Spring 2017

Theory and implementation of scalable, retrodirective distributed arrays

Benjamin Michael Peiffer
University of Iowa

THEORY AND IMPLEMENTATION OF SCALABLE, RETRODIRECTIVE DISTRIBUTED ARRAYS

by

Benjamin Michael Peiffer

A thesis submitted in partial fulfillment of the
requirements for the Doctor of Philosophy
degree in Electrical and Computer Engineering
in the Graduate College of
The University of Iowa

May 2017

Thesis Supervisors: Associate Professor Raghu Mudumbai
Professor Soura Dasgupta

Graduate College
The University of Iowa
Iowa City, Iowa

CERTIFICATE OF APPROVAL

PH.D. THESIS

This is to certify that the Ph.D. thesis of

Benjamin Michael Peiffer

has been approved by the Examining Committee for the thesis requirement for the Doctor of Philosophy degree in Electrical and Computer Engineering at the May 2017 graduation.

Thesis committee: _____
Raghu Mudumbai, Thesis Supervisor

_____
Soura Dasgupta, Thesis Supervisor

_____
Anton Kruger

_____
Er-Wei Bai

_____
Nixon Pendergrass

To all those who believe anything is possible, and especially to all those who don't.

"And the ones who would not make war? Can they stop it?"
"I do not know."

Ernest Hemingway, *A Farewell to Arms*

ACKNOWLEDGEMENTS

It would be impossible for me to properly acknowledge everyone who has contributed to my journey in completing this thesis. First and foremost, I must thank my mom and dad, my grandparents, and my brothers. Growing up, I never considered going to college as a necessity. I still don't. It was these people who taught me to follow my heart and do what I love - and it is for that reason that I have completed this thesis. I have completed this thesis, not for the destination, but for the journey. This has been a tremendous learning experience, one that has forced me to go beyond myself and for that I have to thank my research advisors, Raghuraman Mudumbai and Soura Dasgupta, as well as many other faculty members at the University of Iowa. My thinking was constantly challenged, and I was forced to become better in my craft. I would also like to thank Sairam Goguri, with whom I have formed a valuable friendship and a fruitful research partnership. I would also like to thank the students that I have been able to work with during my time as a doctoral student, they have opened my eyes to many powerful lessons.

Finally, I would like to thank a close friend, who once told me a story about his son who loved to play the saxophone. As the story went, the young man spent all his free time playing the instrument, took extra lessons, played in the honor band, and one day came home and told his father: "Dad, I don't want to play the saxophone anymore." My friend loved his son and only wanted him to be happy, "okay", he said.

ABSTRACT

A Distributed Multi-Input Multi-Output (DMIMO) system consists of many transceivers coordinating themselves into a "virtual antenna array" in order to emulate MIMO capabilities. In recent years, the field of research investigating DMIMO Communications has grown substantially. DMIMO systems offer all of the same benefits of standard MIMO systems on a larger scale because arrays are not limited by the physical constraint of placing many antennas on a single transceiver. This additional benefit does come at a cost, however. Since nodes are distributed and run from independent clock signals and with unknown geometry, each one must its own obtain channel state information (CSI) to the target nodes. In existing DMIMO architectures, array nodes depend on feedback from target nodes to properly synchronize. This means that target nodes must be cooperative and are responsible for the overhead calculating and transmitting CSI feedback to each node in the array.

Within this work, we develop a set of techniques for Retrodirective Distributed Antenna Arrays. Retrodirective arrays have traditionally been used to direct a beam towards a target node, but the work in this thesis seeks to develop a more generalized definition of retrodirectivity. By our definition, a retrodirective array is one that acquires CSI to one or more intended targets simply by listening to the incoming transmissions of those targets; the array may subsequently use this information to do any number of typical MIMO tasks (i.e., beamforming, nullforming, spatial multiplexing, etc.). We explore two primary techniques: i) distributed beamforming and ii)

distributed nullforming. Beamforming involves focusing transmitted power towards a specific target node and nullforming involves directing transmissions of array nodes to cancel one another at a specific target node. We focus on these techniques because they can be thought of as basic building blocks for more sophisticated DMIMO techniques.

We first develop the theory for retrodirective arrays. Then, we present an architecture for the implementation of this theory. Specifically, we focus on the presynchronization of the array, which involves use of a master/slave architecture and a timeslotted message exchange among the array nodes. Finally, developing algorithms to make these arrays both robust and scalable is the focus of this thesis.

PUBLIC ABSTRACT

A Distributed Multi-Input Multi-Output (DMIMO) system allows techniques traditionally used in multiple-antenna devices, such as Wi-Fi routers, to be applied to groups of single-antenna devices, such as cell-phones. In this work, we develop the theory for making these DMIMO systems possible when targets are non-cooperative. This entails distributed arrays coordinating their transmissions such that unknown offsets are cancelled. We are most interested in beamforming, which involves transmissions being coordinated to add constructively at a specific target, and nullforming, which involves transmissions being coordinated to cancel one another at a specific target. These two techniques can be viewed as essential MIMO building blocks.

In addition to presenting theory, we use the Ettus Research Universal Software Radio Peripheral (USRP) to perform experimental verification of our work. These experiments demonstrate array synchronization and beamforming without co-operation from the target and nullforming using an algorithm presented in previous papers.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

CHAPTER 1
INTRODUCTION

1.1 Overview of DMIMO

By definition, a Distributed Multi-Input Multi-Output (DMIMO) system consists of many transceivers coordinating themselves into a "virtual antenna array" in order to emulate MIMO capabilities. In recent years, the field of research investigating DMIMO Communications has grown substantially [25]. DMIMO systems offer all of the same benefits of standard MIMO systems on a larger scale because arrays are not limited by the physical constraint of placing many antennas on a single transceiver [26, 10]. This additional benefit does come at a cost, however. Since nodes are distributed and run from independent clock signals and with unknown geometry, nodes must be synchronized [19].

There are a number of interesting applications for MIMO systems. A few examples are: *Beamforming* causes signals from multiple antennas to constructively interfere with one another at a target, resulting in increased power. *Nullforming* causes signals to destructively interfere with one another at a target, allowing an array to hide a transmission from another user. *Joint Beam and Nullforming* allows users to achieve approximate beams and nulls to more than one target simultaneously. *Spatial Multiplexing* allows multiple streams of data to be sent simultaneously. The primary focus of our work is beamforming and nullforming.

1.1.1 Beamforming

There are several different definitions and types of beamforming; within this work, we define beamforming to mean phase coherent beamforming. Phase coherent beamforming means that the phases of array transmissions are adjusted so that they will arrive at a target in phase with one another. Conceptually, the formation of a beam is straight forward. Multiple users must send a message signal that achieves phase alignment at a target (i.e., adds constructively).

One can imagine a number of interesting applications for beamforming. In a military application, an array may want to jam an enemy target - we refer to this as an electronic warfare application (Fig. 1.1). In the same military application, an array may want to use receive beamforming to pickup a message from a distant enemy transmitter - we refer to this as an electronic sensing application (Fig. 1.2).

1.1.2 Nullforming

In forming a null, each user must adjust its message such that it is negative to the sum of all other messages simultaneously arriving at the target. This provides an implicit challenge for fully wireless arrays because the behavior of a single node is dependent on the behavior of all other nodes. Like beamforming, nullforming has a number of interesting applications. In a military application, an array may want to send a protected transmission to another user and could do so by applying a null to an enemy node to prevent them from observing the transmission.

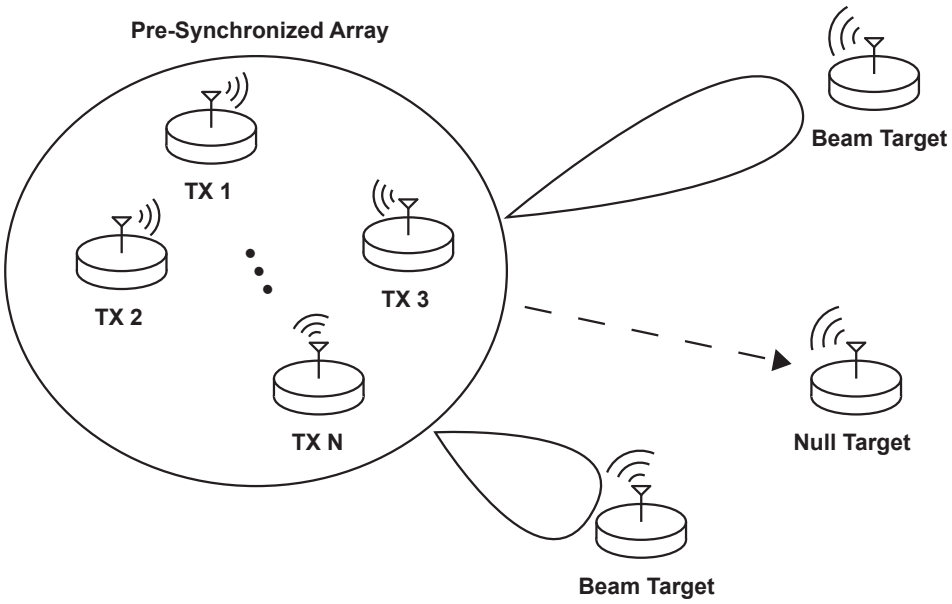

Figure 1.1: An electronic warfare system diagram.

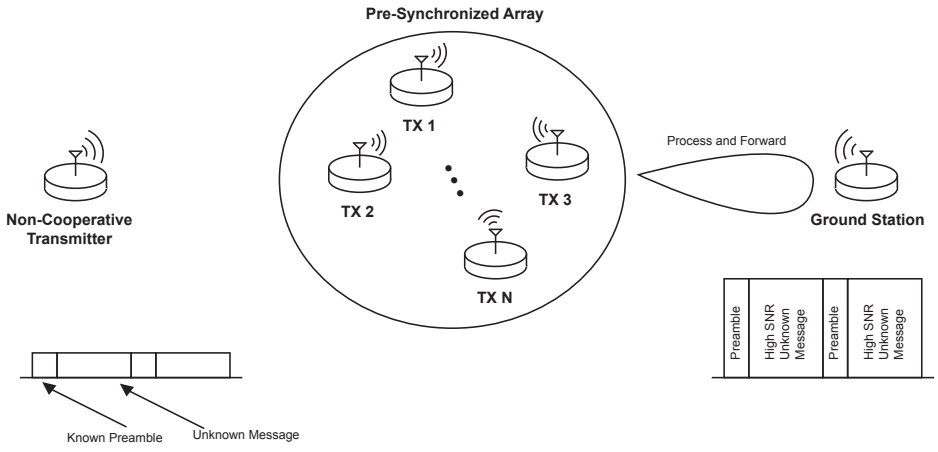

Figure 1.2: An electronic sensing system diagram.

1.1.3 Towards Retrodirective Systems

In the beamforming and nullforming applications we have described, target nodes are by nature uncooperative. This motivates the need for a class of techniques for DMIMO systems that will accomplish these tasks without cooperation from targets, a feature that existing methods for beam and nullforming lack. In the subsequent sections, we will introduce the theory of retrodirective distributed arrays. These arrays can be used to direct beams to noncooperative targets. Developing robust and scalable retrodirective distributed arrays is the focus of this thesis.

1.2 Retrodirective Arrays

Retrodirective techniques rely on reciprocity of wireless channels, i.e. the path from A to B is the same as that from B to A. However, distributed array nodes encounter *effective channels*, which include clock offsets between the nodes and mismatches between the transmit and receive hardware, that *are not reciprocal*. Recent work [8] has developed powerful methods that allow distributed array nodes to jointly compensate for all non-reciprocal effects and synchronize the array for retrodirective transmission. These methods form the basis of our experimental demonstration of retrodirective beamforming in [30, 31].

1.2.1 Generalizing the Concept of Retrodirectivity

It should be noted that classical retrodirectivity [23, 32] implies beamforming but, while that is the focus of our discussion in this section, we really seek to develop a more generalized notion of retrodirectivity in this work. We employ the term

"retrodirectivity" to refer to a class of techniques where an array acquires channel state information (CSI) to one or more intended targets simply by listening to the incoming transmissions of those targets. The CSI can then be used to implement a variety of MIMO transmit precoding techniques.

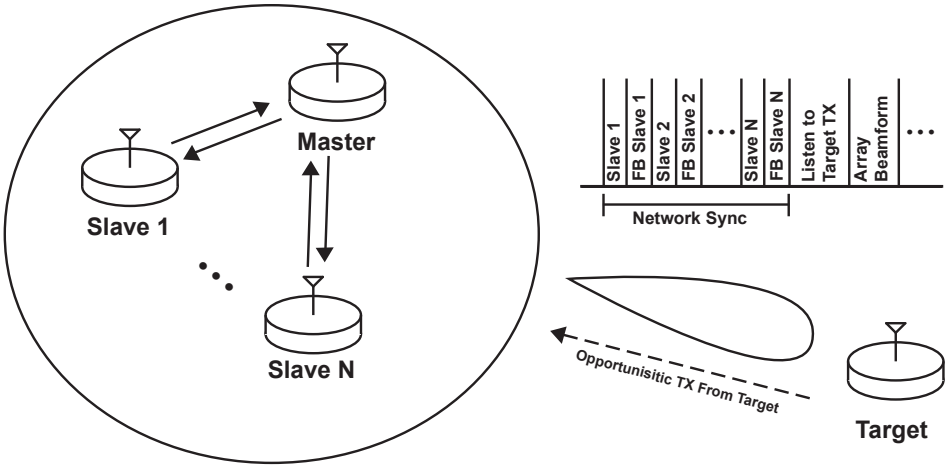

Figure 1.3: An array opportunistically listening to a transmission from the target node to perform retrodirective beamforming.

1.2.2 Non-Retrodirective Approaches to Beamforming

We will now briefly introduce two non-retrodirective methods for distributed beamforming. This will help highlight the advantages and disadvantages of the retrodirective approach.

- **Explicit Channel State Feedback**: One way for an array to send a message to a target such that it arrives with a phase coherence, is for that transmitter

to have a explicit knowledge of the channel to the intended target. This can be achieved with a round-trip exchange between the target and each element of the array. For instance, a transmitter sends a message to a target, and the target responds by sending back a message containing the observed complex channel gain.

Note that the beam target must cooperate with the array. Also note the obvious lack of scalability: as the size of an array grows large, so too does the number of message exchanges required from the target.

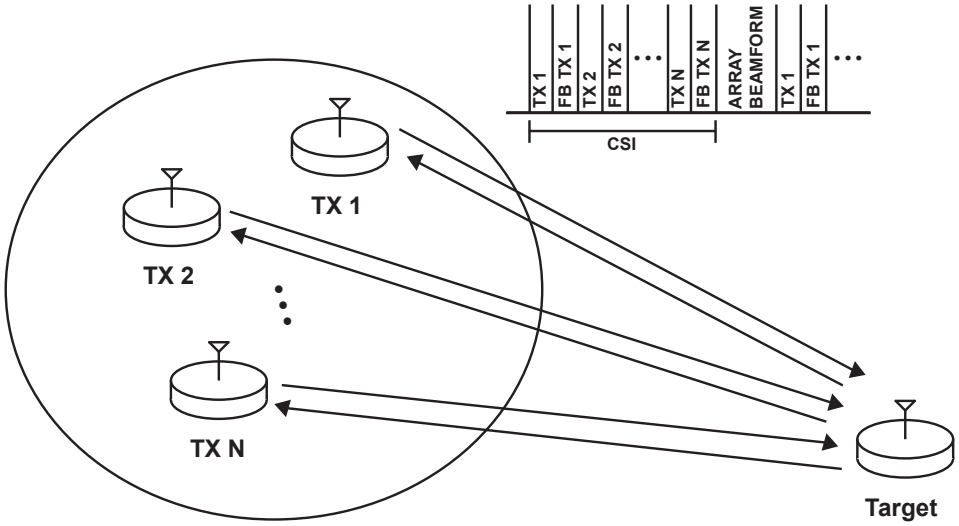

Figure 1.4: An array relying on explicit feedback from the target node.

- **Aggregate Feedback Methods**: Another, less obvious, way of achieving phase coherence is for the target node to provide implicit knowledge of the channel to the array. That is to say, the array forms a joint message to a tar-

get without channel knowledge. The target provides feedback to entire array regarding this message. Members of the array use a sequence of joint messages, phase and amplitude adjustments and resulting feedback to learn their own channel. This idea, known as *aggregate feedback* was introduced in [24] wherein the author shows that an aggregate feedback tied to the beam SNR can be used in an iterative algorithm that achieves sufficient phase coherence to form a beam. The "1-bit Feedback" method [27] demonstrates that this principle can be applied with only 1-bit of aggregate feedback.

Note that the beam target must again cooperate with the array. We revisited this concept in [17] to show that an array can achieve channel estimates more rapidly and more precisely by using more than 1-bit of feedback. Even with a faster rate of convergence, the number of message exchanges required to learn a channel in this way scales with the number of nodes in the array.

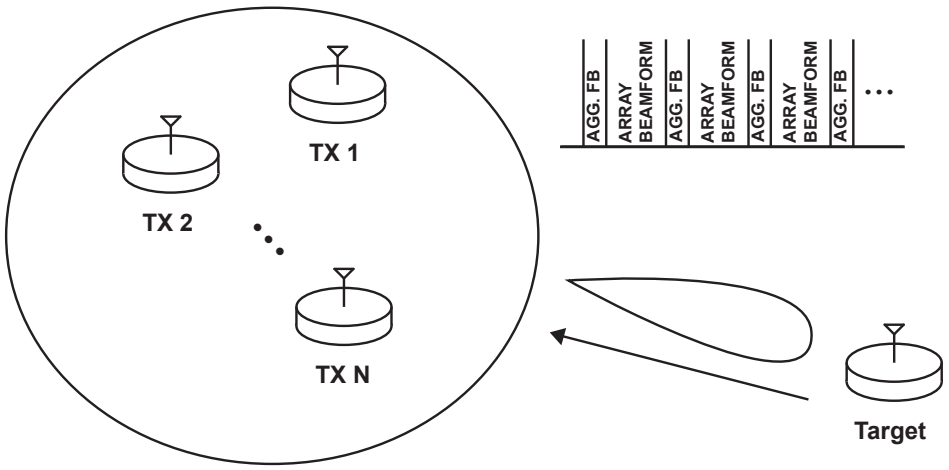

Figure 1.5: An array relying on aggregate feedback from the target node.

