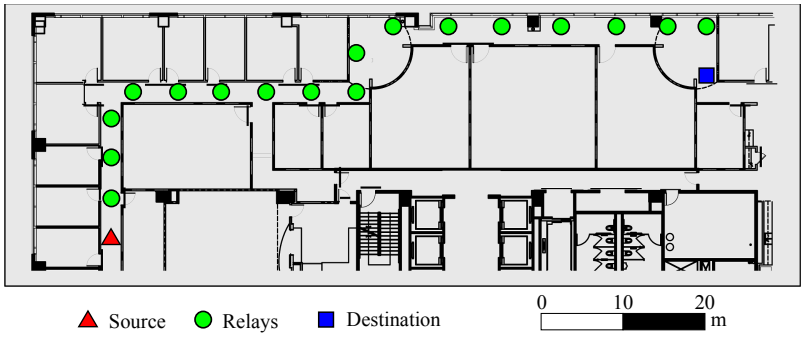
7.3.4 Imperfection of Transmit Time

Not only frequency offsets between nodes but also transmit timing errors can affect the decoding process of SCSF. At the receiver side, a number of SCSF frames are superimposed, in which each frame has a slightly different timing. We insert a guard interval into the end of each SCSF sub-frame, like the OFDM guard interval, to compensate for the multipath delays and transmit time offsets. The content of the guard interval is a copy of the first part of the SCSF sub-frame. Once the receiver determines the starting point of the SCSF sub-frame based on the preamble detection, N_{win} samples can be taken with $|GI|/2$ offset where $|GI|$ is the length of the guard interval.

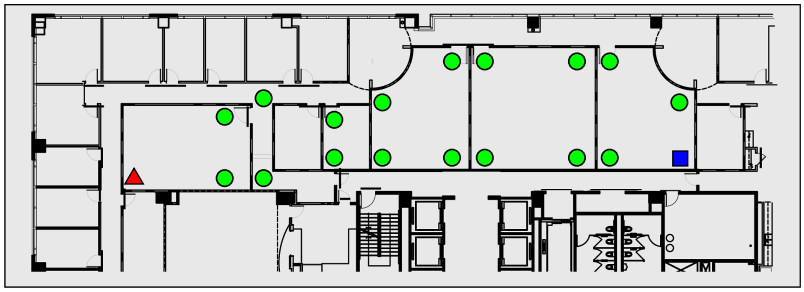
7.4 Experimental Study of Cooperative Analog-and-Digital Protocol

The experiment in this section is conducted using 19 nodes, which were deployed on the fifth floor of the Centergy building of the Georgia Institute of Technology, as shown in Figure 60. This building consists of offices and research laboratories. We conduct multi-hop experiments with two different topologies: equidistant linear network along the hallways and group deployment in the office and laboratory rooms, which correspond to Figure 60a and 60b, respectively. The linear network topology along the corridors has relatively more favorable propagation, while the second topology, where the nodes are placed in the rooms, experiences severe attenuation by walls in direct paths. In other words, the first topology has a strong LOS component, while the second topology undergoes more severe multi-path propagation than the first topology.

In case of a global synchronization for a large-scale multi-hop network, it is likely that



(a) Topology 1



(b) Topology 2

Figure 60: Floor plan and node placement for experiments.

the timing error of a certain TS protocol increases as the hop count (or the latency) from the root node increases. Even though TPSN shows a quite stable average synchronization error in multi-hop experiments that have up to five hops [83], for larger networks, the node with the largest hop distance would suffer from the biggest error by the clock drift that is not compensated in TPSN. Therefore, the error between the root and the final node with largest hop distance can demonstrate the performance and suitability in a large-scale network, which is the case of interest in this chapter.

However, it is practically difficult to measure the timing error of the two nodes physically far apart, accurately. Therefore, we design the measurements in a round trip manner. Suppose that the red triangle and the blue square of each topology in Figure 61 are the source and the destination, respectively, and the green-colored circles in Figure 61 indicate the relay nodes, which connect the both ends by multi-hop communications. Although a network time protocol does not necessarily need a destination node, the role of it in these experiments is to re-initiate the protocol once the node is synchronized.

To be specific, the source and destination nodes are the two ends of the two-way synchronization, which consists of forward- and reverse-path synchronizations. For the forward path, the source node indicated by the red triangle serves as the root node (the time information source), and the other nodes synchronize to the source node. If the destination node is successfully synchronized to the source node after the forward path synchronization, we start the reverse path synchronization by immediately re-initialization of the network-time protocol algorithm using the destination as the root node. In other words, for the reverse path, the other nodes including the source synchronize to the destination. When the source node is also successfully synchronized to the destination after the reverse path synchronization, the initial source node has two time clocks: its original clock and the time clock achieved by the two-way synchronization. In the ideal case, the two time clocks would be the same, but in practice there is a time gap between the two because of the errors occurred during time synchronization process. We use this time gap (timing error) between the two as our performance metric, which is also easy to measure because we extract the two timing information in the same node (the source) without introducing additional errors or distortions in the measurement. We note that the forward and reverse paths for the two-way synchronization through multi-hop SISO communication in TPSN are predefined to have the shortest paths (i.e., smallest hop counts) without any collision.

For orthogonal sub-channel assignment in a CANDI protocol, we manually assign one of the four diversity channels to each node in incremental fashion (i.e., {1,2,3,4,1,2,3,...} from end to end). Even though manual assignment can be replaced by a distributed algorithm, the design of such algorithm is beyond the scope of this dissertation. When multiple nodes assigned to the same sub-channel form the same cooperation set and the instantaneous received powers from the nodes at a receiver are comparable to each other, the receiver node will experience self-fading within the sub-channel. However, frequency diversity and path-loss disparity of each node might mitigate the performance degradation caused by self-fading.

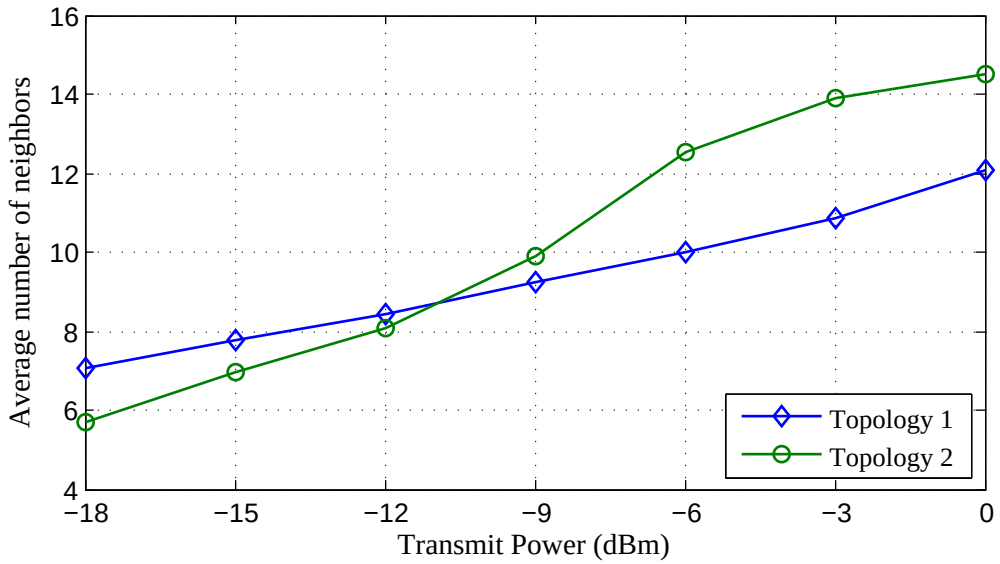


Figure 61: Average number of neighbors in the two topologies.

