A FRAMEWORK FOR COMPUTER VISION ASSISTED BEAMFORMING IN APERIODIC PHASED ARRAYS

A Dissertation

by

JEFFREY SCOTT JENSEN

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Gregory H. Huff
Committee Members,	Jean-Francois Chamberland
	Robert D. Nevels
	Harry A. Hogan
Head of Department,	Miroslav Begovic

December 2015

Major Subject: Electrical Engineering

ABSTRACT

Mobile networks, unmanned air vehicles (UAVs), and other dynamic wireless systems are prevalent in several widespread military and commercial applications due dynamic ability to adapt to an environment and commonly implemented autonomous control. Their mobility and dynamic reconfiguration create randomly dispersed networks that inherently give rise to significant electromagnetic challenges in collaborative applications such as beamforming. Conventional radio and communication systems overcome these challenges through designs that are highly structured with respect to frequency and are static in nature.

Two inherent problems prevent collaborative electromagnetic capabilities in these disparate random geometrical systems: cognizance of node positioning and local synchronization of oscillators, phase, and information. This work proposes the combined use of image processing techniques and infrared depth-of-field sensing to detect and track node position in a phased array control framework for morphing clusters of randomly distributed antennas. This framework is presented and designed to uniquely identify array elements (or platforms) and track the motion-dynamic spatial distribution to provide feedback and control information for phase shifting and beamforming. A primary metric of this work is to examine the core performance of the phased array control system with respect to beamforming accuracy. This begins with the use of image recognition algorithms in a reconnaissance phase to establish element identities and determine their locations in the field of view. This process informs the depth-of-field

sensor to prompt evaluation of the spatial distribution of elements and enable element location tracking through time. This information is communicated to a distributed array controller that identifies the characteristic function of the array (triangular, spheroidal, etc.), and calculates phases for the elements to achieve the desired beam steering operation. The framework also includes a mobile device (smartphone, tablet, etc.) as a user interface which can be used to control the phased array and link geolocation information for autonomous tracking modes. A framework operating at 2.48 GHz has been developed using low-cost off-the-shelf components, as well as custom-designed element platforms so the performance of the system can be observed experimentally. Results for element identification and spatial distribution are included to benchmark the accuracy of the aforementioned system. Next, a series of experiments demonstrates the operation of the proposed system through radiation patterns that incorporate beam steering and other complex control mechanisms. Analysis of the patterns shows from various geometrical topologies is presented to demonstrate the capability of the system to analyze morphing swarms and clusters. Finally, a conclusion presents findings from some noticeable differences in simulated and measured results.

DEDICATION

This dissertation is dedicated to my wife and parents.

ACKNOWLEDGEMENTS

I would like to thank my committee chair, Dr. Gregory Huff, for his help and experience throughout this project. I would also like to thank my committee members, Dr. Chamberland, Dr. Nevels, and Dr. Hogan for their guidance and support throughout the course of this project.

This material is based upon work supported by, or in part by, the U.S. Army Research Laboratory and the U.S. Army Research Office under contract/grant number: W911NF-09-1-0429

NOMENCLATURE

AF	Array Factor
ARP	Address Resolution Protocol
DOA	Direction of Arrival
FR4	Glass based epoxy
GPS	Global Positioning System
HSV	Hue Saturation Value or Brightness
I2C	I-squared-C Bus
ID	Identification
IP	Internet Protocol
IT	Information Technology
LED	Light Emitting Diode
LIDAR	Light Detection And Ranging
LiPO	Lithium Polymer Battery
LO	Local Oscillator
LOS	Line of Sight
LOCUST	Low Cost Unmanned air vehicle Swarm Technology
MAC	Media Access Control
ONR	Office of Naval Research
OSI	Open Systems Interconnection
PCB	Printed Circuit Board