1.3 Theory of Retrodirective Arrays

In this section, we introduce a general theory of retrodirective distributed arrays and the notation associated with different aspects of a retrodirective system.

The complex *effective channels* between nodes i and j are:

$$h_{ij} = \text{clk}_i \; T_i \; \tilde{g}_{ij} \; R_j \; \text{clk}_j^{-1}$$
$$h_{ji} = \text{clk}_j \; T_j \; \tilde{g}_{ji} \; R_i \; \text{clk}_i^{-1} \tag{1.1}$$

where clk_k represents the clock offset of node k (relative to some global clock reference), T_k, R_k represent the complex gain of the transmit and receive hardware, and $\tilde{g}_{kl}$ represents the actual wireless propagation channel gain. Reciprocity implies $\tilde{g}_{ij} = \tilde{g}_{ji}$, but in general $h_{ij} \neq h_{ji}$.

The key insight behind the calibration method in [8] is that (1.1) can be rewritten as:

$$h_{ij} = c_i \; g_{ij} \; c_j^{-1} \text{ and } h_{ji} = c_j \; g_{ji} \; c_i^{-1} \tag{1.2}$$
$$\text{where } c_k \doteq \text{clk}_k \sqrt{\frac{T_k}{R_k}}$$
$$g_{ij} \equiv g_{ji} \doteq \tilde{g}_{ij}\sqrt{T_i T_j R_i R_j}$$

Equation (1.2) shows that the non-reciprocal channels h_{ij}, h_{ji} can be written as a combination of an effective reciprocal channel g_{ij} and *anti-reciprocal* effective clock offsets c_i, c_j, that include also the effects of any hardware mismatches.

The calibration algorithm in [8] is based directly on (1.2) and uses a master-slave architecture where one array element M acting as the master node separately exchanges a pair of messages with each slave node $\{A,B,...\}$. Slave nodes take turns

sending messages containing a known preamble sequence from which the master calculates the complex effective gains (e.g., h_{AM}) of the "reverse channels" from the slaves to the master. The master then sends a message with a known preamble sequence which also contains in its payload its estimates of the effective reverse channel gains. The slave nodes use the known preamble to estimate the effective complex gains of their respective "forward channels" (e.g., h_{MA}) from the master to themselves and obtain the "reverse channel" gains from the payload.

At the end of this message exchange, each slave, k, has estimates of both h_{kM} and h_{Mk}. Using these two with (1.2), each slave is able to calculate a calibration constant:

$$a_k \doteq \frac{c_k}{c_M} \equiv \sqrt{\frac{h_{kM}}{h_{Mk}}} \tag{1.3}$$

Slaves also receive a sounding signal from a target T to calculate channel gains from the target (e.g., h_{TA}). Slaves nodes achieve coherence at T, with M and each other by employing the complex transmit weights b_k where

$$b_k \doteq a_k^2 h_{Tk}. \tag{1.4}$$

The method above assumes that each node knows the target's sounding signal, and thus can estimate the propagation path to it. Our recent work extends this to the case where no transmitter knows the sounding signal from the target. In this more difficult scenario a slave must apply the precoder below, while the master requires no precoding:

$$[h_{kM}]^{-1} h_{TM} [h_{Tk}]^{-1} h_{Mk}. \tag{1.5}$$

Observe the slave still acquires h_{kM} and h_{Mk} by exchanging information with the master. It knows neither h_{TM} nor h_{Tk}. However, it can learn $h_{TM}\left[h_{Tk}\right]^{-1}$ directly. For this each node opportunistically hears an unknown signal from the target. Additionally, the master forwards the signal it receives from the target to all of the slaves. These pairs of signals together suffice to estimate missing quantities like $h_{TM}\left[h_{Tk}\right]^{-1}$.

1.4 Existing Work

Our prior work includes the development of an architecture for retrodirective distributed arrays and the implementation of this architecture in over-the-air demos. This work includes first of kind demonstrations of array pre-synchronization, retrodirective transmit beamforming, and fully wireless transmit nullforming. We briefly introduce this work and relevant work from other authors.

1.4.1 Distributed Transmit Beamforming

In [37], the authors describe a spatial multiplexing protocol to scale WiFi network capacity linearly with the number of base stations. The protocol uses zero-forcing to do downstream interference management, such that each user receives from only one base station at a time. The work does not consider upstream interference management, a deliberate omission given the assumption that upstream channels will be more sparsely utilized. As in many of the experiments discussed, as well as our own, this experimental testbed uses the Universal Software Radio Peripheral (USRP) from National Instruments [39]. In order to achieve LO synchronization, the author makes use of the gigabit ethernet back-channel shared by each base station.

The authors of [14] have received publicity for their work in the development of a distributed MIMO capability for LTE devices. Like the WiFi implementations that were previously discussed, this work is dependent on a high speed wired back-channel, this time the fiber-optic network connecting the various base stations. Once again, this protocol primarily serves to increase downstream capacity. There is no clear discussion of the protocol for LO synchronization. The work in [5] describes an architecture for coordination of a large distributed MIMO network in the context of cellular base stations with a high speed wired backhaul and a coordinating server.

In [27], the authors apply the previously mentioned "1-bit Feedback" method [24] for distributed beamforming to form a beam from an array to a target node. The array and the target have a common LO signal to eliminate the need for frequency synchronization. In [36], this idea of aggregate feedback for beamforming is carried forward by applying the system from [27] with independent LO signals for each node in the array. This produces the first of its kind, fully wireless system for distributed transmit beamforming. In this case, a target node distributes its clock signal wirelessly on a sideband signal. Each of the cooperating nodes estimates the frequency of this sinusoidal signal and the 1-bit aggregate feedback from the target receiver is again used to ensure phase coherence after a period of synchronization. In [34], the authors use an EKF as suggested in [11] to produce more precise frequency and phase estimates for beamforming.

Further developments in [7, 38] showed a long range outdoor demonstration of receiver-coordinated distributed beamforming between fully-wireless nodes. It is

of note that the hardware used in this demonstration was customized and the beamforming is done on continuous-wave (CW) signals. In this system, the cooperative receiver sends explicit phase compensation feedback on a WiFi side channel so that nodes in the array can correct for the offset during their beamforming transmit time slots.

Finally, [12] demonstrates fully-wireless acoustic beamforming using round-trip carrier synchronization. We presented the first pre-synchronized array at RF frequencies in [31] and the first demonstration of retrodirective transmit beamforming in [30]. In demonstrating retrodirective beamforming, the target known sent a beacon message with no explicit feedback - only the preamble sequence of the message was known to the members of the array. These experiments and their challenges will be described in the following chapter.

1.4.2 Distributed Transmit Nullforming

There has been no previous demonstration of distributed transmit nullforming. While the focus of this work is retrodirective techniques, we first lay the foundation for a retrodirective nullforming by demonstrating nullforming with explicit feedback. We will report the first of its kind demonstration of fully-wireless distributed transmit nullforming in [29], based on the algorithm presented in [20].

1.5 Outline: Towards Scalable Retrodirective Arrays

In the remainder of this thesis, we will explore the following topics:

- **Pre-synchronization of distributed arrays using a master-slave archi-**

tecture: In this chapter, we present our architecture for antenna array pre-synchronization and describe a series of proof-of-concept experiments.

- **Advanced Methods in Array Synchronization**: We present two new ideas that will enhance the capabilities of our existing pre-synchronization scheme, improving both scalability and robustness.
- **Applying Retrodirectivity to More Complex MIMO Operations**: We present a pre-coding scheme that will allow us to explore more DMIMO algorithms in addition to phase-only beamforming.

CHAPTER 2
PRE-SYNCHRONIZATION OF DISTRIBUTED ARRAYS USING A MASTER-SLAVE ARCHITECTURE

In this chapter, we present an idealized architecture for application of the theory of retrodirective distributed arrays and a broad discussion of the challenges of applying this theory to our own experimental testbed. This chapter motivates the development of more advanced techniques to take advantage of the power of retrodirective arrays.

2.1 General Theory of Retrodirective Arrays

In chapter 1, we introduced a general theory for retrodirective distributed arrays [8]. From electromagnetic theory, it is well known that wireless propagation channels will be reciprocal at a given frequency [13]. However, the other effects such as those contributed by RF amplifiers, filters and clock mismatches are not guaranteed to be reciprocal. There have been attempts to develop specialized architectures to account for these [28], but, because of the specialized hardware required, an application of these architectures to a general purpose, off-the-shelf, software-defined radio is not possible. While some calibration methods have been proposed [9], it is more interesting to consider the use of relative calibration concepts [18]. Essentially, this method uses a round-trip message exchange to determine the relative clock offset between two nodes.

In [8], we described how to apply this relative calibration concept to a DMIMO

array for distributed beamforming. This retrodirective array method removes the requirement for explicit feedback from the target; instead, the array can opportunistically listen to the target and direct a beam to it based on the one-way channel estimate from the target and internal relative calibration only.

2.2 A System Architecture for Implementing Retrodirective Distributed Arrays

We will now describe an idealized system architecture for retrodirective distributed arrays. Nodes in the array operate with independent local oscillators and must rely on only over-the-air messages for synchronization. Target nodes also operate from an independent local oscillator and are not allowed to provide any explicit feedback to nodes in the array.

2.2.1 Master/Slave, Time-Slotted Architecture

A round-trip exchange is one way to implement a retrodirective system. It is convenient to implement a timeslotted architecture where master and slave devices take turns sending packets with relevant information.

We employ the TDMA scheme shown in Fig. 2.1. The following sections describe each time-slot in this system in detail.

1. **Pre-Synchronize Array:** We refer to a pre-synchronized array as one in which each slave knows its clock offset to the master. In practice, this means both frequency and phase synchronization have occurred. Thus, we first must pre-synchronize the array. Phase synchronization requires a round-trip exchange

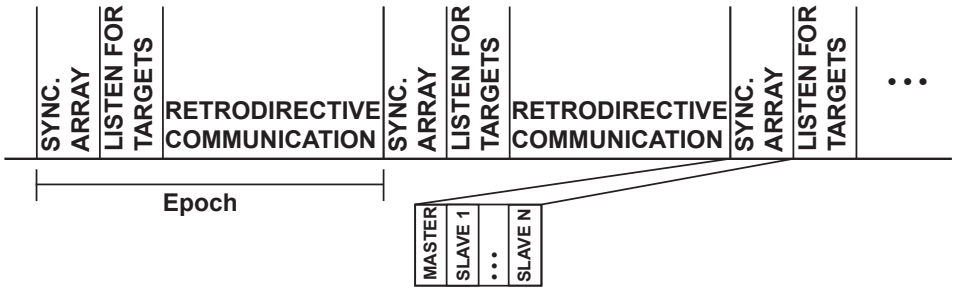

Figure 2.1: A general time-slot structure for our system.

[19, 33]. The master sends a broadcast message (Fig. 2.2), which is then followed by a response message from each slave (Fig. 2.3). The master broadcast packet includes individual channel feedback for each slave node in the array.

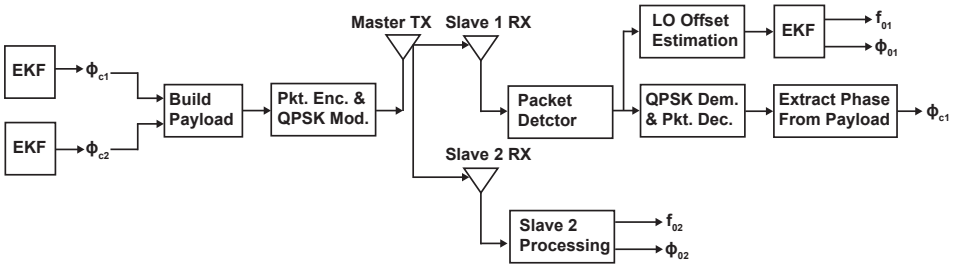

Figure 2.2: The master broadcast packet is received and processed by each slave.

2. **Listen for Targets:** During this time, each member of the array opportunistically listens for messages that target nodes have sent (Fig. 2.4). The members of the array can use this message to make estimates of the channel from the target. The channel information is needed for retrodirective communication.

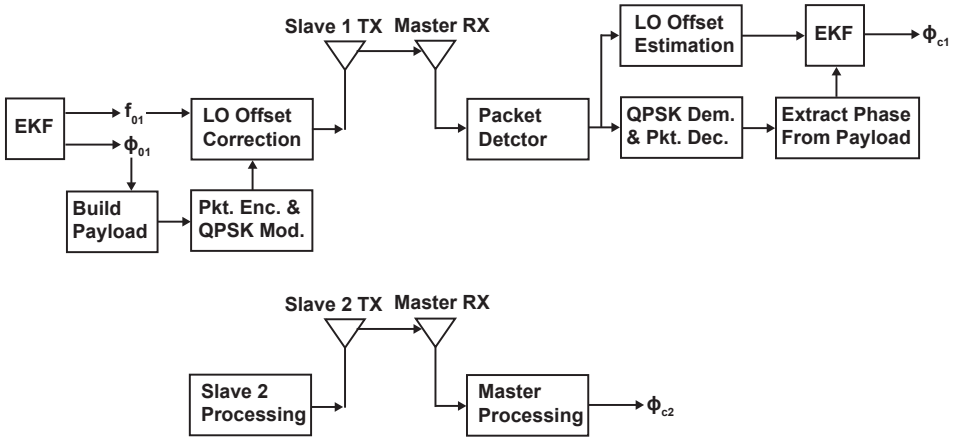

Figure 2.3: Each slave transmits a response packet to the master.

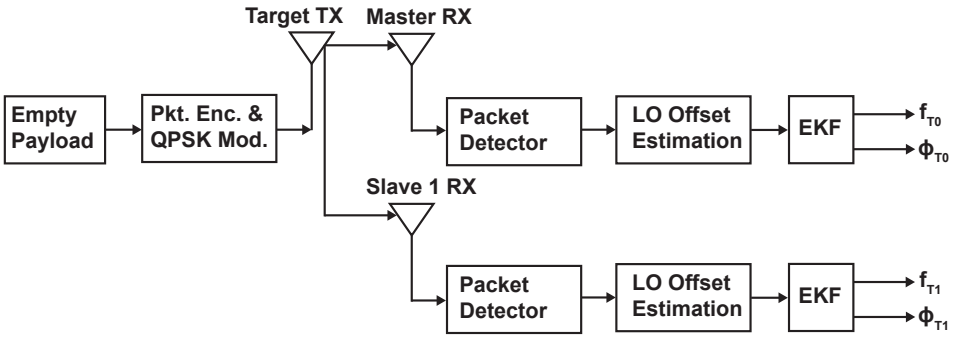

Figure 2.4: Each target periodically sends out messages, to which the array opportunistically listens and makes estimates.

3. **Retrodirective Communication:** The final timeslot is reserved for retrodirective communication. During this timeslot, the members of the array use the channel offsets that they have measured to direct transmissions. [8] describes a process for forming beams to a target, but other techniques are possible and will be developed in later chapters.

2.2.1.1 LO Frequency Offset

Since master, slave and target nodes have independent local oscillators, they will operate with frequency offsets. Managing these frequency offsets is an important part of implementing our idealized architecture for pre-synchronization. We will introduce the intricacies of this process in later sections, but note that in Fig. 2.3, slave nodes make a frequency correction prior to sending a response packet back to the master node.

2.2.2 Fundamental Constraints and Simplifying Assumptions

We make several simplifying assumptions in our experimental work, and those are clarified here.

1. **Epoch Length**: We assume that the epoch length is short relative to the rate of change of channels (i.e., pre-synchronization occurs frequently).
2. **Stationary Nodes**: We assume that nodes are stationary.
3. **Signal Bandwidth**: We restrict ourselves to narrowband systems, although most of our discussion can be generalized to wideband systems.
4. **Preamble Sequence and Modulation Type**: Throughout this work, we

make use of known preamble sequences and known modulation types. Furthermore, external target nodes are are assumed to make transmissions at the same rate as nodes in the array and deliberately avoid collisions with the other transmitters in the system. These assumptions are important in reducing the complexity of our system, but each one of them could be overcome at a cost.

2.3 Performance Requirements for Synchronizing Retrodirective Arrays

In this section, we describe many of the important practical considerations in realizing our idealized architecture. We restrict ourself to general purpose off-the-shelf hardware and open-source software. For our hardware platform, we chose the USRP N2X0 software defined radio platform with the WBX RF daughterboard [39] at a center frequency of 915 MHz, but nothing we have done is limited to this hardware or center frequency. In fact, the beauty of using a software-defined-radio platform is that any hardware that supports our chosen software platform could be used with only minor adjustments to software settings. All nodes used off the shelf computers running Linux software and the open-source GNU Radio platform [15].

In our system, all message signals are encoded as differential QPSK modulated packets which are received and sampled at 200 kbps. Each packet includes a known preamble sequence which is used for channel gain measurement. Again, our choice of QPSK is somewhat arbitrary. Our system could easily be translated to another narrowband modulation scheme with only a change in the frequency offset estimator, which has a few steps that are specific to QPSK. Translating to a wideband scheme

would require more work and is discussed in section 5.5 as an important topic for future work.

Crucial to achieving accurate synchronization is the ability of the nodes to generate precisely timed transmissions, and we make heavy use of the "burst tagging" mechanism in GNU Radio to achieve this. A further discussion of timing and precision is in section 2.3.2.