We consider different transmit power levels by increasing from -18dBm to 0dBm with 3dBm intervals (total of 7 different power levels). Higher transmit power increases the node degree, which means the average number of one hop neighbors as defined in [39], of the network as shown in Figure 61, thereby decreasing hop count from the source to the destination. In the figure, the blue curve with the '⋄'-markers indicates the node degree of Topology 1, while the green curve with 'o'-markers represents the node degree of Topology 2. As shown in the figure, the node degrees of Topology 2 are lower than Topology 1 at the lower transmit powers. However, Topology 2 shows the higher slope compared to Topology 1, which gives greater node degrees than Topology 1 with the high transmit powers.

Figure 62 shows the RMS synchronization error after the two-way sweep, which is averaged over 200 trials. In the figure, and in the remainder of this chapter, the blue curve with the '⋄'-markers and the green curve with the 'o'-markers represent TPSN and CANDI, respectively. The 'I-shaped' bars indicate the 95% confidence intervals of the statistical data. If we look at the trend of the two curves under the change in transmit power, we can observe that the timing errors of the both protocols decrease as the transmit power increases. We consider two possible reasons for this: the SOP-time estimation error and

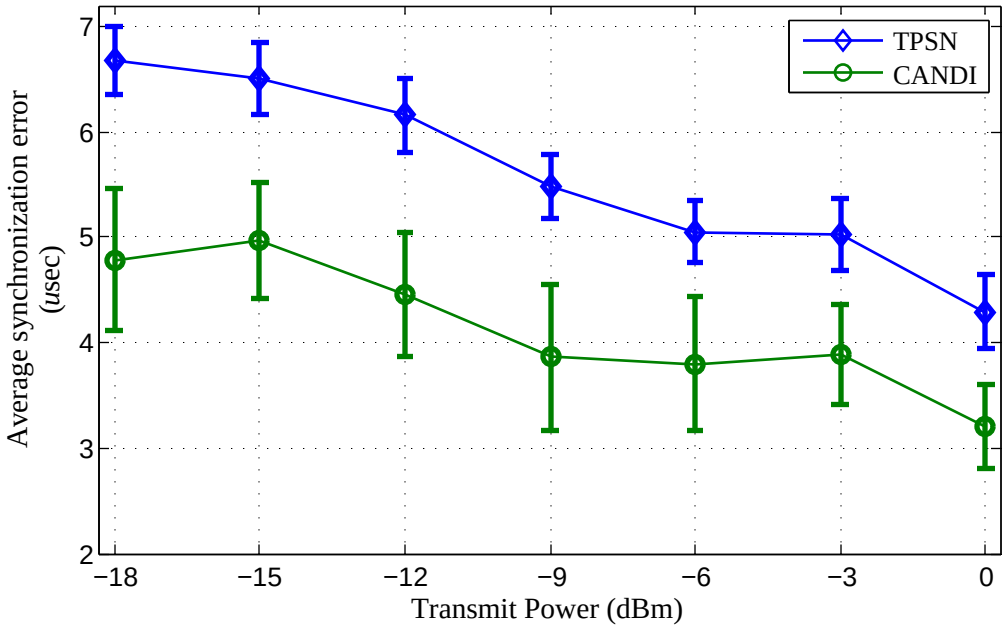


Figure 62: RMS timing error between extreme nodes in Topology 1.

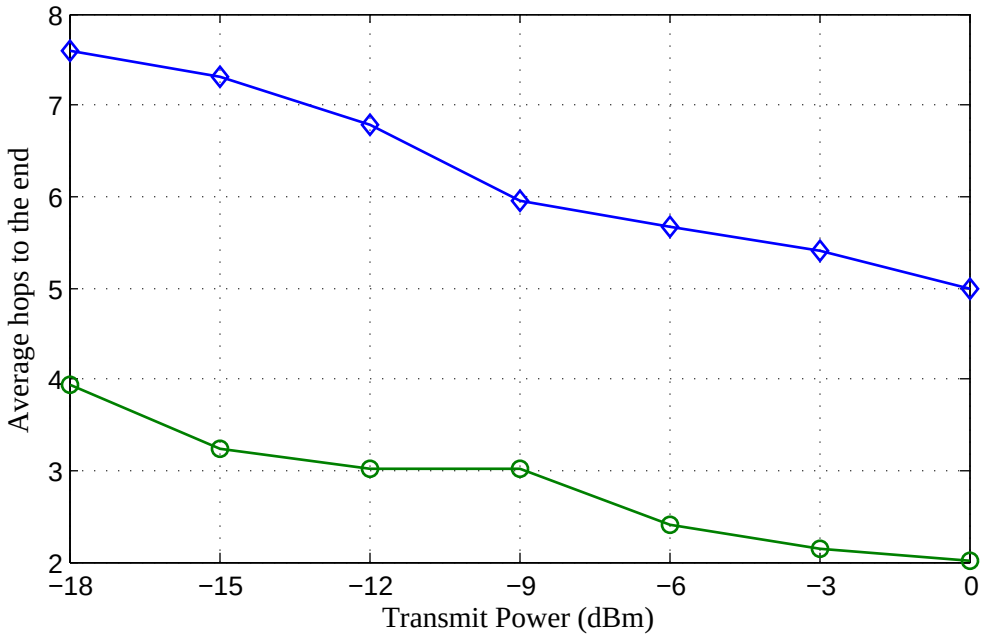


Figure 63: Average hop count in Topology 1.

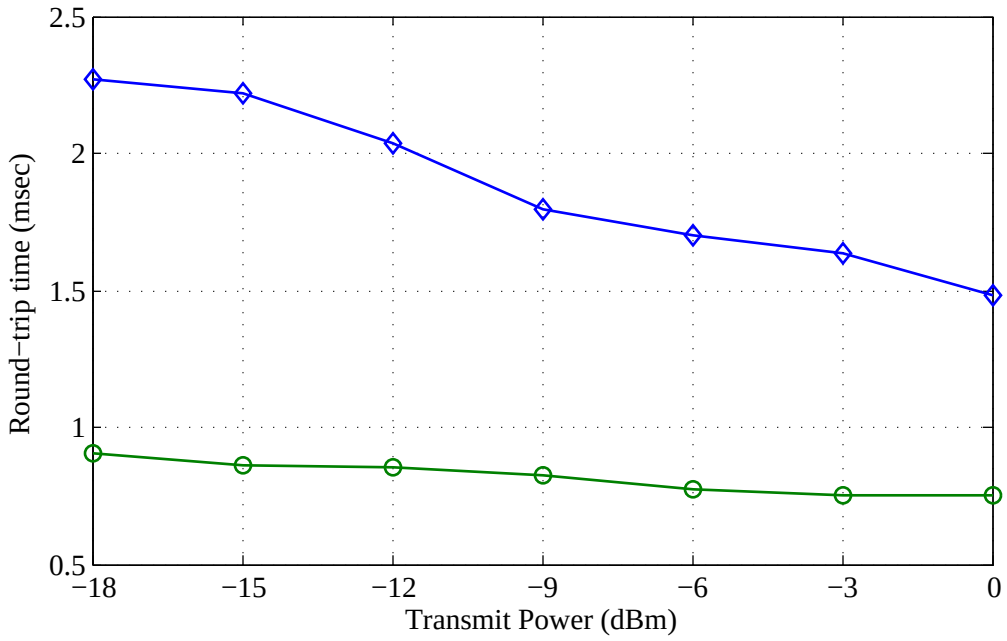


Figure 64: Average round trip time in Topology 1.

the clock drift error. First, the SOP-time error is inversely proportional to the received SNR, so the higher transmit power reduces the SOP-time errors for the both algorithms. Second, as transmit power increases, the hop count from one end to the other end decreases as shown in Figure 63. Also, this hop count saving by the increased transmit power level cuts down the overall latency to finish the two-way sweep of the two protocols. Because the clock drift error is proportional to the latency, TPSN and CANDI will be less distorted by the clock drift as the transmit power increases.