2.3.1 Oscillator Stability: Diagnosis, Analysis, and Solution

2.3.1.1 Desired Characteristics of Oscillators for a Synchronized System

When tracking relative frequency and phase offsets in a DMIMO system, there are two primary oscillator behaviors that we are concerned with. Firstly, the rate of the phase drift of an individual oscillator is critical. If an oscillator drifts too rapidly between synchronization periods (consider Fig. 2.1), it will be impossible to maintain good performance (i.e., beamforming gain or nullforming suppression) by the end of a timeslot. Of course, in a system with poor oscillator drift specifications, the epoch time could be shortened, but is limited by the packet size, the number of packets that needs to be sent in each epoch, and the processing time required by the nodes in the array.

Secondly, oscillator frequency stability is an important concern. Some oscillators may make jumps in frequency to try and correct for certain types of drift (i.e., temperature related drift). This can interfere with the synchronization process because packets arrive intermittently. Any general purpose filter that might be used to

improve noisy frequency estimates would be disrupted when an oscillator is making coarse adjustments without hysteresis. The next section will discuss these phenomena further.

2.3.1.2 Behaviors of Available Oscillators

The first oscillator that we investigated was the USRP built in oscillator, which is a Fox Electronics FOX924 [1] or equivalent. The USRP N2X0 includes a digital compensation mechanism that makes fine adjustments to the oscillator frequency, this mechanism can be switched off by setting the *reference clock* parameter to *external*. If no external clock is connected, this will cause the internal oscillator to free-run without training from the FPGA. When running in this mode, the one-way frequency offset between nodes is typically greater than 2 kHz. This large frequency offset is not a major concern, as long as the relative drift is not too quick. As shown in Fig. 2.5, the frequency and phase do tend to drift quite quickly (as much as 180°per epoch) and this motivated the need to investigate external oscillator options.

The second oscillator that we investigated was the Crystek PPRO30 [2], a temperature-controlled oscillator (TCXO). The one-way frequency offset between nodes with this TCXO is typically less than 500 Hz and the rate of drift is much slower than with the FOX924. However, relative drift rates can be as high as 36° per epoch, still too large to achieve consistently high performance. In addition, the TCXO can cause frequency jumps that are quite large, in some cases even exceeding 100 Hz. As suggested earlier and shown in Fig. 2.6, this behavior disrupts the filtering mech-

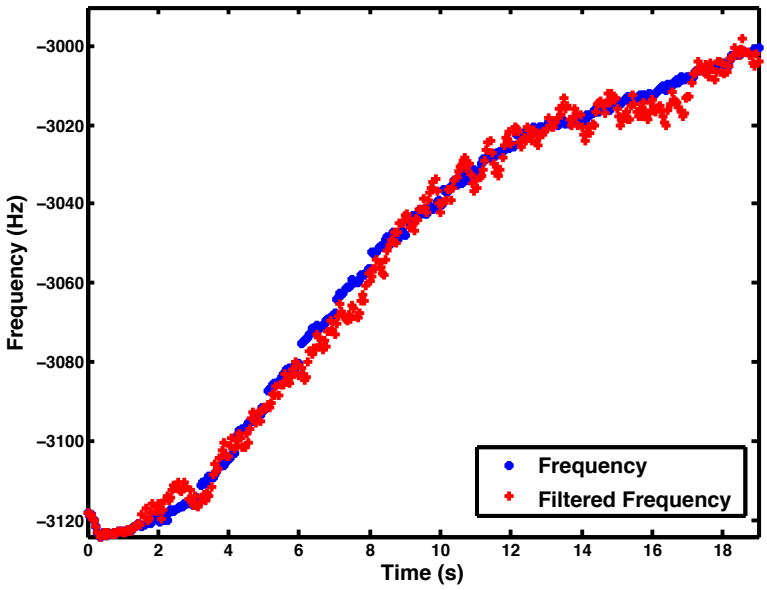

Figure 2.5: A series of unfiltered and filtered estimates of the frequency offset between two nodes in the system while using the USRP internal oscillator without digital training.

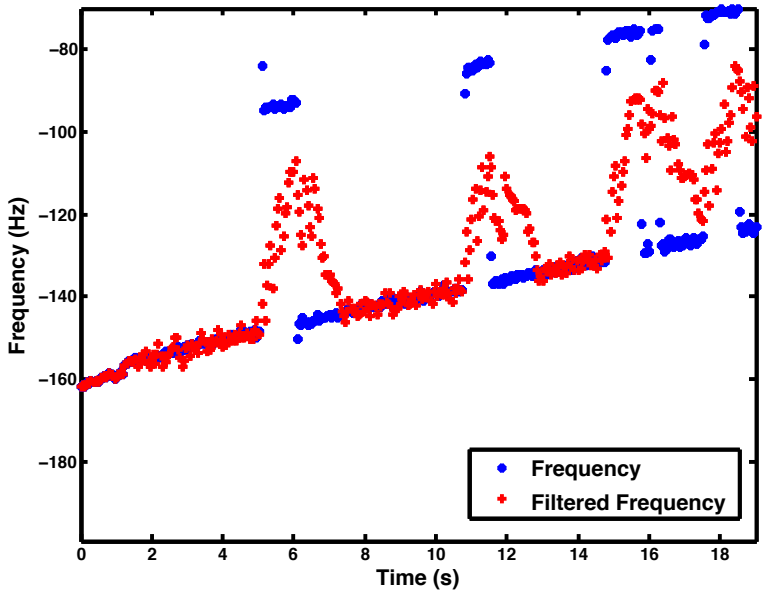

Figure 2.6: A series of unfiltered and filtered estimates of the frequency offset between two nodes in the system while connected to an external TCXO.

anism of our system. We could make our filter more intelligent with some kind of binning structure, but the nature of the frequency discontinuities would make the software dependent on a specific set of hardware - this is not acceptable for our use.

2.3.1.3 Ovenized Oscillator Peripheral

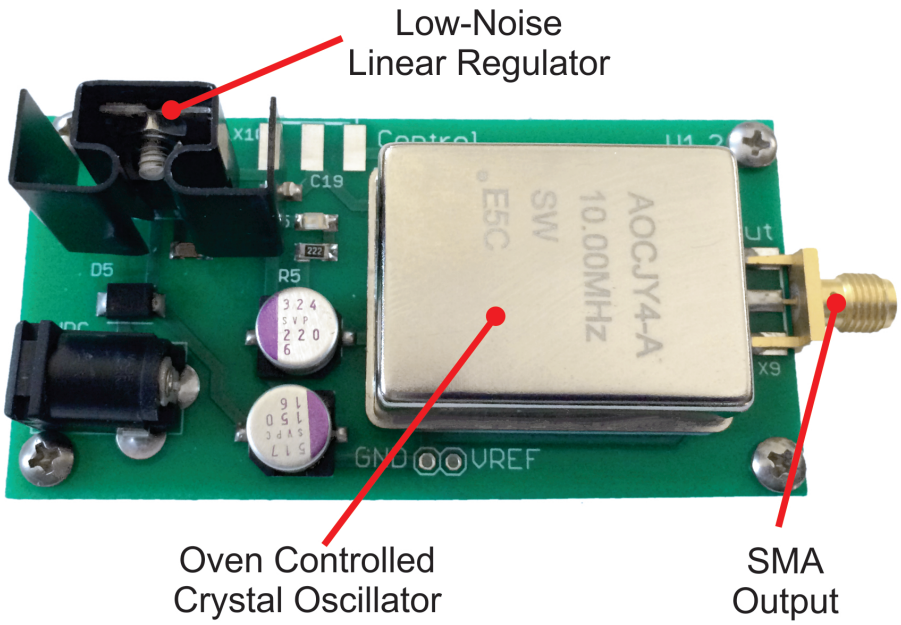

Figure 2.7: A photograph of an external oscillator board that is connected to each USRP in the system to provide a stable frequency reference.

In order to provide a reference that drifts sufficiently slowly and does not employ frequency discontinuities in its controller, we used an external oven-controlled oscillator (OCXO). Fig. 2.7 shows the breakout board that we developed for the Abracon AOCJY4 [4] (an initial prototype of this breakout board was used in [31]

with the similar AOCJY2 [3]).

Fig. 2.8 shows the stability of this oscillator over time. While the relative frequency offset does wander (as one might expect), it is relatively constant over time. In fact, this particular plot shows a net change of only about .2 Hz during the 20 second interval shown.

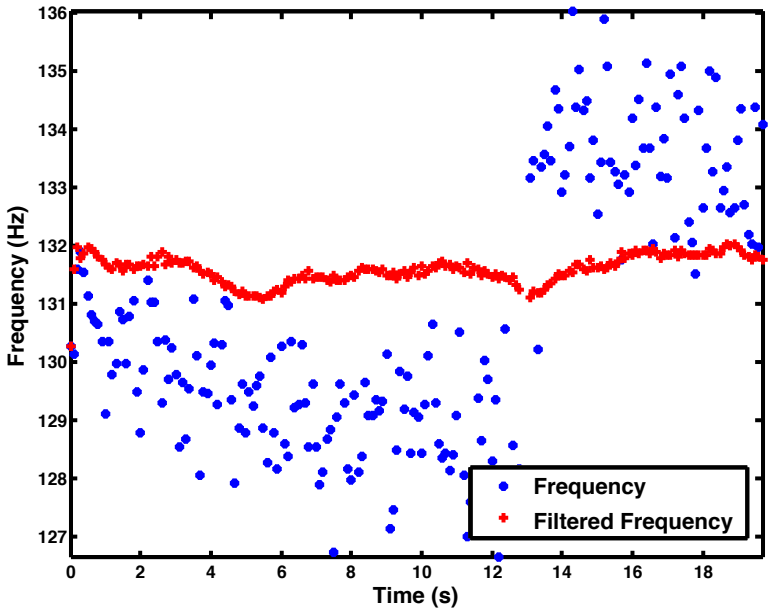

Figure 2.8: A series of unfiltered and filtered estimates of the frequency offset between two nodes in the system while connected to an external OCXO.

2.3.2 Frequency, Phase and Delay Estimation

In order to precisely synchronize the LO of one node to another, we must have algorithms for frequency, phase and timing estimation. Each packet includes a

preamble portion that can be used in making these estimates. In order to make a phase offset estimate, we use the maximum likelihood estimate.

In making a frequency estimate, we use a 2-step process. First, we perform an FFT to narrow the frequency offset to a particular bin. Due to the packet length in our system, a 256-point FFT is used. The second step in the process is to do a search for the correct frequency offset by evaluating the DTFT at various frequencies within the selected bin. This provides a coarse estimate of the system frequency offset.

In order to estimate the delay between packets, we count the number of samples between them. This provides a coarse estimate of the delay in between packets that is accurate to within 1/2-sample, in our case 2.5 μs. As previously mentioned, transmitted packets are timed precisely using the built in USRP "tx_time" tag. However, since transmitter DACs and receiver ADCs are clocked independently, the detection of transmitted energy can slip up to half of a sample as shown in Fig. **??**. This can be considered a coarse estimate of delay.

2.3.2.1 Sub-Sample Delay Estimation

While a coarse estimate of delay was sufficient in our early experiments in beamforming [31, 30], it was evident that a more fine-grained delay estimation would be necessary for more advanced retrodirective and DMIMO techniques such as null-forming. This motivated the need for a joint frequency-delay estimation derived in [16]. This algorithm provides a sub-sample delay estimate that allows us to approach the physical limits of the transmit tagging feature of the USRP (tens of ns).

2.3.3 Tracking Offsets of Unsynchronized External Target Node

2.3.3.1 Maintaining Receive Phase Continuity with Non-Zero Frequency Offset

Whenever a node receives a packet that is not frequency synchronized to its own oscillator, it must make an adjustment to the phase estimate that has been made. This adjustment should remove any anticipated phase change due to estimated frequency offset between the two nodes. This ensures continuity between phase estimates made in different packet periods (i.e., the phase estimates are being made in the same frame of reference). Equation 2.1 shows the process of accounting for this factor by giving the approximate "non-frequency" phase change. ϕ_{meas} is the estimated phase offset for a given packet, $\hat{f}$ is the filtered frequency estimate for a given packet, and T_p is the time since the last packet was received.

$$\Delta\phi_{nf} = \phi_{meas}[k] - (\phi_{meas}[k-1] + 2\pi\hat{f}T_p) \tag{2.1}$$

When considering Fig. 1.1, the nodes $\{\mathrm{TX}_2 ... \mathrm{TX}_N\}$ should apply this process to all the packets they receive. This is because all packets sent within the array will be synchronized to the TX_1 node clock (in general non-zero frequency offset at slave nodes) and all signals sent from external nodes, which the array listens to in an opportunistic way, will be unsynchronized.

2.3.3.2 Transmit Frequency Compensation

Whenever a slave node transmits a packet, it must apply a frequency correction factor. This process is seemingly straightforward, the outgoing packet can be multiplied by a complex sinusoid with a frequency that ensures 0 Hz offset at the master node. However, an additional complexity is introduced when a requirement to maintain phase continuity between packet periods is imposed. In order to meet this requirement, an additional phase continuity term must be added to the complex sinusoid according to equation 2.2 where T_p is the time since the last packet was transmitted.

$$\phi_{cont}[k] = 2\pi \hat{f} T_p + \phi_{cont}[k-1] \tag{2.2}$$

This phase continuity adjustment will ensure that packets are received by all nodes without phase discontinuities from one packet to the next. In essence, we are ensuring that packets arrive at all nodes with the same frame of reference. The frame of reference shown here is one that ensures zero frequency offset at the master node, but any frame of reference that is common to all co-operating nodes is acceptable.

2.3.4 A Rule of Thumb for Required Synchronization Accuracy

In order for pre-synchronization to support distributed beamforming, phase error between nodes can be quite large; as discussed in [25] phase error of 35° will result in a beamforming gain of 81%. Frequency synchronization among nodes needs to be precise because it affects the level of phase synchronization that can be achieved. For instance, a 2 Hz frequency error in a 50 ms packet period will result in a 36° phase

error. We use this as an approximate benchmark for the required level of synchronization accuracy for a viable distributed array.

2.4 Experimental Results

2.4.1 Round-Trip Frequency Accuracy and Phase Stability

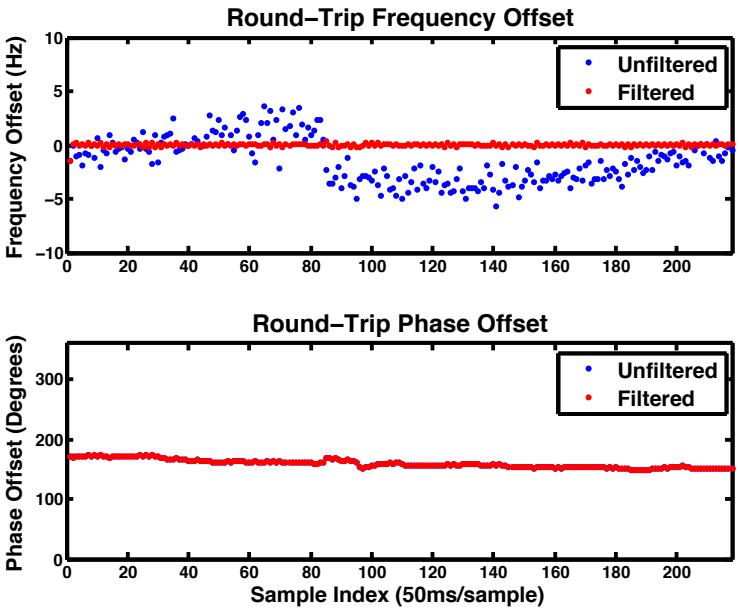

Figure 2.9: Round-trip frequency and phase synchronization over time for one slave and one master.

In order to verify that a single Master and single Slave can properly synchronize, experiments were run with no beamforming timeslot and with only one Slave active. The objective of these experiments is to ensure that round-trip frequency synchronization and phase stability will maintain the levels calculated in 2.3.4. Fig.

2.9 shows one set of results from these experiments where the average filtered frequency offset is 0.01Hz and the round-trip phase offset is stable to 1.6°/s. This test far exceeds the minimal benchmark described earlier. Several repeated trials of this experiments produced a typical frequency synchronization error of less than 0.05 Hz and the round-trip phase offset is stable to within 2°/s over 30 seconds or so.

2.4.2 Verifying Round-Trip Synchronization

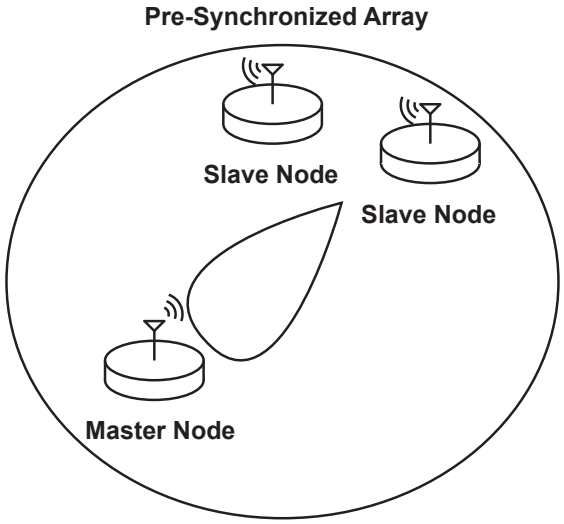

Figure 2.10: The experimental setup includes one master and two slaves. The array undergoes synchronization once during each epoch. The slave nodes form a beam to the master to test the precision of the synchronization.

A series of experiments were also run with both Slaves active and a beamforming timeslot active. The idea behind this set of experiments is that when the

Slave nodes are fully pre-synchronized, their joint transmissions will arrive coherently at the Master node. In other words, a fully pre-synchronized array of Slave nodes can achieve distributed beamforming at the Master node. Thus by observing the beamforming gain at the Master node, we can verify the level of pre-synchronization achieved by the Slave nodes.

Figure 2.11: An abbreviated timeslot diagram for a system with no external target.

Over the course of a full experiment, average beamforming gain at the Master node was typically greater than 90%. Fig. 2.12 shows a series of 4 received packets at the Master node during a beamforming experiment. These packets correspond to the packets shown in Fig. **??**, where the Master broadcast packet is the final packet shown in the figure. The Master broadcast packet is shown with very low amplitude because the packet energy is being detected through the isolation of the nodes antenna switch. It is clear that the amplitude of the beamforming packet is approximately equal to the sum of the amplitude of the two Slave packets that precede it. In fact, the average amplitude of the Slave 1 packet shown is .05, the average amplitude of the Slave 2 packet shown is 0.09, and the average amplitude of the beamforming packet shown is

0.14; so the beamforming gain of this particular sequence of packets is 97.5%.

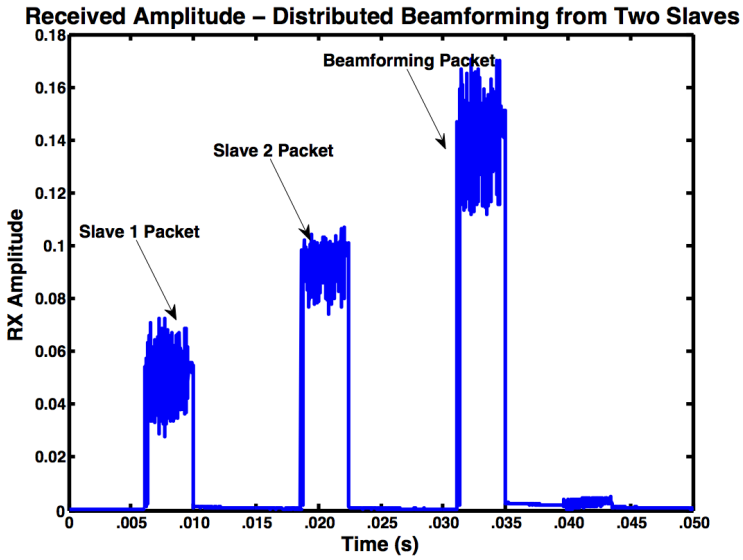

Figure 2.12: Series of received packets at the master node for an experiment verifying round-trip synchronization.

In Fig. 2.13, another experiment is shown. In this case, the plots show how the beamforming gain changes over the course of an experiment. There are a few places within the experiment where the beamforming gain drops temporarily, but the system quickly locks back into sync. Initially, the EKF on the Slave 2 has not reached steady state and is producing phase estimates that are nearly 180°out of phase with the true value (shown in Fig. 2.14). This phase error corrects itself within less than 10 samples or .5s. At around sample 100 the OCXO appears to change frequency by about 10 Hz, this also causes the system to temporarily lose sync, this time for

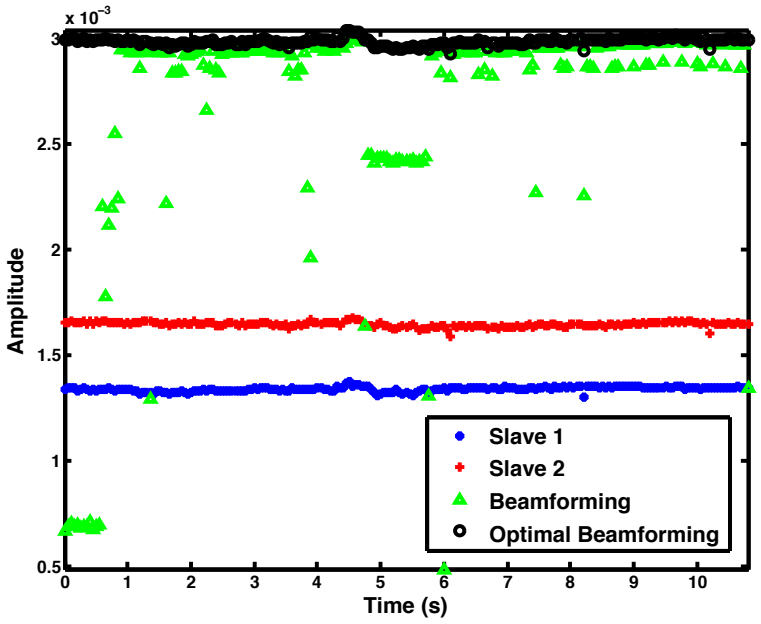

Figure 2.13: Achieved average beamforming amplitude at the master compared to the expected optimal beamforming amplitude based on the slave amplitudes of the most recently received packets.

about 15 epochs. Even with these disturbances, the system maintains an average beamforming gain of 90%.

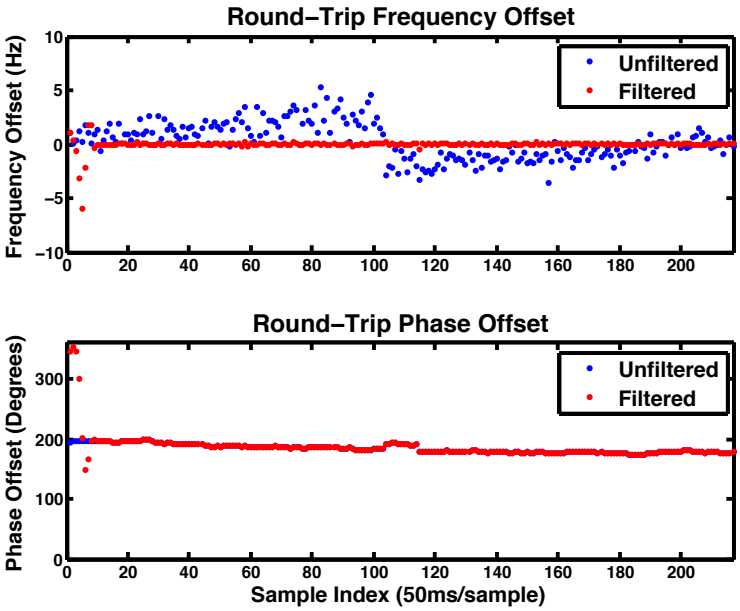

Figure 2.14: Round-trip frequency and phase offset for one of the slave nodes in an experiment verifying synchronization.

2.4.3 Retro-directive Beamforming

Fig. 2.17 shows one epoch of received packets at the target node during the experiment. The complete series of packets referenced in Fig. 2.16 can be seen in this plot. The first three packets represent the received energy during the master, slave 1 and slave 2 synchronization time slots respectively. As a reminder, the target does not process these packets in any way. The only response the target makes to receiving

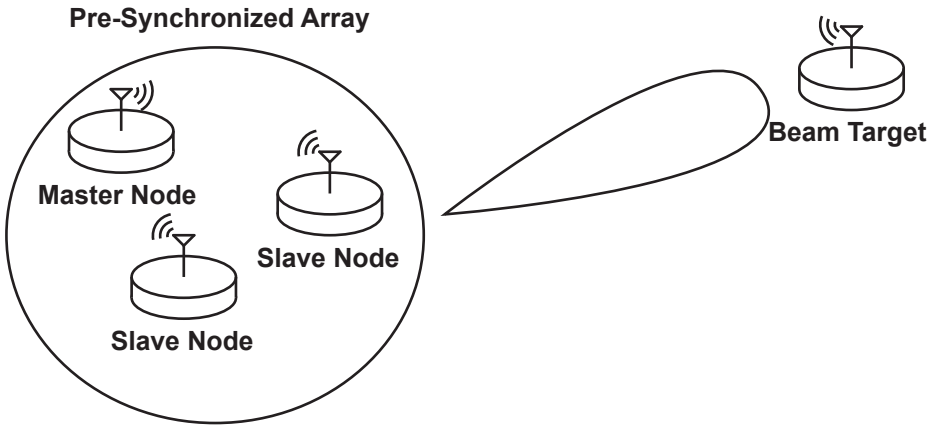

Figure 2.15: The experimental setup includes one master, two slaves and one target. The target periodically sends a message with a known preamble, but no explicit feedback is provided to the array. The array undergoes synchronization with itself once during each epoch.

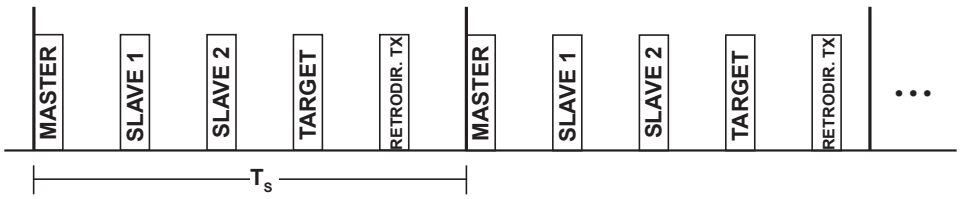

Figure 2.16: A timeslot diagram showing the periodic exchange of messages in the experimental setup.

the master packet is to schedule its own transmission. This packet transmission can be observed around 75 ms, where a small amount of energy is detected by the target receiver through the isolation of the its own antenna switch hardware. Finally, the joint packet is observed with approximately 98% beamforming gain.

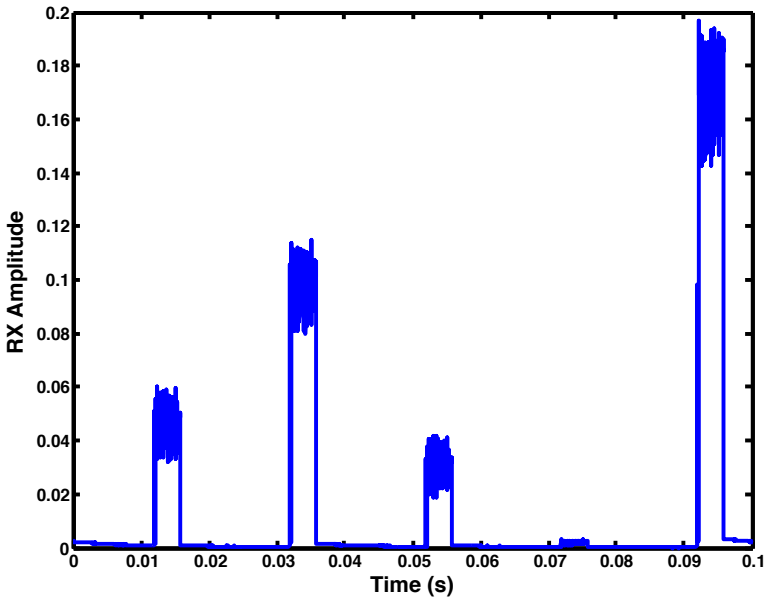

Figure 2.17: A series of packets received by the target node during one epoch. The first packet is the master packet, the second is the slave packet and the third is the beamforming packet.

Fig. 2.18 shows the average amplitude of the QPSK symbols in each packet received during the course of the experiment as well as the estimated maximum beamforming gain during each timeslot. The estimated maximum beamforming gain is computed using the most recently received packet from each node in the array. To

account for dropped packets, this approximation allows for included packets to be up to 1 second old. With this calculation, the experiment has an average beamforming gain of over 90%. In fact, the beamforming gain is much higher when the array is in steady state, with an average beamforming gain of over 97%.

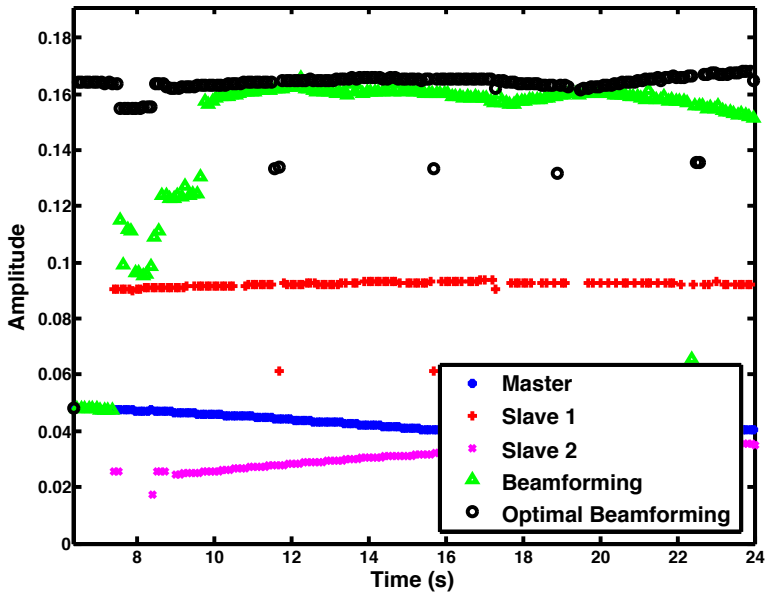

Figure 2.18: Achieved average beamforming amplitude at the target compared to the expected optimal beamforming amplitude based on the master and slave amplitudes of the most recently received packets.

2.4.4 Nullforming

In the fully wireless experiments, Fig. 2.21 shows a sequence of packets representing two epochs as observed at the receiver after the algorithm has reached steady

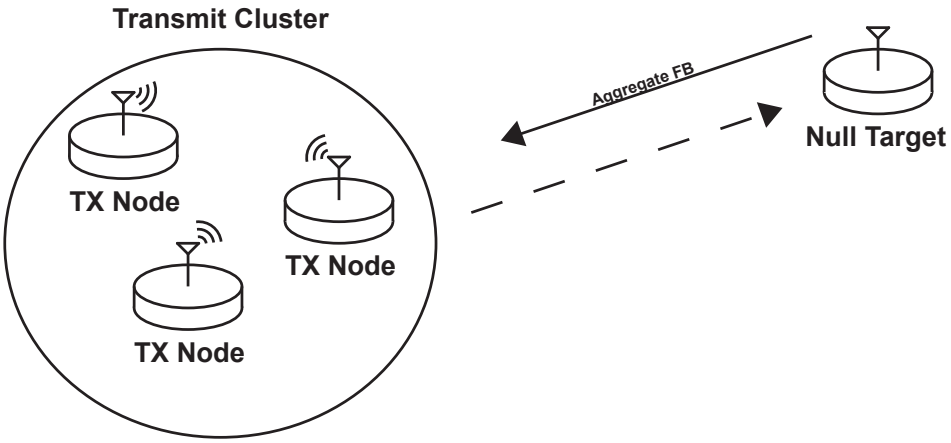

Figure 2.19: The experimental setup includes an array of three transmitter nodes and a target node. The target periodically sends a message with an aggregate feedback term and channel state information.

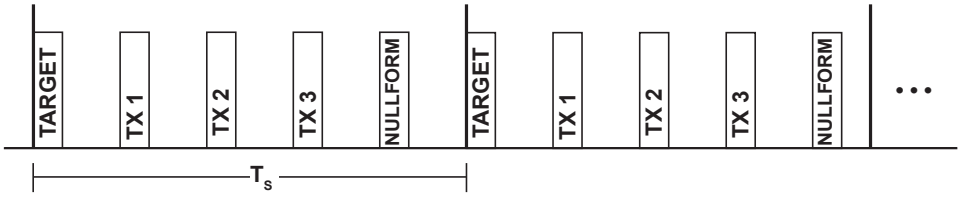

Figure 2.20: A timeslot diagram showing the periodic exchange of messages in the experimental setup.

state. During each epochs, there are five packets: the first being the receiver's own transmission of the feedback message as observed over the isolation of its antenna switch, the next three packets training messages from each of the transmitters, and finally, the fifth packet being the joint transmission from the array.

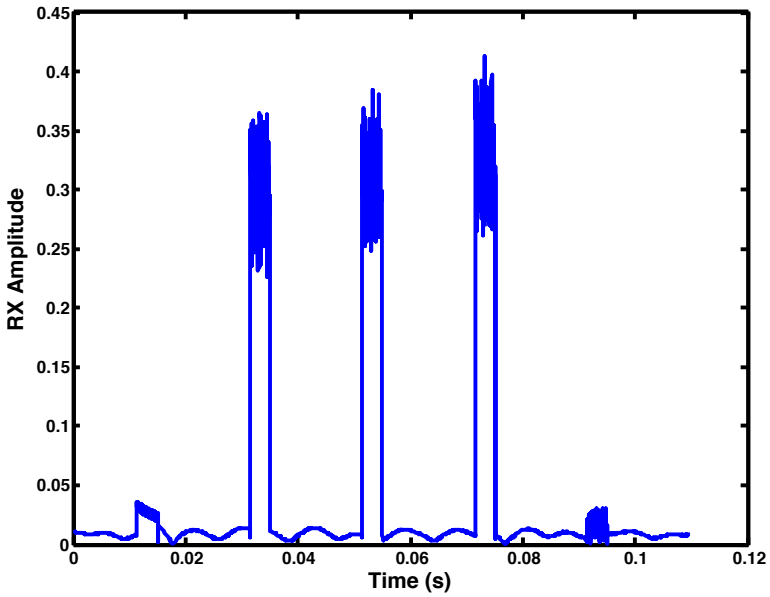

Figure 2.21: A series of packets received by the target node during one epoch. The first packet is the target's own packet, followed by three individual packets from the members of the transmit array, and finally the nullforming packet.

Fig. 2.22 shows the signal strength of the individual transmitters and the nullforming signal in each epoch over a 35 second experiment run; also shown is an "incoherent power level" that is inferred from the signal strengths of the individual transmitters; we see that the nullforming algorithm converges to an amplitude $20-25$

dB lower as compared to the incoherent power level in this experiment.