A more important observation is that CANDI outperforms TPSN across all transmit power levels; specifically (as shown in Figure 62), CANDI provides reductions in the RMS timing error, relative to TPSN, of 22.6% at -3dBm up to 29.5% at -18dBm, which can be explained as follows. First, CANDI achieves SNR advantage though array and diversity gains of CCT in the SOP detection, which makes the SOP estimation error smaller compared to TPSN based on the conventional SISO links. Second, the range extension property of CT allows hop count saving of CANDI as shown in Figure 63 over the all transmit power ranges, which makes CANDI take less time to finish the two-way sweep as in Figure 64.

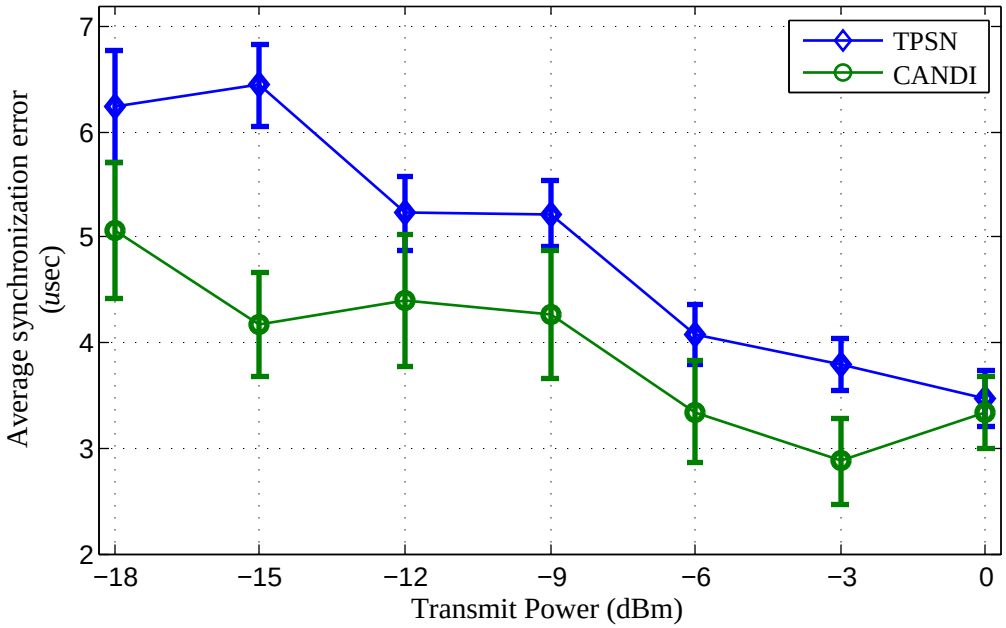


Figure 65: RMS timing error between extreme nodes in Topology 2.

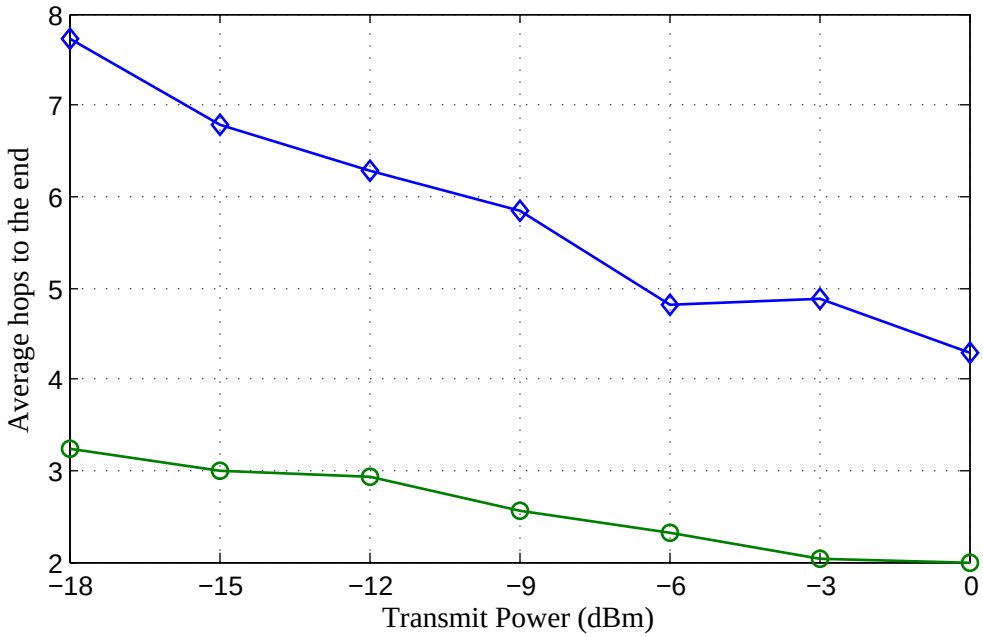


Figure 66: Average hop count in Topology 2.

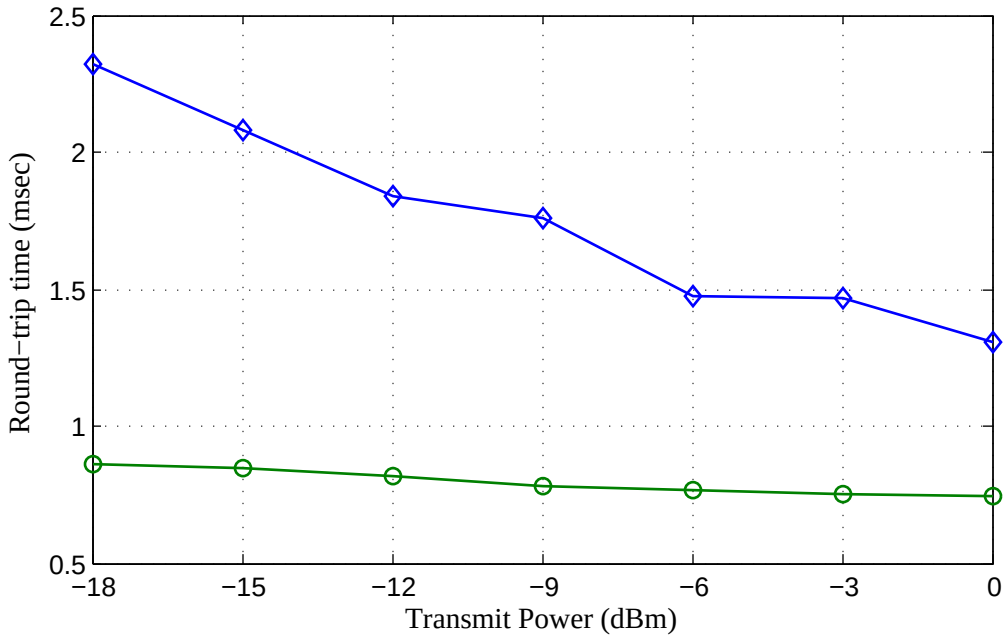
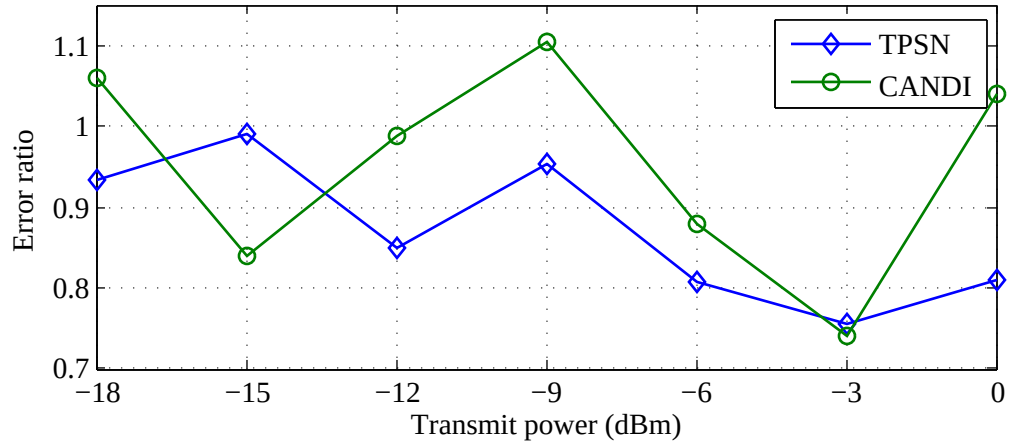


Figure 67: Average round trip time in Topology 2.

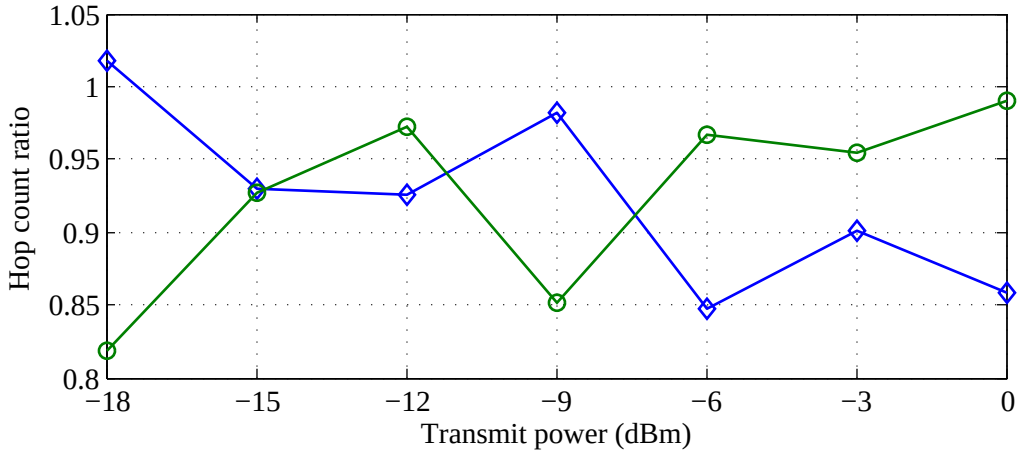
Therefore, compared to TPSN, CANDI suffers less from the clock drift.

Figure 65 shows the RMS timing error of the second (grouped) topology. We observe CANDI still outperforms TPSN with reductions in the rms timing error up to 35.2% at -15dBm. However, the performance gap between CANDI and TPSN becomes smaller especially with high transmit powers (e.g., 3.7% rms timing error reduction at 0dBm) compared to the results with the first topology shown in Figure 62. That is because TPSN benefits more from the shorter end-to-end distance in Topology 2 compared to Topology 1. In other words, as shown in Figure 66 and 67, while CANDI has almost the same end-to-end hop counts and round trip times in the both topologies by CT, TPSN based SISO links can reduce the end-to-end delay significantly in Topology 2 compared to the linear network in Topology 1. Also, the significant hop-count saving of TPSN in Topology 2 can be expected in the higher slope of the node degree curve of Topology 2 compared to Topology 1 in Figure 61.

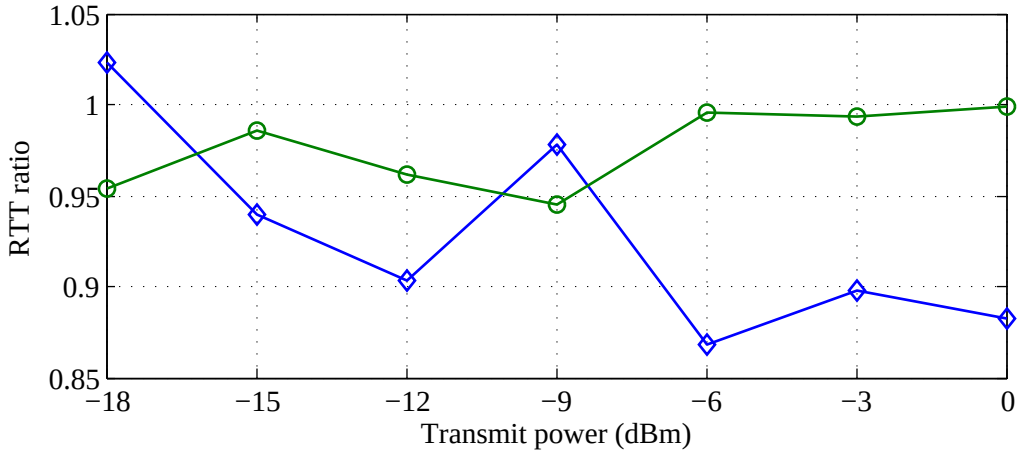
For further analysis on the experimental results in the two topologies, Figure 68 shows the ratios of the experimental results in Topology 2 to Topology 1, where the graphs in



(a) RMS timing errors



(b) Average hop counts



(c) Average round trip times

Figure 68: Measurement result ratios of Topologies 2 to 1.

Figs 68a, (b), and (c) correspond to the ratios of the rms timing errors, average end-to-end hop counts, and average round trip times, respectively. As shown in Fig 68a, with the high transmit powers (e.g., -6, -3 and 0dBm), the decrease of the timing error in TPSN from Topologies 1 to 2 is more significant, because the blue TPSN curve is much smaller than the green CANDI curve. Moreover, as shown in Figs. 68b and (c), the hop count saving and the corresponding round trip time reduction, with increasing transmit power, are more significant in TPSN than CANDI.

We note that both topologies in the experiment cause path-loss disparity, where the received powers from one or two closet nodes dominate, while the received powers from the other cooperative transmitter are negligible. To be specific, the linear network in Topology 1 inherently has dissimilar path losses by its equally separated node placement. Similarly, the nodes in Topology 2 are widely dispersed at the corners of the rooms, which make significant path-loss disparity. Because CT used in CANDI suffers from the SNR penalty in presence of this path-loss disparity as shown in [84], we expect that CANDI can show even better performance with higher node degree (density),which provides enough number of cooperators. Moreover, considering the exponential CT range extension in the high-density network as shown in [85], the performance gap between CANDI and TPSN should become greater, as the network size increases.

7.5 Summary

In this chapter, two forms of CT, digital-based CCT and analog-based SCSF, are combined to create a new distributed method of network time synchronization protocol to achieve fast and accurate time synchronization over large multi-hop WSNs. The experimental results show that the averaging and range extension benefits of CCT lead to reductions in worst-case RMS timing errors of up to 29.5%.

CHAPTER 8

CONCLUDING REMARKS AND FUTURE RESEARCH

This dissertation addresses the practical design and implementation of concurrent cooperative transmission (CCT) to form a distributed multi-input and multi-output (DMIMO) link in the context of multi-hop wireless networks. The novel pre-synchronization method enables, to the best of our knowledge, for the first time, a cluster of single-antenna radios, to transmit as a virtual array, in response to a packet commonly received from another virtual array. The method is shown to produce transmit time and frequency spreads that are less than or comparable to typical multipath delay and Doppler spreads. The method is realized for both narrowband non-coherent binary frequency-shift keying (NCBFSK) and wideband orthogonal frequency division multiplexing (OFDM). In both cases, the novelties include frame design and receiver signal processing algorithms. Such DMIMO techniques can be used in various ways, including to extend range to broadcast more quickly, as a basis for a more accurate network time synchronization protocol, or to overcome a network partition. They can also be used in cooperative routing to increase reliability through diversity or throughput via spatial multiplexing.

Another contribution is a testbed, including a flexible protocol stack, for a network of software-defined radios (SDRs). Since CCT is not supported by any existing standard or off-the-shelf radios, the advantage of CCT usually has been proven theoretically in the literature or demonstrated by using SDRs. However, the existing SDR testbeds for CCT fall short of realizing the self-contained and real-time SDR implementation. In addition, the lack of support for the multiple layers in the existing testbed hinders the testing of various CCT-based protocols in real environments. The SDR testbed designed in this dissertation enables the CCT-based protocols that had previously been studied only theoretically or through simulation, to be realized and evaluated over the air. The designed SDR testbed for CCT is fully self-contained and real-time. The designed SDR testbed has also been used to

experimentally demonstrate and study certain CCT-based broadcasting and routing protocols, which are not reported in this dissertation. Because the testbed has a systematic and flexible framework, it can be further utilized as a powerful physical layer tool to evaluate and demonstrate future CCT-based protocols.