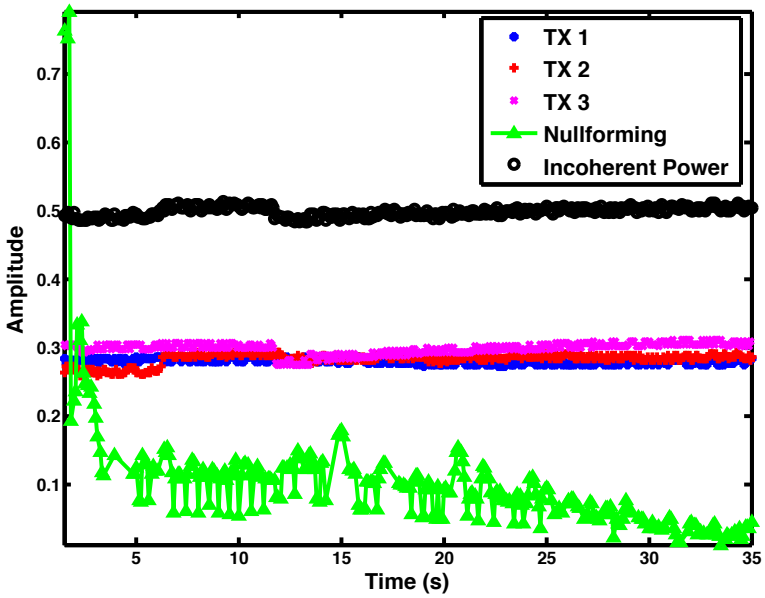

Figure 2.22: Achieved average nullforming amplitude at the target compared to the expected incoherent amplitude based on the amplitudes of the most recently received packets.

2.5 Chapter Summary

During this chapter, we have presented an architecture to take advantage of retrodirective beamforming techniques. we have also presented a discussion of the challenges that are present when implementing this architecture on an off-the-shelf SDR with open-source software. Finally, we presented some experimental results that show the architecture works in practice. In the following chapters, we discuss

how to enable additional retrodirective techniques and enhance scalability of existing systems.

CHAPTER 3
ADVANCED METHODS IN ARRAY SYNCHRONIZATION

So far, we have shown the capability of an array to synchronize and co-operate to form beams and nulls to a target. A major contribution of this thesis is the enhancement of this capability by increasing both robustness and scalability. In this chapter, we offer algorithms to improve both of these during the synchronization process. Not surprisingly, increasing robustness has a negative impact on scalability, and the contrary is also true. We show the trade-offs between these two and allow for design decisions to be made, depending on a particular use case. First, we discuss the topic of *aggregate feedback*-based synchronization and then we discuss using *multiple masters* for synchronization.

3.1 Aggregate Feedback for Array Pre-Synchronization

Our existing method for synchronization depends on a 1-to-1 exchange of packets - the master sends a packet and then each slave responds in an assigned timeslot. For an array of size N, N packets must be transmitted to achieve initial synchronization (i.e., $N_{packets} = N$). As changes occur, an additional N packets must be transmitted for the array to adapt to those changes. If either the packet size or the timeslot structure of the MAC is limited, the number of nodes in the array is automatically limited.

As discussed in the introductory chapter, aggregate feedback methods for transmit beamforming can be used to approximately learn ones own channel to the

target. As shown in Fig. 3.1 we can adjust this method very simply to fit a retrodirective scheme by directing transmissions to the master rather than the target. Each node will learn its relative channel to the target after observing h_{Ti} (channel from target to node i) and h_{mi} (channel from master to node i) and learning h_{im} through aggregate feedback.

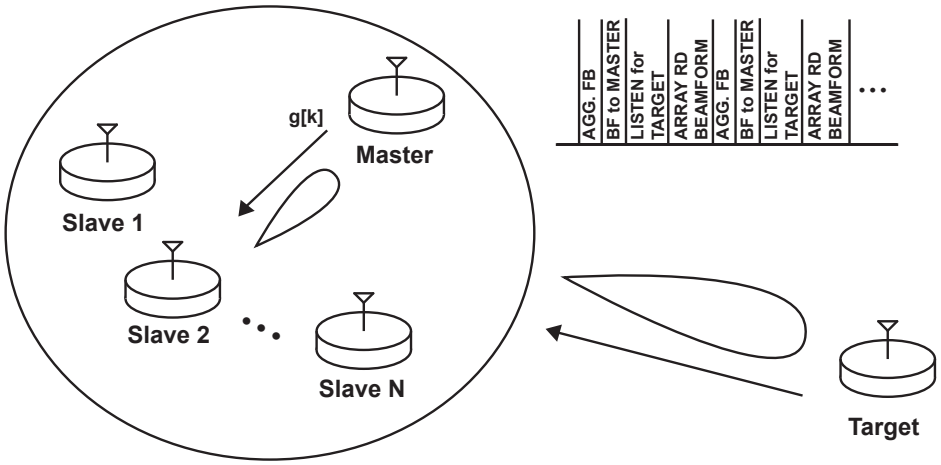

Figure 3.1: A group of slaves using an aggregate feedback beamforming to learn their channels to the master node. The master provides feedback $g[k]$. The target node does not provide any explicit feedback, instead members of the array listen opportunistically for transmissions in order to learn their relative channels to the target.

While previous aggregate feedback schemes have been discussed in [27] and [17], they do not converge to a precise channel estimate quickly. These methods rely on statistics to converge to correct channel estimates; as shown in Fig. 3.2, for

an array of size N, more than N timeslots of 2 packets must be exchanged (i.e., $N_{packets} \gg 2N$). They do have some advantages though: they do not restrict the number of users and do not require any handshaking for a node to join the network. The drawback is that time to synchronization grows quickly as the number of users grows. In order to control the time to synchronization, we suggest a scheme that has performance guarantees, scaling linearly with the number of users.

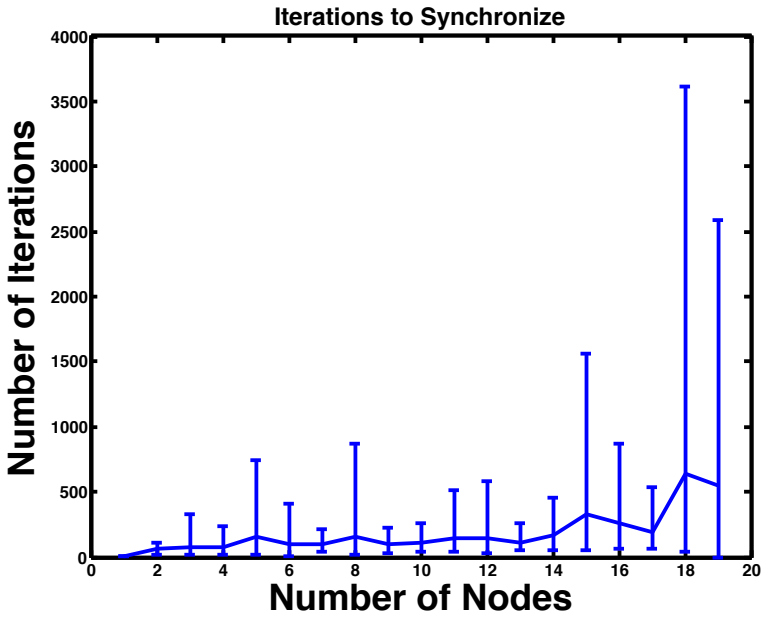

Figure 3.2: A plot showing how the number of iterations required for phase synchronization changes with the number of nodes using LLN aggregate feedback, averaged over 20 trials. The maximum and minimum are also shown.

3.1.1 Methodology: Periodic Basis Functions

The aggregate feedback system is, by its very nature, a system identification problem. If N nodes are present in an array, at least N pieces of feedback will be required for estimation. Previous algorithms, including the 1-bit feedback and LLN feedback algorithms, have been shown to require much more than this. If any node could know the pre-coding applied by all other nodes, it could estimate its channel in *exactly* N steps. If we weaken this requirement slightly, we can say that if each node knows all other nodes apply pre-coding chosen from a set of P basis functions, where $P > N$, each node can estimate the channel of every other node in P steps. When the array assigns basis functions efficiently, i.e., $N \to P$, the channel learning process is nearly optimal. We will refer to this as the basis function method for aggregate feedback.

In this method, after the initial channel has been learned, each node can update its channel estimates after every packet exchange by using the most recent P observations.

By making these basis functions periodic, one could also potentially design basis functions that allow some nodes to determine their channels prior to P pieces of feedback. For instance, imagine the function $z(A_i, k) = 1 + cos(2\pi k \frac{1}{A_i})$ is used as a basis function with $P = 6, A_i = 2, 4, 8, 16, 32, 64$. Due to the co-periodic nature of the basis functions, a node with basis function $z(A_1, k)$ learns its channel in 2-steps, $z(A_2, k)$ learns its channel in 3-steps and so-on. For this particular method of choosing basis functions, the average amount of feedback required is $\frac{1}{P}(P + \sum_{i=1}^{P-1} i + 1) < P$.

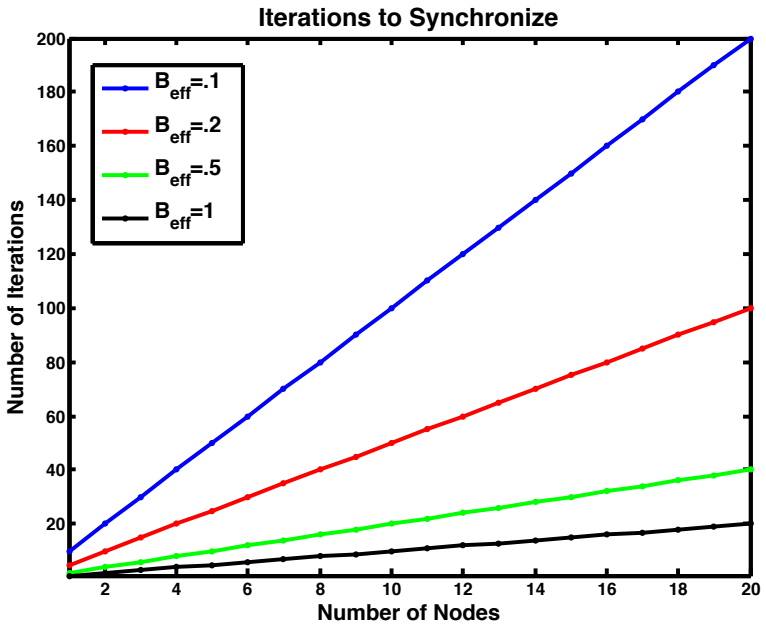

Figure 3.3: A plot showing how the number of iterations required for phase synchronization changes with the number of nodes when using our method. B_{eff} varying efficiencies in the assignment of basis functions.

3.1.1.1 Mis-assigned Basis Functions

If the same basis function is assigned to two different nodes a collision will occur during synchronization. The array as a whole is very robust to these collisions: only the two nodes with the mis-assignment are affected; all non-collision nodes should be able to continue as though nothing is wrong. The collision nodes will learn their channels incorrectly. If basis function reassignment occurs frequently, nodes should be able to identify that a collision is possibly affecting them and temporarily discontinue their participation in retrodirective MIMO activities. This detection can be done by observing a large discontinuity in channel estimate immediately after a change of basis functions.

This method for collision detection and avoidance suggests that, in the absence of a better mechanism for basis function assignment, basis functions could be chosen randomly and then frequently re-chosen. Nodes that choose a function that places them in a collision state would hold on to their previous estimate until functions are reassigned.

3.1.2 System Architecture and Assumptions

3.1.2.1 Key Concepts

- **Existence of Basis Functions**: Nodes will have apriori knowledge of a pool of P unique basis functions.
- **Array Size**: Arrays will be limited to a maximum of Q users, where Q is a number small enough that each node can choose a basis function P in a way

that is guaranteed not to collide with other nodes choice of basis functions.

- **Choice of Basis Functions**: Ideally, nodes choose a basis function in a way that is guaranteed not to collide with other nodes choice of basis functions, but collisions will not disturb the overall array.
- **Channel Stability**: Transmit channels are assumed to be constant relative to the period of synchronization. It may be necessary for synchronization to occur frequently.
- **Basis Function Seed/Index**: Nodes will be guaranteed to use the same index, k, during any particular timeslot.

3.1.2.2 System Architecture

A master sends a packet periodically. Each of N slaves, where $N \leq Q$, use the master packet to time a response. These nodes simultaneously transmit a packet with precoding basis functions $z(A_i, k)$ over their complex channels $h_{im}(k)$. We assume that the channel gain is approximately constant over short time intervals, so $h_{im}(k) \approx h_{im}$. The precoding basis functions are chosen from a pool of size P. The master receives the signal $r(k) = \sum_{i=1}^{N} z(A_i, k)h_{im}$.

By collecting the most recent P signals $r(k)$, h_{im} can each be estimated directly.

3.1.3 Evaluating the Basis Function Method

This method has improves upon the LLN aggregate feedback method by requiring that nodes pull their pre-coders from a known set of basis functions. From

Fig. 3.2 and 3.3, we can see that for 18 slave nodes the LLN method takes more than 500 iterations to synchronize on average while the basis function method is guaranteed to synchronize in 180 iterations even with only 10% efficiency in assigning basis functions. Not only does the new method beat the average of the LLN method, but it provides a performance guarantee, a stark contrast to the large variance seen when using the LLN method for channel estimation.

The basis function method improves upon the explicit feedback method in terms of its simplicity. However, when comparing this method directly to the explicit feedback method used in our experimental setup from the previous chapter, the explicit feedback method is more efficient. For a system with 1000 slaves, the explicit feedback only needs to exchange 12 kB of data. Meanwhile, the basis function method needs to exchange 16 kB of data. In a Wi-Fi type system (where preambles and thus minimum packet sizes are larger), this overhead cost becomes larger, with an explicit feedback system requiring only 73 kB of data and the aggregate feedback system requiring 138 kB of data to obtain initial synchronization. As we stated earlier, this initial synchronization cost is higher for the aggregate feedback system, which will then require much less overhead in order to obtain new information, while the explicit feedback network will need to go through the full overhead of synchronization in order to obtain new information. Still, we are motivated to reduce this initial overhead cost. The next section explores a method that will both reduce the overhead cost of initial synchronization, as well as increase the robustness of our distributed arrays.

3.2 Using Multiple Sources of Information for Antenna Array Pre-Synchronization

Up to this point, we have considered only systems where one master is present. This inherently lacks robustness. If the single master node is eliminated, the entire array falls apart. Another problem to consider is the hardware that might be available to nodes in large DMIMO arrays. It is possible that many *disadvantaged* nodes will be a part of the network (i.e., UAVs or Manpack radios). In this case, the disadvantaged nodes might act as slave nodes, but their transmissions might not be strong enough to reach a single, central master node. One could imagine a network where the small handful of able-bodied users act as master nodes while hundreds or even thousands of disadvantaged users act as slave nodes.

In this section, we establish a method for using multiple master nodes in an array to solve this problem. The key idea, illustrated in Fig. 3.4, is that one master acts as a primary master, master nodes synchronize among themselves, and slave nodes transmit in an aggregate arrangement as described in the previous section. If the primary master is lost, the other masters can go on as if nothing has changed.

3.2.1 Anticipated System Architecture and Assumptions

In the most general case, we consider a distributed MIMO system with N slave nodes and M master nodes. At any given time, one of the master nodes will be the *Active Master* and all other master nodes will be *Secondary Masters*.

As in previous sections, no specific physical arrangement of nodes is required

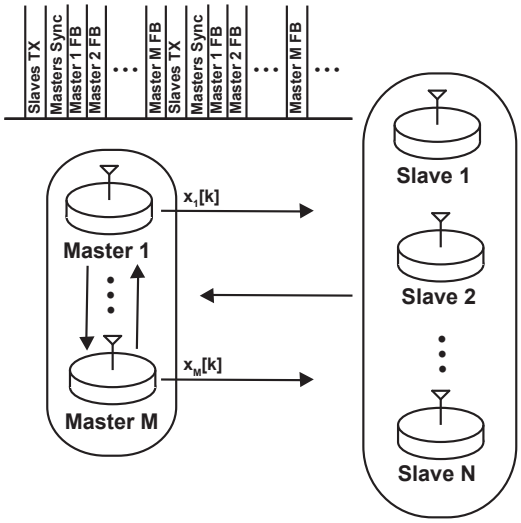

Figure 3.4: An array organized with a N slaves and M masters.

and all nodes are fully-wireless.

Initially, we assume that the secondary master nodes synchronize to the active master node using an explicit feedback system like the one described in [31].

3.2.2 Algorithm Design

Within this section, we describe each step of the algorithm and its logic.

3.2.2.1 Slave Joint Transmit

Slave nodes simultaneously send a transmission with a pre-coder z_i where i is the index of the slave node.

3.2.2.2 Active/Secondary Sync

The active and secondary master nodes perform synchronization. This could be using the method in [31] or some other method.

3.2.2.3 Secondary Feedback Adjustment

The secondary master nodes adjust their feedback.

3.2.2.4 Aggregate Feedback

The master nodes each send aggregate feedback to the slaves.

3.2.2.5 Slave Feedback Adjustment

Slave nodes adjust the feedback that they have received.

3.2.3 Proof

Here, we show a proof that feedback from different masters can be used interchangeably at a particular slave node.

The M master nodes will be indexed by j in $\{0...M-1\}$, and the N slave nodes in the array will be indexed by i in $\{M...M+N-1\}$. Without loss of generality, we choose a particular master node, $j = 0$, as the active master.