The dissertation also addresses consecutive DMIMO hops, such as might be needed in a cooperative route. The synchronization center-of-mass method was proved and experimentally demonstrated to be statistically stable with hop count. The statistical stability of transmit time and frequency error was demonstrated through the "ping-pong" experiments in the aforementioned chapters. Taking advantage of the simple broadcasting nature of CCT in a multi-hop network, we designed and demonstrated the CCT-based network-time synchronization protocol, called the Cooperative Analog and Digital (CANDI), which promises significantly shorter protocol time and smaller time errors in multi-hop networks.

Suggestions for future work include implementation and modeling for very long-range and/or very high date-rate DMIMO links with very large numbers of nodes at each end. One issue with the long-range link is that each diversity channel will be shared by many nodes, and will cause self-fading. Self-fading effects are expected to depend on the number of co-diversity-channel transmitting nodes. Another interesting investigation would be to compare, for very long ranges, CCT to time-division CT. Time-division CT would require more preamble energy from each transmitter, to ensure successful synchronization in the receiver.

APPENDIX A

COVARIANCE OF THE MARKOV PROCESS MODEL

From Equation 23, $C_d^{(j)}$ and $C_o^{(j)}$ can be written as

$$C_d^{(j)} = C_d^{(j-1)} \sum_{n=1}^{N} E\{(a_{gn}^{(j)})^2\} + C_o^{(j-1)} \sum_{\substack{x=1 \\ x \neq y}}^{N} \sum_{y=1}^{N} E\{a_{gx}^{(j)} a_{gy}^{(j)}\}$$
$$+ \sum_{n=1}^{N} E\{a_{gn}^{(j)} \omega_{gn}^{(j)}\} + \sum_{n=1}^{N} E\{\xi_n^{(j)}\}$$

and

$$C_o^{(j)} = C_d^{(j-1)} \sum_{n=1}^{N} (E\{a_{gn}^{(j)}\})^2 + C_o^{(j-1)} \sum_{\substack{x=1 \\ x \neq y}}^{N} \sum_{y=1}^{N} E\{a_{gx}^{(j)}\} E\{a_{gy}^{(j)}\}$$

respectively. From [86], the expectation of the combining coefficient $a_{gn}^{(j)}$ does not depend on any indexes and its expectation value is given by

$$E\{a_{gn}^{(j)}\} = 1/N. \tag{67}$$

Assuming that $X = |h_{gn}^{(j)}|^2$ and $Y = \sum_{k=1}^{N} |h_{gk}^{(j)}|^2$, the expectation can be written as

$$E\{(a_{gn}^{(j)})^2\} = \frac{E\{X^2\}}{E\{Y^2\}}. \tag{68}$$

Since X and Y have exponential and gamma distribution respectively, $E\{a_{gn}^{(j)}\}$ can be simplified to $\frac{2}{N(N+1)}$. Also we can get

$$\sum_{\substack{x=1 \\ x \neq y}}^{N} \sum_{y=1}^{N} E\{a_{gx}^{(j)} a_{gy}^{(j)}\} = E\{(\sum_{n=1}^{N} a_{gn}^{(j)})^2\} - E\{\sum_{n=1}^{N} (a_{gn}^{(j)})^2\}$$
$$= 1 - E\{\sum_{n=1}^{N} (a_{gn}^{(j)})^2\} = \frac{N-1}{N+1}$$

It is noted that the third and fourth term of $C_d^{(j)}$ do not depend on the index j because we assume that the statistics of all channels, clocks and noises do not vary.

APPENDIX B

ERROR VARIANCE OF CENTER-OF-MASS ESTIMATION

Assuming a single-bit preamble $P = 1$, at any index n, $\Lambda(n)$ can be expressed as

$$\Lambda(n) = \left|d(n)\right|^2 = \left|h \cdot q(n) + w(n)\right|^2$$
$$= \alpha_R^2(n) + \alpha_I^2(n)$$

where

$$\alpha_R(n) = Re\{h \cdot q(n) + w(n)\} \tag{69}$$

and

$$\alpha_I(n) = Im\{h \cdot q(n) + w(n)\} \tag{70}$$

with means μ_{α_R} and μ_{α_I}, and variances $\sigma^2_{\alpha_R}$ and $\sigma^2_{\alpha_I}$. $\alpha_R(n)$ and $\alpha_I(n)$ are non-zero mean Gaussian random variables with variance $N_0/2$.

The sum of squared Gaussian random variable has a non-central chi-square distribution. Let $(X_1, X_2, \ldots, X_k)$ denote k independent, Gaussian random variables with μ_i and variances σ_i^2. Then the random variable

$$\sum_{i=1}^{k} = \left(\frac{X_i}{\sigma_i}\right)^2 \tag{71}$$

has non-central chi-square distribution where k specifies the number of degrees of freedom. The mean and variance of the non-central chi-square distribution are $k + \lambda$ and $2(k + 2\lambda)$ respectively with the non-centrality parameter λ that is defined as

$$\lambda = \sum_{i=1}^{k} \left(\frac{\mu_i}{\sigma_i}\right)^2. \tag{72}$$

Let us define the variable β as

$$\beta = \left(\frac{\alpha_R(n)}{\sigma_{\alpha_R}}\right)^2 + \left(\frac{\alpha_I(n)}{\sigma_{\alpha_I}}\right)^2. \tag{73}$$

The mean and variance of β are given by

$$\mu_\beta = 2 + \left(\frac{\mu_{\alpha_R}}{\sigma_{\alpha_R}}\right)^2 + \left(\frac{\mu_{\alpha_I}}{\sigma_{\alpha_I}}\right)^2 = 2 + \frac{2}{N_0} \cdot \left|h\right|^2 \cdot \left|q(n)\right|^2$$
$$\sigma_\beta^2 = 2 \cdot \left(2 + 2 \cdot \left(\frac{\mu_{\alpha_R}}{\sigma_{\alpha_R}}\right)^2 + 2 \cdot \left(\frac{\mu_{\alpha_I}}{\sigma_{\alpha_I}}\right)^2\right) = 4 + \frac{8}{N_0} \cdot \left|h\right|^2 \cdot \left|q(n)\right|^2.$$

From the definition of $\Lambda(n) = \frac{N_0}{2} \cdot \beta$, the mean and variance of $\Lambda(n)$ at any index n can be expressed as

$$\mu_{\Lambda(n)} = N_0 + \left|h \cdot q(n)\right|^2$$

and

$$\sigma^2_{\Lambda(n)} = N_0^2 + 2N_0\left|h \cdot q(n)\right|^2.$$

Similarly, the mean and variance of $n \cdot \Lambda(n)$ are

$$\mu_{n\cdot\Lambda(n)} = n \cdot (N_0 + |h \cdot q(n)|^2)$$

and

$$\sigma^2_{n\cdot\Lambda(n)} = n^2 \cdot (N_0^2 + 2N_0|h \cdot q(n)|^2).$$

By the central limit theorem, the probability distribution of $\sum n \cdot \Lambda(n)$ and $\sum \Lambda(n)$ can be approximately Gaussian distribution. Let X and Y be denoted as $\sum n \cdot \Lambda(n)$ and $\sum \Lambda(n)$ respectively and define $Z = X/Y$. By Geary-Hinkley transformation [87], a transformation of the ratio distribution $Z = X/Y$ has been suggested so that the transformed variable T would have an approximately standard Gaussian distribution as

$$t = \frac{\mu_y z - \mu_x}{\sqrt{\sigma_y^2 z^2 - 2\rho\sigma_x\sigma_y z + \sigma_x^2}} \tag{74}$$

where ρ is correlation between X and Y. It is known that the approximation is good unless Y is negative. Since $\sum \Lambda(n) \geq 0$ for all n,

From (29), $w(n)$ is a filtered white Gaussian noise so called a *colored* Gaussian noise . Its auto-covariance is

$$\mathrm{Cov}(w(i), w(j)) = N_0 \cdot \sum_l s^*(l)s(i - j + l).$$

The mean and variance of X and Y are

$$\mu_x = 0,$$
$$\mu_y = \sum_i \mu_{\Lambda(i)},$$
$$\sigma_x^2 = \sum_i \sum_j \left(\sigma_{i\cdot\Lambda(i)} \cdot \sigma_{i\cdot\Lambda(j)} \cdot \left(\sum_l s^*(l) s(i-j+l) \right)^2 \right),$$
$$\sigma_y^2 = \sum_i \sum_j \left(\sigma_{\Lambda(i)} \cdot \sigma_{\Lambda(j)} \cdot \left(\sum_l s^*(l) s(i-j+l) \right)^2 \right).$$

the random variable ω can be transformed to

$$\omega = \frac{T_s \sqrt{KN_0}}{|h|} u,$$

where

$$K = \frac{2 \cdot \sum_i \sum_j \left(i \cdot j \cdot \left| q(i) \right| \cdot \left| q(j) \right| \cdot \left(\sum_l s^*(l) s(i-j+l) \right)^2 \right)}{\left(\sum_i q^2(i) \right)^2},$$

and u is a zero-mean unit variance Gaussian random variable. Also the variance of the estimation error ω can be approximated by

$$\sigma_\omega^2 = \frac{N_0 K T s^2}{|h|^2}. \tag{75}$$

APPENDIX C

ESTIMATION OF COMBINED FRACTIONAL-CFO

To simplify the problem, we assume that $K = 2$. From the trigonometry identity,

$$A_1 e^{j\omega_1} + A_2 e^{j\omega_1} = A_3 e^{j\omega_3} \tag{76}$$

where

$$A_3^2 = A_1^2 + A_2^2 + 2A_1A_2\cos(\Delta\omega),$$
$$\omega_3 = \arctan\left(\frac{A_1 sin(\omega 1) + A_2 sin(\omega 2)}{A_1 cos(\omega 1) + A_2 cos(\omega 2)}\right)$$

where $\Delta\omega = \omega_1 - \omega_2$. Since we assume the difference of the angles ω_1 and ω_2 is small, from the small-angle approximation of *cosine* A_3 can be approximated as $A_1 + A_2$. By plugging $A3$ into (76), we can have

$$e^{j\omega_3} \approx \frac{A_1}{A_1 + A_2} e^{j\omega_1} + \frac{A_2}{A_1 + A_2} e^{j\omega_2}. \tag{77}$$

Setting the new variable ω_c as a weighted average of ω_1 and ω_2 weighted by A_1 and A_2, then (77) can be rewritten as

$$e^{j\omega_3} = e^{j\omega_c}\left(\frac{A_1}{A_1 + A_2} e^{j\hat{\omega}_1} + \frac{A_2}{A_1 + A_2} e^{j\hat{\omega}_2}\right).$$

where $\hat{\omega}_1 = \omega_1 - \omega_c$ and $\hat{\omega}_2 = \omega_2 - \omega_c$. It is noted that $|\hat{\omega}_1| << 1$ and $|\hat{\omega}_2| << 1$ by construction. By the approximation of a exponential function, (77) can be approximated as

$$e^{j\omega_3} \approx e^{j\omega_c}\left(\frac{A_1}{A_1 + A_2}\hat{\omega}_1 + \frac{A_2}{A_1 + A_2}\hat{\omega}_2\right).$$

Similarly the second term of the right-hand side is also small, the above equation can be approximated to the product of two exponential terms as

$$e^{j\omega_3} \approx e^{j\omega_c} \cdot e^{j(\frac{A_1}{A_1+A_2}\hat{\omega}_1 + \frac{A_2}{A_1+A_2}\hat{\omega}_2)}$$
$$= e^{j(\frac{A_1}{A_1+A_2}\omega_1 + \frac{A_2}{A_1+A_2}\omega_2)}$$

REFERENCES

[1] J. R. Vig, "Quartz crystal resonators and oscillators; for frequency control and timing applications - A Tutorial," *U.S. Army Communications-Electronics Command*. `http://www.am1.us/Local_Papers/U11625VIG-TUTORIAL.pdf`.

[2] T. Schmid, O. Sekkat, and M. B. Srivastava, "An experimental study of network performance impact of increased latency in software-defined radios," WinTECH, pp. 59–66, ACM, 2007.

[3] A. Mohammad, X. Hong, M. Islam, and K. Zunnurhain, "Delay analysis of wireless ad hoc networks: Single vs. multiple radio," in *IEEE Conference on Local Computer Networks*, pp. 814–820, 2010.

[4] P. Zetterberg, C. Mavrokefalidis, A. S. Lalos, and E. Matigakis, "Experimental investigation of cooperative schemes on a real-time DSP-based testbed," *EURASIP Journal on Wireless Communication and Networking*, vol. 2009, pp. 1:1–1:15, Feb. 2009.

[5] P. Murphy, A. Sabharwal, and B. Aazhang, "On building a cooperative communication system: testbed implementation and first results," *EURASIP Journal on Wireless Communication and Networking*, vol. 2009, pp. 1–9, 2009.

[6] G. Bradford and J. Laneman, "An experimental framework for the evaluation of cooperative diversity," in *Proc. IEEE CISS*, pp. 641 –645, Mar. 2009.

[7] J. Zhang, J. Jia, Q. Zhang, and E. Lo, "Implementation and evaluation of cooperative communication schemes in software-defined radio testbed," in *Proc. IEEE INFOCOM*, pp. 1 –9, Mar. 2010.

[8] P. Murphy and A. Sabharwal, "Design, implementation, and characterization of a cooperative communications system," *IEEE Transactions on Vehicular Technology*, vol. 60, pp. 2534 –2544, July 2011.

[9] A. Scaglione, D. Goeckel, and J. Laneman, "Cooperative communications in mobile ad hoc networks," *IEEE Signal Processing Magazine*, vol. 23, no. 5, pp. 18–29, 2006.

[10] J. N. Laneman, D. N. C. Tse, and G. W. Wornell, "Cooperative diversity in wireless networks: Efficient protocols and outage behavior," *IEEE Transactions on Information Theory*, vol. 50, pp. 3062–3080, Dec. 2004.

[11] M. Dohler and Y. Li, *Cooperative Communications: Hardware, Channel & Phy*. John Wiley & Sons, Ltd, 2010.

[12] M. Yu and J. Li, "Is amplify-and-forward practically better than decode-and-forward or vice versa?," in *Acoustics, Speech, and Signal Processing, 2005. Proceedings. (ICASSP '05). IEEE International Conference on*, vol. 3, pp. iii/365–iii/368 Vol. 3, 2005.

[13] M. Souryal and B. Vojcic, "Performance of amplify-and-forward and decode-and-forward relaying in rayleigh fading with turbo codes," in *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, vol. 4, pp. IV–IV, 2006.

[14] R. Mudumbai, D. R. Brown, U. Madhow, and H. V. Poor, "Distributed transmit beamforming: challenges and recent progress," *IEEE Communication Magazine*, vol. 47, pp. 102–110, Feb. 2009.

[15] A. Nosratinia, T. E. Hunter, and A. Hedayat, "Cooperative communication in wireless networks," *IEEE Communication Magazine*, vol. 42, pp. 74–80, Oct. 2004.