After the slave joint transmit, each master node will have an estimate of the complex channel from the joint transmission:

$$y_j = \sum_{k=M}^{M+N-1} z_k h_{kj} \tag{3.1}$$

After a round trip synchronization, each master will have access to the complex channel estimates $\hat{h}_{0j}$ and $\hat{h}_{j0}$. The masters can now make an adjustment to the joint complex channel estimate y_j:

$$x_j = \frac{\hat{h}_{j0}}{\hat{h}_{0j}} y_j = \frac{\hat{h}_{j0}}{\hat{h}_{0j}} \sum_{k=M}^{M+N-1} z_k h_{kj} \tag{3.2}$$

The master nodes can now send x_j as feedback to the slaves, who will learn complex channels h_{ji} upon receipt of this feedback. Using this feedback, the slave nodes can calculate an adjusted feedback:

$$v_{ji} = \frac{1}{\hat{h}_{ji}} x_j = \frac{\hat{h}_{j0}}{\hat{h}_{0j}\hat{h}_{ji}} \sum_{k=M}^{M+N-1} z_k h_{kj} = \frac{z_i h_{ij} \hat{h}_{j0}}{\hat{h}_{ji}\hat{h}_{0j}} + w_{ij} \approx \frac{z_i T_i c_i^2 R_0}{T_0 c_0^2 R_i} + w_{ij} \tag{3.3}$$

We can remove z_i because it is a known pre-coder and over time we can remove the noise term w_{ij} through filtering. Because this quantity is really a measure of the approximate channel from 0 to i using estimates from j, we change notation slightly.:

$$u_{0i}^{(j)} = z_i / v_{ji} \approx \frac{T_0 c_0^2 R_i}{T_i c_i^2 R_0} \tag{3.4}$$

If every slave node wanted to do phase-only beamforming to an external target node, t, this could be achieved by listening to that node and transmitting using $u_{0i}^{(j)}$ and $\hat{h}_{ti}$ as an adjustment factor. The complex gain h_{it} is picked up during transmission:

$$u_{0i}^{(j)} \frac{h_{it}}{\hat{h}_{ti}} \approx \frac{T_0 c_0^2 R_i}{T_i c_i^2 R_0} \frac{T_i c_i g_{it} c_t^{-1} R_t}{T_t c_t g_{ti} c_i^{-1} R_i} = \frac{T_0 c_0^2 R_i}{T_i c_i^2 R_0} \frac{T_i c_i^2 R_t}{T_t c_t^2 R_i} = \frac{T_0 c_0^2 R_t}{T_t c_t^2 R_0} \tag{3.5}$$

Notice that the selected phase-only beamforming pre-coder results in a quantity that is dependent on neither i nor j, but is instead only dependent on the active master and the target, regardless of where the estimate $u_{0i}^{(j)}$ has come from. Further, notice that the estimate $u_{0i}^{(j)}$ is the same for any value of j, with the exception of the noise term, w_{ij} (which has been negated by filtering).

3.2.4 Implications

3.2.4.1 Robustness

Due to the fact that, less the noise term, $u_{0i}^{(j)}$ is not dependent on j, each slave node only needs to be within range of one master and that master need not be the same between various slaves. Each secondary master must be able to reach the primary master during active/secondary synchronization. Therefore, this topology is a good fit for systems with some users who are powerful and others who are disadvantaged, as described in the introduction.

This type of system is also robust for another reason. If master node 0 becomes unavailable, master node 1 can take over the role of the primary master. Since slave nodes have spent time learning the $u_{0i}^{(j)}$ synchronization, can they quickly transition to a $u_{1i}^{(j)}$ synchronization?

On the first step of node 1 synchronization, nodes will receive:

$$\bar{x}_j = \frac{\hat{h}_{j1}}{\hat{h}_{1j}} y_j = \frac{\hat{h}_{j1}}{\hat{h}_{1j}} \sum_{k=M}^{M+N} z_k h_{kj} \tag{3.6}$$

This term can be compared directly to priors by converting:

$$x_j = \frac{\hat{h}_{j0}}{\hat{h}_{0j}} y_j \approx \frac{\hat{h}_{10}}{\hat{h}_{01}} \bar{x}_j \tag{3.7}$$

Thus, the new active master can send the term $\frac{\hat{h}_{10}}{\hat{h}_{01}}$ to all secondary masters when the change of masters occurs. These nodes can now use this adjustment factor for sending all future feedback. If there is some estimation error in the terms $\hat{h}_{10}$ and $\hat{h}_{01}$, it will cause some initial disturbance in the channel estimates of the slaves, but over time this disturbance will be removed from the system because the error will be

constant (and they are now learning their channels relative to 1 rather than 0).

3.2.4.2 Scalability

We have already established that using an aggregate feedback system is one way to increase scalability. Having a single synchronization timeslot shared by all slave users allows users to easily join and leave the network. In [17] we established that the number of iterations required to learn the channel in an aggregate feedback system is proportional to the number of masters. By showing that the noise term w_{ij} is not identical across j, we prove that the number of iterations can be decreased by a factor of the number of reachable masters, M^*, at any slave. Without loss of generality, we consider the case of the slave where $i = M$:

$$w_{Mj} = \frac{\hat{h}_{j0}}{\hat{h}_{0j}} \sum_{k=M+1}^{M+N} z_k h_{kj} \approx \frac{T_j c_j g_{j0} c_0^{-1} R_0}{T_0 c_0 g_{0j} c_j^{-1} R_j} \sum_{k=1}^{N} z_k T_k c_k g_{kj} c_j^{-1} R_j = \frac{T_j c_j R_0}{T_0 c_0^2} \sum_{k=1}^{N} z_k T_k c_k g_{kj} \tag{3.8}$$

Notice that w_{Mj} is dependent on which master, j, the feedback has come from. For instance, during one iteration, the term T_1 has no relation to the term T_2. This leads to the conclusion that (at least for certain types of aggregate feedback systems) during a single iteration of the channel learning process, the feedback from two masters can be used as though it was received in two different iterations.

The channel estimates from multiple masters can be averaged. In some systems, it may make sense to combine the feedback from multiple masters through a straight arithmetic mean, while in others it might be preferred to perform a weighted

average based on RSSI or estimated SNR.

3.3 Simulation Results

3.3.1 Aggregate Feedback Using Basis Functions

We performed simulations on the basis functions aggregate feedback method in this chapter and compared it to the LLN method. In Fig. 3.5 we can see the effect of using basis functions for synchronization. Synchronization occurs precisely after the initial P steps, where P is the number of basis functions available. The plot does not show this initial period, because during this time we have no estimate of the channel. After P steps, the channel estimate is updated during each iteration (i.e., we have 1 new piece of information and $P-1$ old pieces of information). The simulation models LO frequency drift and channel phase changes (i.e., movement of nodes), and yet the phase estimation error is nearly 0 at all times. Fig. 3.6 provides a comparison of the LLN method for aggregate synchronization with the basis function method (shown in Fig. 3.5). These simulations were run with identical parameters and the same number of masters and slaves. The LLN method takes longer to converge. In running multiple trials of both simulations, it is clear that the basis function method provides much more consistent performance.

3.3.2 Using Multiple Masters for Synchronization

Our simulations also allowed us to see the effect of having multiple masters with which slaves could synchronize. The various data sets plotted in Fig. 3.7 show how feedback from multiple masters can be used to reduce the effects of noise from

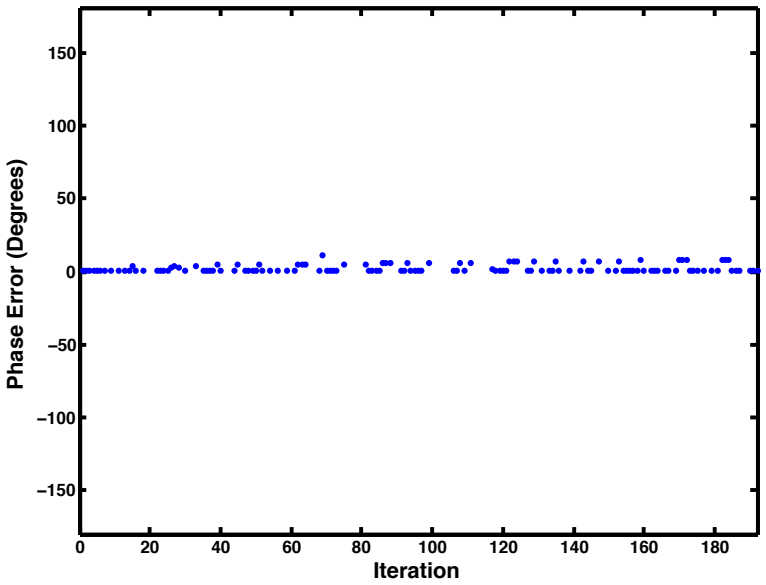

Figure 3.5: A plot showing the phase synchronization on an 8 slave system using an algebraic solver to decode basis functions. The plot is shown after the first 8 iterations.

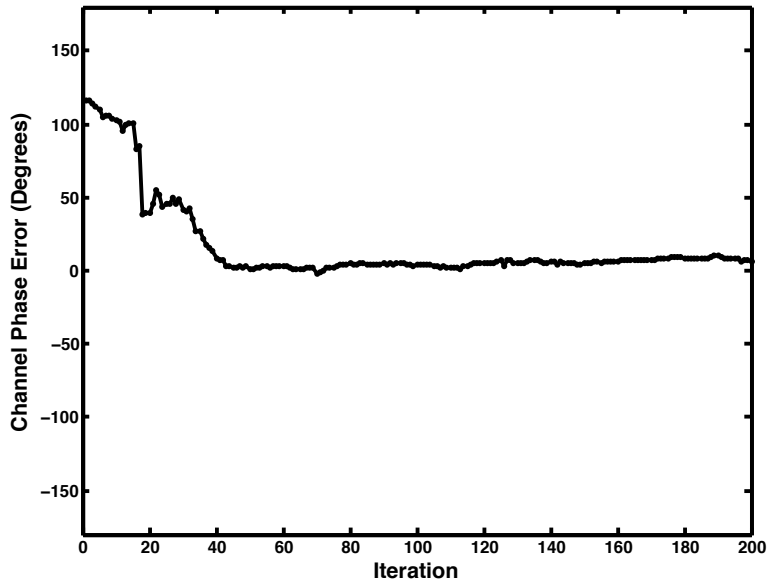

Figure 3.6: A plot showing the phase synchronization on an 8 slave system using the LLN aggregate feedback approach.

individual master feedback. In Fig. 3.8 we show the effect of using multiple masters on the LLN feedback method. This simulation contained 100 slave nodes and 20 master nodes. Three of the twenty independent channel estimates are shown in the plot; comparatively, the averaged estimate has less error, particularly early in the process. Later in the process, the averaged estimate is sometimes thrown off by outliers, but we could improve the performance by removing these outlier estimates from the data set. This plot indicates that a multiple master approach can reduce the number of iterations required for synchronization in aggregate feedback systems.

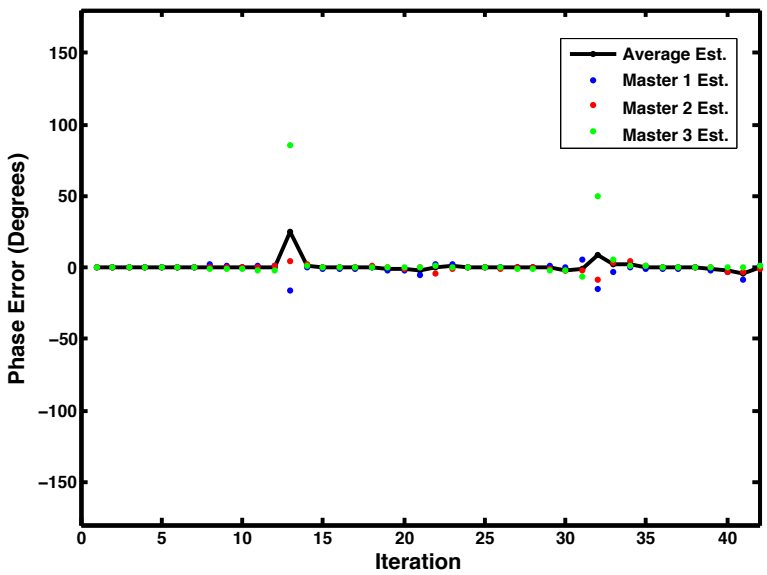

Figure 3.7: A plot showing the phase synchronization on an 8 slave system using an algebraic solver to decode basis functions. Three different masters are used for synchronization and are shown both independently and as an average. The plot is shown after the first 8 iterations.

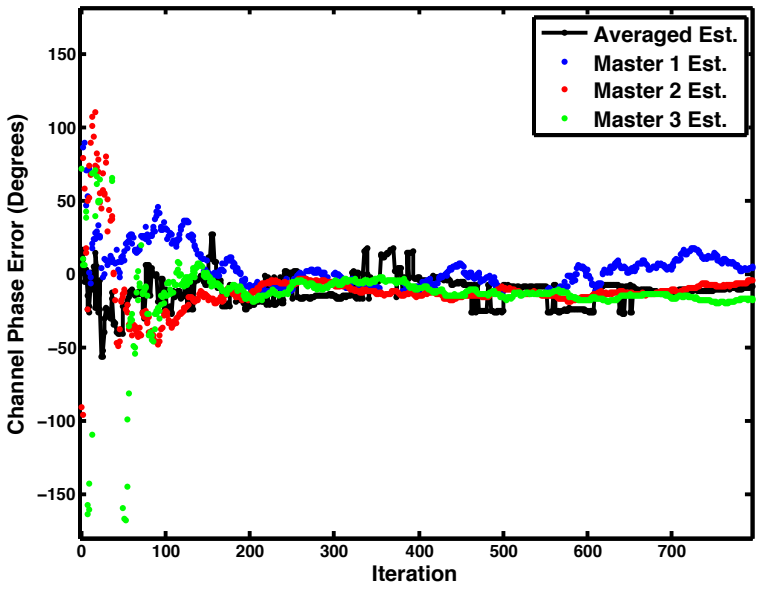

Figure 3.8: A plot showing the effect of applying averaging over estimates obtained from 20 masters with 100 slaves using the LLN feedback method.

3.3.3 Comparing Synchronization Overhead in Aggregate and Explicit Feedback Systems

We generated a series of plots to show the affect of using aggregate synchronization and of using more than one master in the synchronization process. In Fig. 3.9 we show that an aggregate synchronization system using the basis function methodology can approach the performance of an explicit feedback system when a large number of masters are used and basis functions are assigned efficiently. Fig. 3.10 shows how the number of packet transfers required for synchronization suffers when basis functions are not assigned efficiently. Notice that by using a modest number of

masters, performance improves rapidly. Adding more masters to the system does not improve performance as much, but certainly does add robustness.

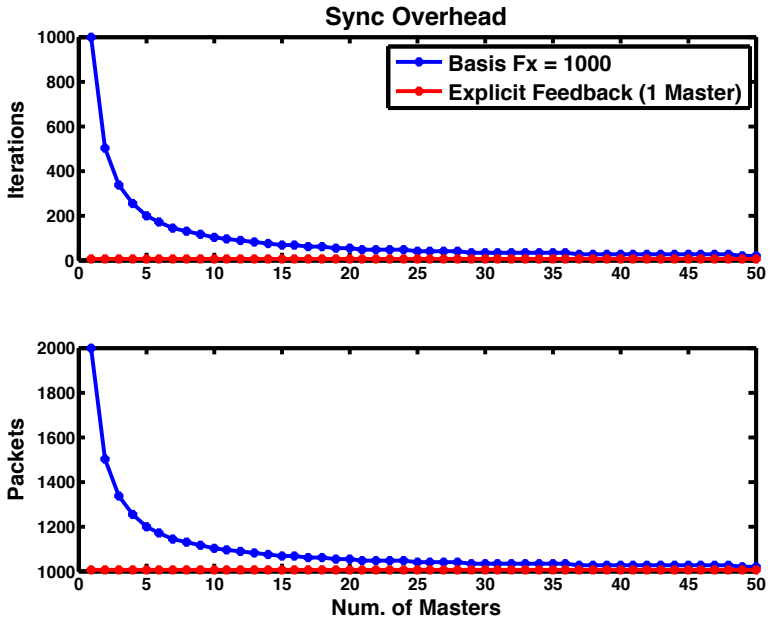

Figure 3.9: A plot showing the number of iterations that must be used and the number of packets transmitted in a 1000 slave system with efficient assignment of basis functions.

In Fig. 3.11 we illustrate how increasing the number of masters can actually increase the amount of data overhead required in systems where variable length packets can be used. This particular plot shows what happens when preamble sequences are short relative to synchronization payloads. The preamble and payload sizes are based on the experiments presented in this paper and [31, 30, 29]. Fig. 3.12 is based on longer preambles like the ones found in 802.11g WiFi, but while maintaining the

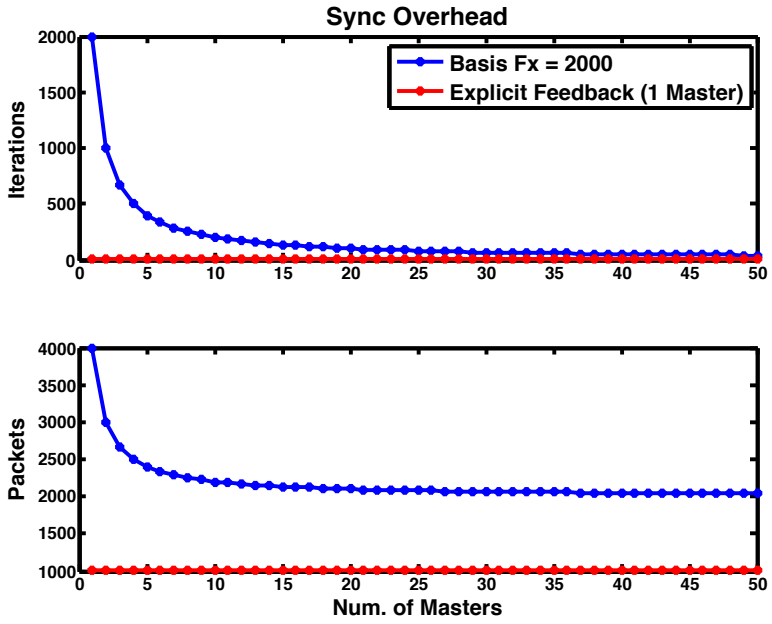

Figure 3.10: A plot showing the number of iterations that must be used and the number of packets transmitted in a 1000 slave system with inefficient assignment of basis functions.

payload sizes required by our experimental implementation.