[16] G. J. Bradford and J. N. Laneman, "An experimental framework for the evaluation of cooperative diversity," *Conference on Information Sciences and Systems*, pp. 641–645, Mar. 2009.

[17] S. Alamouti, "A simple transmit diversity technique for wireless communications," *IEEE Journal on Selected Areas in Communication*, vol. 16, pp. 1451–1458, Oct. 1998.

[18] J. N. Laneman and G. W. Wornell, "Distributed space-time-coded protocols for exploiting cooperative diversity in wireless networks," *IEEE Transactions on Information Theory*, vol. 49, pp. 2415–2425, Oct. 2003.

[19] D. Doberstein, *Fundamentals of GPS Receivers: A Hardware Approach*. Springer, 2012 ed., 2011.

[20] V. F. Kroupa, *Frequency Synthesis: Theory, Design & Applications*. Griffin, 1973.

[21] S. Jagannathan, H. Aghajan, and A. Goldsmith, "The effect of time synchronization errors on the performance of cooperative MISO systems," *IEEE Global Telecommunications Conference*, pp. 102 – 107, Nov. 2004.

[22] A. Mohammad, X. Hong, M. Islam, and K. Zunnurhain, "Delay analysis of wireless ad hoc networks: Single vs. multiple radio," in *IEEE Conference on Local Computer Networks*, pp. 814 –820, Oct. 2010.

[23] N. Benvenuto, S. Tomasin, and D. Veronesi, "Multiple frequency offsets estimation and compensation for cooperative networks," in *IEEE Wireless Communications and Networking Conference*, pp. 891 –895, Mar. 2007.

[24] X. Li, F. Ng, and T. Han, "Carrier frequency offset mitigation in asynchronous cooperative OFDM transmissions," *IEEE Transactions on Signal Processing*, vol. 56, no. 2, pp. 675–685, 2008.

[25] J. Jiang, J. Wei, B. Li, and Y. Wang, "Multiple frequency offset estimation and mitigation for cooperative OFDM system," in *WiCOM*, pp. 1 –7, Oct. 2008.

[26] Z. Li and X.-G. Xia, "An alamouti coded ofdm transmission for cooperative systems robust to both timing errors and frequency offsets," *IEEE Transactions on Wireless Communications*, vol. 7, pp. 1839 –1844, May 2008.

[27] Z. Gao and M.-A. Ingram, "Self-cancellation of sample frequency offset in ofdm systems in the presence of carrier frequency offset," in *IEEE Vehicular Technology Conference*, pp. 1–5, 2010.

[28] O.-S. Shin, A. Chan, H. Kung, and V. Tarokh, "Design of an OFDM cooperative space-time diversity system," *IEEE Transactions on Vehicular Technology*, vol. 56, pp. 2203 –2215, July 2007.

[29] P. Parker, P. Mitran, D. Bliss, and V. Tarokh, "On bounds and algorithms for frequency synchronization for collaborative communication systems," *IEEE Transactions on Signal Processing*, vol. 56, pp. 3742 –3752, Aug. 2008.

[30] V. Jungnickel, M. Schellmann, A. Forck, H. Gäbler, S. Wahls, T. Haustein, W. Zirwas, J. Eichinger, E. Schulz, C. Juchems, *et al.*, "Demonstration of virtual MIMO in the uplink," in *IET Smart Antennas and Cooperative Communications Seminar*, 2007.

[31] Q. Huang, M. Ghogho, J. Wei, and P. Ciblat, "Practical timing and frequency synchronization for OFDM-based cooperative systems," *IEEE Transactions on Signal Processing*, vol. 58, no. 7, pp. 3706–3716, 2010.

[32] A. Nasir, H. Mehrpouyan, S. Durrani, S. Blostein, R. Kennedy, and B. Ottersten, "Transceiver design for distributed STBC based AF cooperative networks in the presence of timing and frequency offsets," *IEEE Transactions on Signal Processing*, vol. 61, no. 12, pp. 3143–3158, 2013.

[33] A. Blair, T. Brown, K. M. Chugg, T. R. Halford, and M. Johnson, "Barrage relay networks for cooperative transport in tactical MANETs," *IEEE Military Communications Conference*, Nov. 2008.

[34] J. W. Jung and M. Ingram, "An rf channel emulator-based testbed for cooperative transmission using wireless sensor devices," in *New Technologies, Mobility and Security, 2008. NTMS '08.*, pp. 1–4, 2008.

[35] L. Thanayankizil and M. A. Ingram, "Reactive robust routing with Opportunistic Large Arrays," *IEEE International Conference on Communications Workshop*, June 2009.

[36] A. Ozgur, R. Johari, D. Tse, and O. Leveque, "Information-theoretic operating regimes of large wireless networks," *IEEE Transactions on Information Theory*, vol. 56, no. 1, pp. 427–437, 2010.

[37] G. Jakllari, S. V. Krishnamurthy, M. Faloutsos, P. V. Krishnamurthy, and O. Ercetin, "A cross-layer framework for exploiting virtual miso links in mobile ad hoc networks," *IEEE Transactions on Mobile Computing*, vol. 6, no. 6, pp. 579 –594, 2007.

[38] A. Scaglione and Y.-W. Hong, "Opportunistic large arrays: cooperative transmission in wireless multihop ad hoc networks to reach far distances," *IEEE Transactions on Signal Processing*, vol. 51, pp. 2082 – 2092, Aug. 2003.

[39] L. Thanayankizil, A. Kailas, and M. A. Ingram, "Routing for wireless sensor networks with an opportunistic large array (OLA) physical layer," *Ad Hoc & Sensor Wireless Networks*, vol. 8, no. 1-2, pp. 79–117, 2009.

[40] H. Dubois-ferriere, D. Estrin, and M. Vetterli, "Packet combining in sensor networks," in *Proc. SenSys*, 2005.

[41] D. O'Rourke and C. Brennan, "A practical implementation of an improved packet combining scheme for wireless sensor networks," in *IEEE International Conference on Communications Workshop*, pp. 332 –336, May 2008.

[42] T. Korakis, Z. Tao, S. Makda, B. Gitelman, and S. Panwar, "It is better to give than to receive: implications of cooperation in a real environment," in *Proc. IFIP Networking*, (Berlin, Heidelberg), pp. 427–438, Springer-Verlag, 2007.

[43] E. Uhlemann and A. Willig, "Hard decision packet combining methods for industrial wireless relay networks," in *International Conference on Communications and Electronics*, pp. 104–108, 2008.

[44] E. Blossom, "GNU software radio." Website. `http://www.gnuradio.org/`.

[45] Ettus Research, "The universal software radio peripheral." Website. `http://www.ettus.com/`.

[46] A. S. Tanenbaum, *Modern Operating Systems*. Upper Saddle River, NJ, USA: Prentice Hall Press, 3rd ed., 2007.

[47] Nsnam, "Network simulator NS-3." Website. `http://www.nsnam.org/`.

[48] M. R. Group, "Georgia tech network simulator (GTNetS)." Website. `http://www.ece.gatech.edu/research/labs/MANIACS/GTNetS/`.

[49] R. Serfling, "Expected value of the sample variance." `http://www.utdallas.edu/~serfling/handouts/`.

[50] G. Lui and H. Tan, "Frame synchronization for gaussian channels," *IEEE Transactions on Communication*, vol. 35, pp. 818 – 829, Aug 1987.

[51] M. I. Skolnik, "Theoretical accuracy of radar measurements," *IRE Transactions on Aeronautical and Navigational Electronics*, vol. ANE-7, pp. 123 –129, Dec. 1960.

[52] L. Zheng and D. N. C. Tse, "Diversity and multiplexing: a fundamental tradeoff in multiple-antenna channels," *IEEE Transactions on Information Theory*, vol. 49, pp. 1073–1096, May 2003.

[53] J. Proakis, *Digital Communications*. McGraw-Hill, 5th ed., 2008.