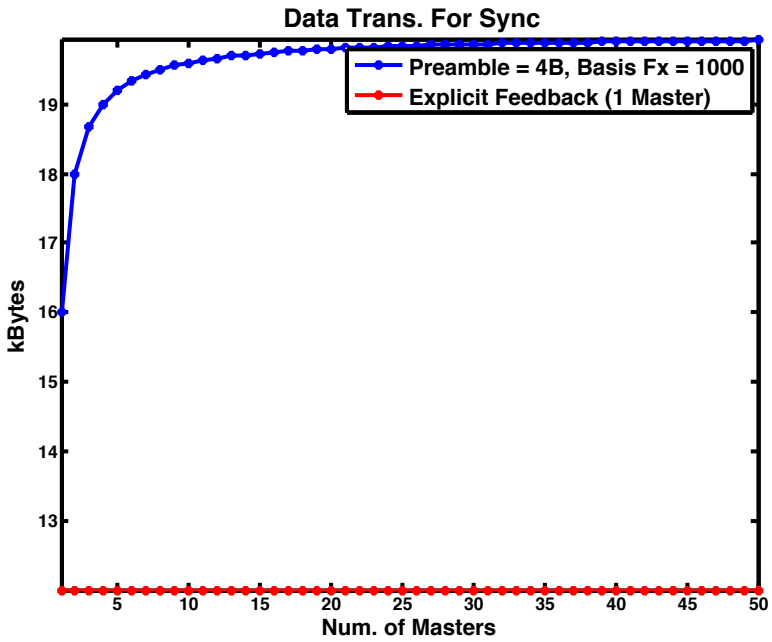

Figure 3.11: A plot showing the amount of data that must be transmitted in a 1000 slave system with efficient assignment of basis functions. This system has short preambles like the one in [31].

3.4 Chapter Summary

During this chapter, we presented two new ideas for distributed antenna array pre-synchronization. The first, provides for enhanced scalability by creating an aggregate feedback system with the fewest iterations possible. The cost of this system is creation of a prior set of basis functions and assignment of these functions to participating nodes. The second idea can be used to improve both robustness and

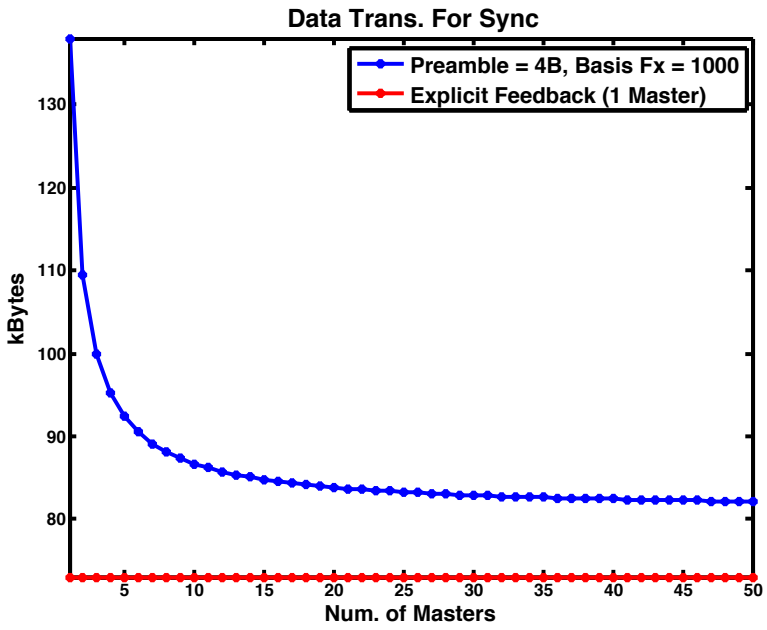

Figure 3.12: A plot showing the amount of data that must be transmitted in a 1000 slave system with efficient assignment of basis functions. This system has preambles comparable in length to an 802.11g WiFi system.

scalability of arrays. By having multiple nodes act in the role of master, the array has a level of redundancy if one master becomes disabled or is unreachable by certain slaves. If multiple masters are reachable by a slave node, then the number of iterations for synchronization can be reduced.

CHAPTER 4
APPLYING RETRODIRECTIVITY TO MORE COMPLEX MIMO OPERATIONS

In the previous chapters, we have introduced the concept of retrodirectivity and have shown an experimental proof-of-concept by applying retrodirective beamforming to distributed arrays. We have also shown experimental data for a nullforming experiment, which relies on explicit synchronization and aggregate feedback from the target node. Since beamforming and nullforming can be viewed as the basic building blocks of any MIMO system, we should ask the question: how can we create a robust method for distributed nullforming, that will not require explicit co-operation from a target? In this chapter, we answer this question, but in doing so, we create a much more general approach - creating a system wherein any MIMO technique that usually relies on aggregate feedback can be made to work without aggregate feedback from the target.

In general, our scheme is as follows: the array operates as described in previous chapters, listening opportunistically for transmissions from the targets. The slave nodes apply some pre-coding to make a test transmission at the master appear as if it is actually arriving at the target. We will show that the error of this co-ordination is small and that the scheme performs well under simulation.

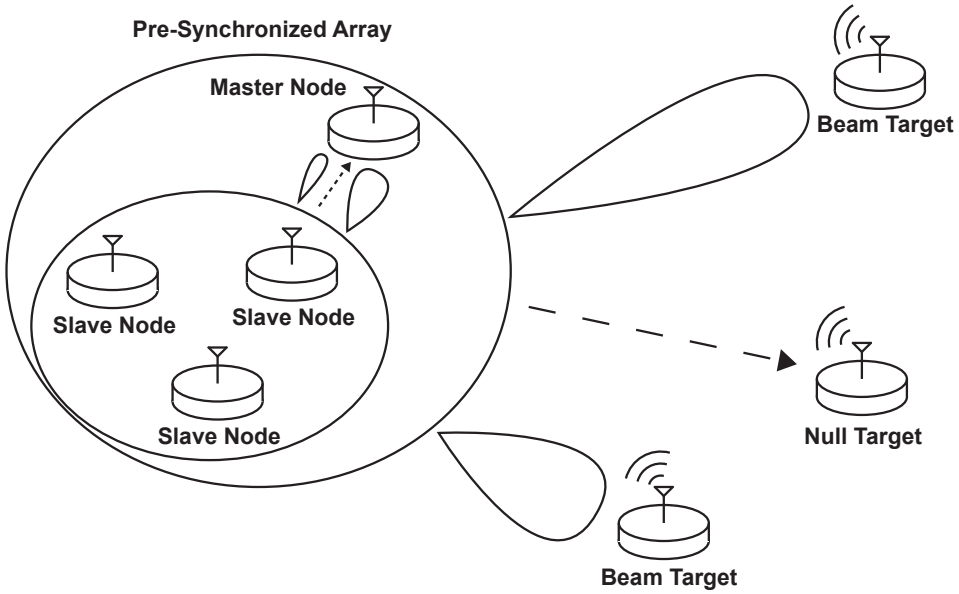

Figure 4.1: The concept system for a retrodirective nullforming system where the external target periodically transmits a message, which for simplicity is assumed to have a known preamble sequence, but does not provide any feedback to the array.

4.1 Anticipated System Architecture and Assumptions

In the most general case, we consider a distributed MIMO system with N nodes co-operating in an array and M external nodes. One node in the array is designated as a test node (denoted by m, for convenience this is also the master node).

As in our previous discussions, no specific physical arrangement of nodes is required and all nodes are fully-wireless.

We assume that each of the M targets make some intermittent transmissions that can be detected by members of the array, but as in prior chapters the transmissions may be unknown and do not need to be decoded.

The nodes in the array can perform the desired MIMO operations on the test node, who will provide all necessary feedback. Then, these operations can be transferred directly to the M target nodes using relative calibration concepts. This process can occur simply by emulating the channels of each target node while operating on the test node.

4.1.1 An Open-Loop Approach

Since we plan to use the master node as an analog for the external target nodes, it is perfectly reasonable to ask why it is necessary to use a feedback based approach for any MIMO operation. An alternative approach would be for all nodes in the array to send their CSI to the master node. The master node would then be able to calculate all of the necessary pre-coders and distributed them back out to the

individual slave nodes.

While this approach may work (it has not been simulated, but works in principle), we choose to use a feedback-based approach during this chapter. The feedback based approach allows for algorithms like the aggregate feedback phase-only nullforming and joint beam and null-forming algorithms in [20, 22] to be used. The reason for this choice is that these algorithms were designed to be highly scalable, and they fit well with the scalable/robust synchronization algorithms proposed in the previous chapter, where masters are unaware of the number of slaves that have synchronized to them. However, we will later show that these type of closed-loop algorithms act as a filtering mechanism that we consider imperative to our method for emulating unknown channels to determine proper pre-coders for target transmission.

4.2 Algorithm Design

Figure 4.2: A timeslot diagram describing the activity that must occur during each epoch.

Within this section, we describe each step of the algorithm and its rationale.

1. **Array Synchronization**

   The array will synchronize using the explicit feedback (or a functionally equivalent process). For convenience, the test node is the master node and the other nodes are the slaves.

2. **Listen for Targets**

   Each node in the array will listen for incoming transmissions from external nodes. The array nodes do not necessarily need to know any portion of the incoming transmission, they just need to be able to detect when it occurs.

3. **Local Joint Transmissions**

   Using the information from the pre-synchronization process and the partial channel knowledge learned from receiving target transmissions, the array can make a series of joint transmissions to the test node. These transmissions are representative of the transmissions that will be made to the external targets.

4. **Local Feedback**

   The test node will provide feedback to the array.

5. **Joint Transmission to Targets**

   Now, the array can translate the joint transmission from test node to the target nodes.

4.3 Proof and Sensitivity Analysis

Here, we show a proof that the translation from local to external array transmission is possible. The test node will be denoted by m, the $N-1$ other nodes in

the array will be indexed by i, and the M target nodes by j.

After the synchronization process, each node, i, will have access to quantities $\hat{h}_{im}$, $\hat{h}_{mi}$, and $\hat{h}_{ji}$. A message $m(t)$ could be sent to each target without any precoding:

$$r_i(t) = m(t)h_{ij} = m(t)T_i\ c_i\ g_{ij}\ c_j^{-1}\ R_j \tag{4.1}$$

A message $m(t)$ could be sent to the test node without any precoding:

$$s_i(t) = m(t)h_{im} = m(t)T_i\ c_i\ g_{im}\ c_m^{-1}\ R_m \tag{4.2}$$

This message, $m(t)$, could instead be sent to the test node using a precoder $\frac{\hat{h}_{ji}}{\hat{h}_{mi}}$:

$$s_i'(t) = m(t)h_{im}\frac{\hat{h}_{ji}}{\hat{h}_{mi}} \approx m(t)T_i\ c_i\ c_m^{-2}\ R_m\ T_j\ c_j\ g_{ji}\ T_m^{-1} \tag{4.3}$$

Now we can show that the difference between the precoded test transmission and the transmission to any target node is not dependent on the node i:

$$\frac{s_i'(t)}{r_i(t)} = \frac{h_{im}}{h_{ij}}\frac{\hat{h}_{ji}}{\hat{h}_{mi}} \approx \frac{T_j\ c_j^2\ R_m}{T_m\ c_m^2\ R_j} \tag{4.4}$$

By applying the precoder, a transmission from i to m will be approximately the same as a transmission from i to j. This means that feedback from m to i will be approximately the same as if j sent feedback to i.

4.3.1 Sensitivity Analysis

From the above equations, it is clear that any error in the joint transmission to j will be a result of estimation error in h_{ji} and h_{mi}. We could perform an analysis

of how errors in these quantities would affect a particularly sensitive type of transmission, such as nullforming, to get an idea of how good our estimates need to be. This would allow a systematic design of estimators, preamble lengths, filter types, etc.

4.4 Simulation Results

In order to show the proposed algorithm performs well under various effects, we ran simulations of the phase-only nullforming algorithm with our pre-coding scheme. The pre-coding scheme is applied to allow a master node to send feedback to slave nodes, who make phase-only adaptations. The attempted null is applied to the external target node at the end of any iteration. All simulations were run with 20 nodes.

Fig. 4.3 shows the performance of the retrodirective scheme under noise-free conditions. The null power at the target follows the null power at the master node. In practice, they should be identical, but they are off slightly, possibly due to a numerical issue.

Fig. 4.4 introduces clock effects into the simulation. All nodes have independent local oscillators that drift and are running at different frequencies (i.e., 10 MHz $\pm$100 Hz). In this plot we once again see that the two null power trends track closely with one another. The clock offsets seem to have an effect on the phase-only nullforming algorithm, but not on how well we can transfer nulls to an external target.

Fig. 4.5 adds the effect of offset estimation error. In other words, nodes fail to precisely estimate phase and amplitude of received signals. In this plot, we note

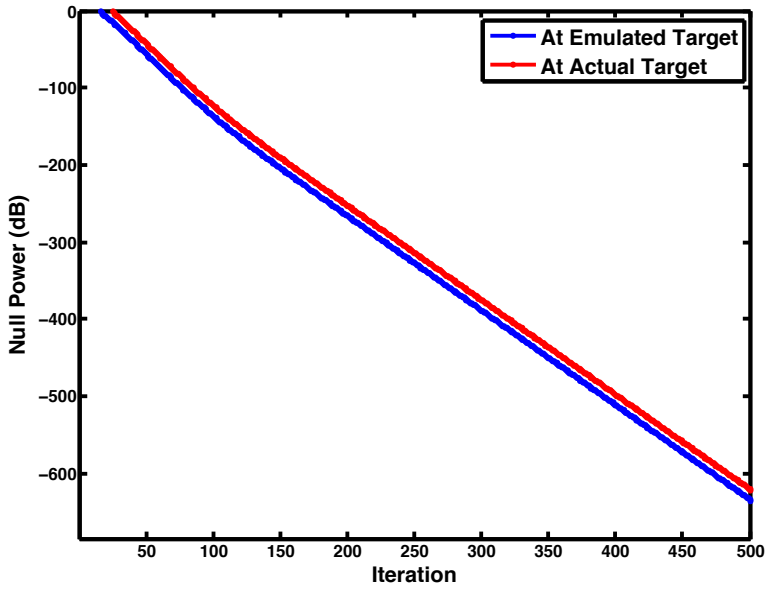

Figure 4.3: A plot showing signal suppression at a single target without any clock effects, channel dynamics, or channel estimation error.

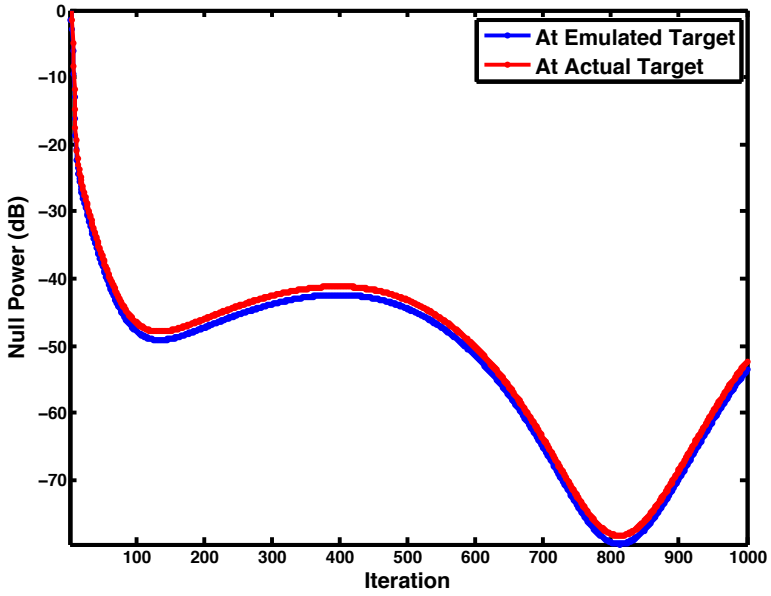

Figure 4.4: A plot showing signal suppression at a single target with LO offset, LO drift, and channel dynamics.

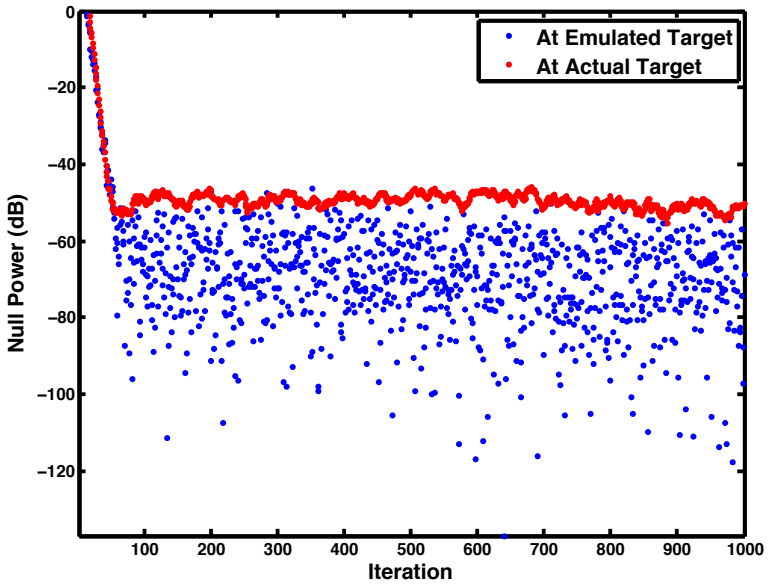

Figure 4.5: A plot showing signal suppression at a single target with channel estimation error in addition to LO offset, LO drift, channel dynamics.

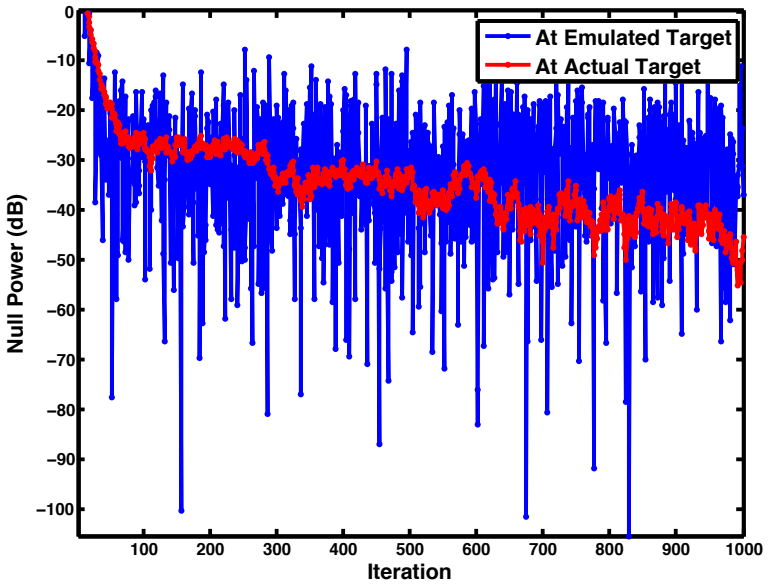

Figure 4.6: A plot showing signal suppression at a single target with even larger channel estimation error.

that the local null is suppressed more deeply than the external null, but also has greater volatility. This is not surprising, because there are actually two estimated terms included in the precoder for local messages ($\hat{h}_{ji}$ and $\hat{h}_{mi}$) , but there aren't any estimated terms included in the precoder for the external messages, instead only the feedback algorithm output (i.e., phase-only nullforming adjustment is included).

Fig. 4.6 has been included to show another example of the effect of offset estimation error. In this case, offset estimation error is slightly larger than before, but now the emulated target null is sometimes worse than the one at the actual target. This is due to the fact that the phase-only nullforming algorithm can be set up to adapt slowly, while the estimation errors happen on a timeslot-to-timeslot basis. This means that the null at the actual target is less affected by these variations, as the phase-only adaptations act like a filter.

This final plot also speaks to the earlier point about whether or not it is viable to use open-loop MIMO solutions with this emulated target method. This plot suggests, that unless channel estimation is very precise, it is wise to use a feedback-based method. These methods will act as an additional layer of filtering to prevent noisy channel measurements from disrupting beams and nulls to external targets.

4.5 Chapter Summary

During this chapter, we presented an idea for enabling a vast array of MIMO techniques within the context of retrodirective arrays. We presented simulation results, showing that this method can be applied to enable retrodirective nullforming,

even in adverse conditions. We showed that using a feedback based algorithm with this method can be advantageous. Our method can be applied to a wide variety of problems, but is particularly interesting in application to information privacy. When a user wants to send high-power transmissions to an external node, but does not want other nodes to be able to determine its location, it can first undergo a low power, local Joint Beam and Null-forming search, and then translate those beams and nulls to external users.

CHAPTER 5
CONCLUSION AND OPEN PROBLEMS

Within this thesis, we have introduced retrodirective systems, presented an experimental implementation and discussed algorithms to improve the scalability and robustness of these systems. The next logical step in this development is to implement these new algorithms in a proof-of-concept system. In addition to this line of work, there is another class of problems that has not yet been considered: receive beamforming and electronic sensing.

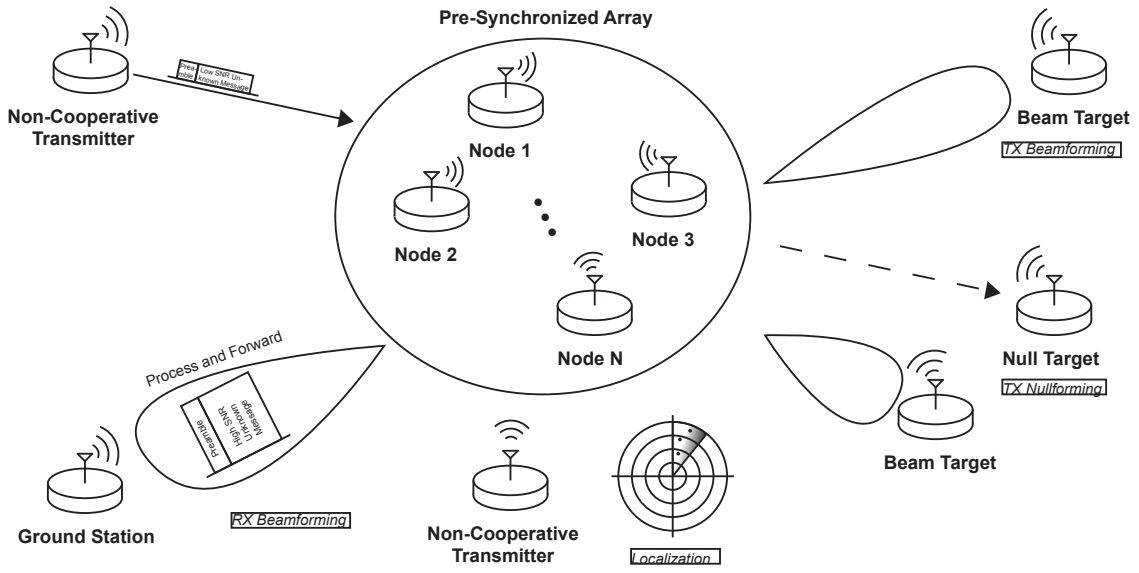

Figure 5.1: A concept diagram showing a variety of techniques that can be enabled by a scalable, retrodirective DMIMO system.

5.1 Advanced Methods in Array Synchronization

Now that we have established a range of methods for antenna array synchronization, designers can choose which one best suits their application. There is a range of complexity and performance, from the simplicity of the LLN aggregate feedback method to the efficiency of the basis function method and the explicit feedback method. However, the basis function method is not fully-developed in the sense that sets of optimal basis functions are still to be developed. It is possible that these might have already been developed to solve another problem (for example, CDMA systems), but there are some unique considerations in our proposed architecture. Even with our simple simulation, we created a large enough set of functions that this aggregate synchronization method could be applied to our experimental setup.

5.2 Applying Retrodirectivity to More Complex MIMO Operations

In addition to the simulations presented in this thesis, it would be interesting to implement a retrodirective phase-only nullforming algorithm in our experimental setup. Perhaps even more interesting would be to implement a retrodirective JBNF algorithm. This would be a substantial step forward and will require improvements to our experimental setup.

5.3 Receive-Side Beamforming

While the primary focus of work in our research group has been to use virtual antenna arrays as transmitting elements, there is an also opportunity to use the array as a receive array. This comes at a potentially high cost, as the amount of

information overhead required to do traditional MIMO RX beamforming is very high in a distributed setting (i.e., every message must be relayed to a central location for processing). Prior work has suggested a process and forward approach [35].

A preliminary contribution in this area could be to implement a JBNF algorithm, wherein the process and forward approach is used on desired data, but undesired noise sources (such as jammers) are nulled out.

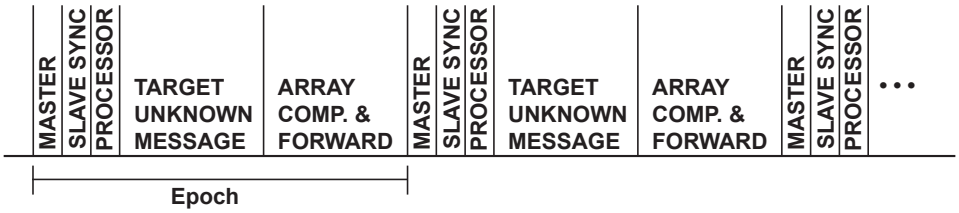

Figure 5.2: A timeslot diagram for a receive beamforming system, where the array compresses and forwards data to an external processor.

5.4 Miscellaneous

5.4.1 Target Localization

Another possible use of the pre-synchronized array is to determine the location of an external target. In fact, by using a receive beamforming, one could actually localize a target in a passive way, allowing for the localization of non-cooperative targets. There are some difficulties in localization using beamforming, for instance the localization process typically relies on a known array geometry and will have some

directional ambiguity unless the orientation of the array is known.

5.4.2 Application to 5G and Wi-Fi Type Wideband Systems

Our discussion so far has been applied to narrowband systems with a simple differential-QPSK modulation scheme. Meanwhile, other systems [5, 14, 37] have applied distributed MIMO concepts to advanced wideband systems such as 5G cellular and 802.11n Wi-Fi base-stations. While these systems are more advanced in terms of data-rate and modulation type, they lack the fully-wireless capability of our scheme, relying heavily on high-speed wired back-hauls for synchronization and coordination. It would be interesting to apply our methods to these wideband standards, as our system would enable DMIMO at the end user, rather than at the base-station level.

5.5 Chapter Summary

In this thesis, we have provided building blocks that can enable Massive MIMO, where hundreds or perhaps thousands of users can coordinate seamlessly. Future work will expand on these building blocks to allow for application to practical systems. Pushing into these frontiers will enable widespread use of DMIMO in cognitive radio, wireless sensors, cellular and Wi-Fi, and electronic warfare systems.

REFERENCES

[1] 3.3v ultra miniature smd hcmos tcxo/vctcxo, 2007. [online] `http://www.foxonline.com/pdfs/fox924.pdf`.

[2] Crystek crystals precision pocket reference oscillator, 2009. [online] `http://www.crystek.com/crystal/spec-sheets/ppro/PPRO.pdf`.

[3] Oven controlled crystal oscillator datasheet, 2014. [online] `http://www.abracon.com/Precisiontiming/AOCJY2.pdf`.

[4] Oven controlled crystal oscillator datasheet, 2014. [online] `http://www.abracon.com/Precisiontiming/AOCJY4.pdf`.

[5] Horia Vlad Balan, Ryan Rogalin, Antonios Michaloliakos, Konstantinos Psounis, and Giuseppe Caire. Airsync: Enabling distributed multiuser mimo with full spatial multiplexing. *IEEE/ACM Transactions on Networking (TON)*, 21(6):1681–1695, 2013.

[6] Paolo Baronti, Prashant Pillai, Vince WC Chook, Stefano Chessa, Alberto Gotta, and Y Fun Hu. Wireless sensor networks: A survey on the state of the art and the 802.15. 4 and zigbee standards. *Computer communications*, 30(7):1655–1695, 2007.

[7] P. Bidigare, M. Oyarzyn, D. Raeman, D. Chang, D. Cousins, R. O'Donnell, C. Obranovich, and D.R. Brown. Implementation and demonstration of receiver-coordinated distributed transmit beamforming across an ad-hoc radio network. In *Signals, Systems and Computers (ASILOMAR), 2012 Conference Record of the Forty Sixth Asilomar Conference on*, pages 222–226, Nov 2012.

[8] Patrick Bidigare, D Richard Brown III, Upamanyu Madhow, Raghuraman Mudumbai, Amy Kumar, Benjamin Peiffer, and Soura Dasgupta. Wideband distributed transmit beamforming using reciprocity with endogenous relative calibration. In *Asilomar Conference on Signals, Systems, and Computers, 2015 IEEE*. IEEE, 2015.

[9] André Bourdoux, Boris Come, and Nadia Khaled. Non-reciprocal transceivers in ofdm/sdma systems: Impact and mitigation. In *Radio and Wireless Conference, 2003. RAWCON'03. Proceedings*, pages 183–186. IEEE, 2003.

[10] D. R. Brown, P. Bidigare, S. Dasgupta, and U. Madhow. Receiver-coordinated zero-forcing distributed transmit nullforming. In *2012 IEEE Statistical Signal Processing Workshop (SSP)*, pages 269–272, Aug 2012.

[11] D. R. Brown, R. Mudumbai, and S. Dasgupta. Fundamental limits on phase and frequency tracking and estimation in drifting oscillators. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5225–5228, March 2012.

[12] D Richard Brown III, Boyang Zhang, Boris Svirchuk, and Min Ni. An experimental study of acoustic distributed beamforming using round-trip carrier synchronization. In *Phased Array Systems and Technology (ARRAY), 2010 IEEE International Symposium on*, pages 316–323. IEEE, 2010.

[13] G. Fettweis, E. Zimmermann, V. Jungnickel, and E. A. Jorswieck. Challenges in future short range wireless systems. *IEEE Vehicular Technology Magazine*, 1(2):24–31, 2006.

[14] Antonio Forenza, Stephen Perlman, Fadi Saibi, Mario Di Dio, Roger van der Laan, and Giuseppe Caire. Achieving large multiplexing gain in distributed antenna systems via cooperation with pcell technology. In *Asilomar Conference on Signals, Systems, and Computers, 2015 IEEE*, 2015.

[15] Gnu radio, 2015. [online] `http://www.gnuradio.org`.

[16] Sairam Goguri, Raghu Mudumbai, Ben Peiffer, Soura Dasgupta, and Upamanyu Madhow. Near optimal algorithm for joint delay and doppler shift estimation in the presence of colored noise. *IEEE Transactions on Signal Processing*, in preparation.

[17] Sairam Goguri, Ben Peiffer, Raghu Mudumbai, and Soura Dasgupta. A class of scalable feedback algorithms for beam and null-forming from distributed arrays. In *Asilomar Conference on Signals, Systems, and Computers, 2016 IEEE*. IEEE, 2016.

[18] Maxime Guillaud, Dirk TM Slock, and Raymond Knopp. A practical method for wireless channel reciprocity exploitation through relative calibration. In *ISSPA*, pages 403–406, 2005.

[19] D. R. Brown III and H. V. Poor. Time-slotted round-trip carrier synchronization for distributed beamforming. *IEEE Transactions on Signal Processing*, 56(11):5630–5643, Nov 2008.

[20] A. Kumar, R. Mudumbai, S. Dasgupta, M. M. U. Rahman, D. R. Brown III, U. Madhow, and T. P. Bidigare. A scalable feedback mechanism for distributed nullforming with phase-only adaptation. *IEEE Transactions on Signal and Information Processing over Networks*, 1(1):58–70, March 2015.

[21] Amy Kumar. *Scalable Algorithms For Distributed Beamforming And Nullforming*. PhD thesis, University of Iowa, 2016.

[22] Amy Kumar, Raghu Mudumbai, and Soura Dasgupta. Scalable algorithms for joint beam and null-forming using distributed antenna arrays. In *Global Communications Conference (GLOBECOM), 2014 IEEE*, pages 4042–4047. IEEE, 2014.

[23] L.B. L. Van atta array, February 17 1970. US Patent 3,496,570.

[24] R. Mudumbai, J. Hespanha, U. Madhow, and G. Barriac. Scalable feedback control for distributed beamforming in sensor networks. In *Proceedings. International Symposium on Information Theory, 2005. ISIT 2005.*, pages 137–141, Sept 2005.

[25] Raghuraman Mudumbai, Gwen Barriac, and Upamanyu Madhow. On the feasibility of distributed beamforming in wireless networks. *Wireless Communications, IEEE Transactions on*, 6(5):1754–1763, 2007.

[26] Raghuraman Mudumbai, D Richard Brown III, Upamanyu Madhow, and H Vincent Poor. Distributed transmit beamforming: challenges and recent progress. *Communications Magazine, IEEE*, 47(2):102–110, 2009.

[27] Raghuraman Mudumbai, Ben Wild, Upamanyu Madhow, and Kannan Ramchandran. Distributed beamforming using 1 bit feedback: from concept to realization. In *Proceedings of the 44th Allerton conference on communication, control and computation*, pages 1020–1027, 2006.

[28] David M Parish, Farhad Farzaneh, and Craig H Barratt. Method and apparatus for calibrating radio frequency base stations using antenna arrays, March 14 2000. US Patent 6,037,898.

[29] Ben Peiffer, Raghu Mudumbai, Sairam Goguri, Anton Kruger, and Soura Dasgupta. Experimental demonstration of nullforming from a fully-wireless distributed array. In *2016 IEEE Global Conference on Signal and Information Processing*, pages 1–5, December 2016.

[30] Ben Peiffer, Raghu Mudumbai, Sairam Goguri, Anton Kruger, and Soura Dasgupta. Experimental demonstration of retrodirective beamforming from a fully wireless distributed array. In *Military Communications Conference, MILCOM 2016 IEEE*, pages 1–6, November 2016.

[31] Ben Peiffer, Raghu Mudumbai, Anton Kruger, Amy Kumar, and Soura Dasgupta. Experimental demonstration of a distributed antenna array presynchronized for retrodirective transmission. In *Information Sciences and Systems (CISS), 2016 50th Annual Conference on*, pages 1–6, March 2016.

[32] C. Pon. Retrodirective array using the heterodyne technique. *IEEE Transactions on Antennas and Propagation*, 12(2):176–180, Mar 1964.

[33] R. D. Preuss and D. R. Brown III. Two-way synchronization for coordinated multicell retrodirective downlink beamforming. *IEEE Transactions on Signal Processing*, 59(11):5415–5427, Nov 2011.

[34] Franeois Quitin, Muhammad Mahboob Ur Rahman, Raghuraman Mudumbai, and Upamanyu Madhow. A scalable architecture for distributed transmit beamforming with commodity radios: Design and proof of concept. *Wireless Communications, IEEE Transactions on*, 12(3):1418–1428, 2013.

[35] Franois Quitin, Andrew T Irish, and Upamanyu Madhow. A scalable architecture for distributed receive beamforming: analysis and experimental demonstration. *IEEE Transactions on Wireless Communications*, 15(3):2039–2053, 2016.

[36] Muhammad M. Rahman, Henry E. Baidoo-Williams, Raghuraman Mudumbai, and Soura Dasgupta. Fully wireless implementation of distributed beamforming on a software-defined radio platform. In *Proceedings of the 11th International Conference on Information Processing in Sensor Networks*, IPSN '12, pages 305–316, New York, NY, USA, 2012. ACM.

[37] Hariharan Rahul, Swarun Suresh Kumar, and Dina Katabi. Megamimo: Scaling wireless capacity with user demand. *Proc. of ACM SIGCOMM 2012*, 2012.

[38] Dzulkifli Scherber, Patrick Bidigare, Ryan O'Donnell, Matthew Rebholz, Miguel Oyarzun, Charles Obranovich, William Kulp, David Chang, and D Richard Brown. Coherent distributed techniques for tactical radio networks: Enabling long range communications with reduced size, weight, power and cost. In *Military Communications Conference, MILCOM 2013-2013 IEEE*, pages 655–660. IEEE, 2013.

[39] Ettus research, 2015. [online] `http://www.ettus.com`.