[54] A. Kailas and M. A. Ingram, "OLA with transmission threshold for strip networks," *IEEE Military Communications Conference*, Oct. 2009.

[55] R. Bhatia and C. Davis, "A better bound on the variance," *The American Mathematical Monthly*, Apr. 2000.

[56] C. R. N. Athaudage and K. Sathananthan, "Probability of error of space-time coded ofdm systems with frequency offset in frequency-selective rayleigh fading channels," in *IEEE International Conference on Communications*, vol. 4, pp. 2593–2599 Vol. 4, 2005.

[57] L. Rugini, P. Banelli, H. Suraweera, and C. Yuen, "Performance of Alamouti space-time coded OFDM with carrier frequency offset," in *IEEE Global Telecommunications Conference*, pp. 1–5, 2011.

[58] M. Krondorf and G. Fettweis, "Numerical performance evaluation for Alamouti space time coded OFDM under receiver impairments," *IEEE Transactions on Wireless Communications*, vol. 8, no. 3, pp. 1446–1455, 2009.

[59] M.-K. Oh, X. Ma, G. Giannakis, and D.-J. Park, "Cooperative synchronization and channel estimation in wireless sensor networks," in *Asilomar Conference on Signals, Systems and Computers*, vol. 1, pp. 238–242, 2004.

[60] T. Schmidl and D. Cox, "Robust frequency and timing synchronization for OFDM," *IEEE Transactions on Communications*, vol. 45, no. 12, pp. 1613–1621, 1997.

[61] Y. S. Cho, J. Kim, W. Y. Yang, and C. G. Kang, *MIMO-OFDM wireless communications with MATLAB*. Wiley. com, 2010.

[62] B. Park, H. Cheon, C. Kang, and D. Hong, "A novel timing estimation method for OFDM systems," *IEEE Communication Letters*, vol. 7, no. 5, pp. 239–241, 2003.

[63] W. Ding, F. Yang, J. Song, and L. He, "Signaling-embedded preamble design for OFDM system with transmit diversity," in *IEE IWCMC*, pp. 211–215, 2012.

[64] R. Prasad, *OFDM for wireless communications systems*. Artech House, 2004.

[65] V. Tarokh, N. Seshadri, S. Member, and A. R. Calderbank, "Space-time codes for high data rate wireless communication: Performance criterion and code construction," *IEEE Transaction Information Theory*, vol. 44, pp. 744–765, 1998.

[66] V. Tarokh, H. Jafarkhani, and A. Calderbank, "Space-time block codes from orthogonal designs," *IEEE Transactions on Information Theory*, vol. 45, no. 5, pp. 1456–1467, 1999.

[67] D. Agrawal, V. Tarokh, A. Naguib, and N. Seshadri, "Space-time coded OFDM for high data-rate wireless communication over wideband channels," in *IEEE Vehicular Technology Conference*, vol. 3, pp. 2232–2236 vol.3, 1998.

[68] S. Mudulodu and A. Paulraj, "A transmit diversity scheme for frequency selective fading channels," in *IEEE Global Telecommunications Conference*, vol. 2, pp. 1089–1093 vol.2, 2000.

[69] F. Ng and X. Li, "Cooperative STBC-OFDM transmissions with imperfect synchronization in time and frequency," in *Asilomar Conference on Signals, Systems and Computers*, pp. 524–528, 2005.

[70] Z. Li and X.-G. Xia, "An alamouti coded OFDM transmission for cooperative systems robust to both timing errors and frequency offsets," *IEEE Transactions on Wireless Communications*, vol. 7, no. 5, pp. 1839–1844, 2008.

[71] G. Ganesan and P. Stoica, "Space-time block codes: a maximum SNR approach," *IEEE Transactions on Information Theory*, vol. 47, pp. 1650–1656, 2001.

[72] B. Sirkeci-Mergen and A. Scaglione, "Randomized space-time coding for distributed cooperative communication," *IEEE Transactions on Signal Processing*, vol. 55, no. 10, pp. 5003–5017, 2007.

[73] T. Paul and T. Ogunfunmi, "Wireless LAN comes of age: Understanding the IEEE 802.11n amendment," *IEEE Circuits and Systems Magazine*, vol. 8, no. 1, pp. 28–54, 2008.

[74] L. Vangelista, N. Benvenuto, S. Tomasin, C. Nokes, J. Stott, A. Filippi, M. Vlot, V. Mignone, and A. Morello, "Key technologies for next-generation terrestrial digital television standard DVB-T2," *IEEE Communications Magazine*, vol. 47, no. 10, pp. 146–153, 2009.

[75] P. Wolniansky, G. Foschini, G. Golden, and R. Valenzuela, "V-BLAST: an architecture for realizing very high data rates over the rich-scattering wireless channel," in *International Symposium on Signals, Systems, and Electronics*, pp. 295–300, 1998.

[76] C.-Y. Lin, W.-C. Peng, and Y.-C. Tseng, "Efficient in-network moving object tracking in wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 5, pp. 1044 –1056, Aug. 2006.

[77] S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, and M. Turon, "Health monitoring of civil infrastructures using wireless sensor networks," *International Symposium on IPSN*, pp. 254 –263, Apr. 2007.

[78] I. F. Akyildiz and M. C. Vuran, *Wireless Sensor Networks*. John Wiley and Sons, Inc., 2010.

[79] B. Sklar, *Digital communications: fundamentals and applications*. Prentice-Hall, Inc., 1988.

[80] H. Jung and M. A. Ingram, "Analysis of intra-flow interference in opportunistic large array transmission for strip networks," *IEEE International Conference on Communications*, June 2012.

[81] A. Akanser and M. A. Ingram, "Semi-cooperative spectrum fusion (SCSF) for aerial reading of a correlated sensor field," *International Conference on Wireless Communication, VITAE*, pp. 732 –736, May 2009.

[82] A. Baghaie and B. Krishnamachari, "Fast flooding using cooperative transmissions in wireless networks," *IEEE International Conference on Communications*, pp. 1–5, june 2009.

[83] S. Ganeriwal, R. Kumar, and M. B. Srivastava, "Timing-sync protocol for sensor networks," *International Conference on Embedded Networked Sensor Systems*, pp. 138–149, 2003.

[84] H. Jung and M. A. Ingram, "SNR penalty from the path-loss disparity in virtual multiple-input-single-output (VMISO) link," *IEEE International Conference on Communications*, June 2013.

[85] B. Sirkeci-Mergen, A. Scaglione, and G. Mergen, "Asymptotic analysis of multistage cooperative broadcast in wireless networks," *IEEE Transactions on Information Theory*, vol. 52, pp. 2531 – 2550, June 2006.

[86] R. Heijmans, "When does the expectation of a ratio equal the ratio of expectations?," *Statistical Papers*, vol. 40, pp. 107–15, Jan. 1999.

[87] D. V. Hinkley, "On the ratio of two correlated normal random variables," *Biometrika*, vol. 56, pp. 635–639, Dec. 1969.

VITA

Yong Jun Chang received his M.S. in Electrical and Computer Engineering from Georgia Institute of Technology in 2009, and B.S. in Electrical and Computer Engineering from Ajou University, South Korea in 2007. During his graduate in Georgia Institute of Technology, he was awarded his Management of Technology (MOT) Certificate in 2012. During his undergraduate, he joined Samsung Software Membership, which is a research incubator established by Samsung Electronics for undergraduate and graduate students. During and after his undergraduate, he had worked for Kyungwoo Systech in South Korea for eight years as an embedded system engineer, where he developed the system control units and infortainment systems for construction equipment and industrial machinery. In 2008, he joined Smart Antenna Research Laboratory (SARL) in Georgia Institute of Technology as a graduate research assistant, under the tutelage of Prof. Mary Ann Weitnauer (formerly Mary Ann Ingram). His doctoral research work addressed the practical design and implementation of the distributed multi-input and multi-output (DMIMO) system on the software-defined radio (SDR) testbed.