POE	Power over Ethernet
P2P	Point to Point
RGB	Red Green Blue
RGBD	Red Green Blue Depth
SPI	Serial Peripheral Interface
TCP	Transmission Control Protocol
UAV	Unmanned Air Vehicle
UDP	User Datagram Protocol
UI	User Interface

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

CHAPTER I

INTRODUCTION

Nearly 10 billion individual wireless devices are connected to the Internet as of 2014 with an expected 30 billion wireless devices connected by 2020. This exponential increase of wireless devices offers a unique opportunity to look beyond the capabilities of single or even dual radio devices for point-to-point communication and envision how these devices can interact and collaborate to take advantage of their electromagnetic components such as radios, antennas, and power. Grouping elements into collaborative networks by synchronizing their radios and information could allow distributed radios and communication components to act as contributing nodes to a larger array [1-5]. Arrays inherently bring several electromagnetic advantages in comparison with a single antenna such as increased gain, bandwidth, and directivity. These benefits heavily contribute to the design of communication networks as technical considerations such as power, battery lifetime, and communication range hinge on a system's ability to utilize communication components efficiently.

Several examples of collaborative radio clusters already exist as prototypes or completely functional units. ONR started project LOCUST (currently a prototype) to develop low-cost autonomous UAVs for military applications. The pre-flight capsule LOCUST UAVs are launched from a single canister launcher and then extend their wings to begin autonomous flight after successful propulsion. After many successful launches the UAVs collectively organize into a mesh network and collaborate to carry

out a mission. A current constraint of the UAVs lies with their inability to communicate over large distances due to a small onboard battery and physical space for a large, highly directive antenna. While the UAVs can communicate with one another through the mesh network, longer range bi-directional communication is simply not possible. Synchronization of the UAV communication systems could overcome this barrier by leveraging the independent communication systems of each UAV as a collective group into a larger array to achieve the communication distances needed. An additional example is seen with nano-quad rotors made by KMel robotics (now Qualcomm) that rely on localized wireless communication to collaboratively accomplish group tasks that single UAVs could not. The quad rotors carry a small battery, an onboard communications system, and a small dipole antenna. Individual quad rotors are capable of receiving commands from a central control system or managing quad rotor; however, greater distance communication must be achieved by leveraging the power and antennas of the individual quad rotors together.

There are several challenges that preclude successful collaborative morphing clusters of random nodes into an array. One primary challenge relates to positional awareness of each node in the system with respect to other nodes. The theory that dictates the array behavior requires that the physical position of each node is known in order to generate accurate beam steering information. Surveys of literature in Section II highlight several methods using GPS and DOA techniques to determine position of nodes in a random geometrical arrangement. It should be mentioned that positional awareness is also dependent on frequency. Higher frequencies have shorter wavelengths

that leave less room for error estimation of position. A second challenge relates to synchronizing the communication systems of the individual nodes or elements. Specifically, the disparate physical location of each element typically implies that each node's communication system will run on a self-determined or instantiated clock signal often termed LO [1, 2, 5, 6]. The LOs from each individual node must be synchronized so that the communication modulation schemes, data flow, and phasing signals are the same. Furthermore, once the LOs of the nodes have been synchronized, the information and data rate of the individual nodes must also be scheduled. For example, a node which receives information first must transmit information coinciding with a node that receives information 100 milliseconds after it, requiring a schedule of sending and receiving to be developed for the nodes. These problems are further exacerbated by the fact that UAVs, mobile swarms, and other dynamic networks constantly change position and state, adding further complexities to determine position and synchronize individual nodes. Although synchronization of communication systems is an important aspect to this problem, it will not be researched within the scope of this work. Element and node positional awareness is an important precursor to synchronization and will be studied in support of future research of wireless synchronization.

An additional challenge in leveraging random nodes collectively into an array framework is to predict the array and respective radiation behavior given a characteristic distribution function of elements. Literature survey in Section II shows that comprehensive theory for random arrays has been developed and tested to support deterministic behavior given several geometrical distributions. An outcome of this

implemented framework supports the theory and literature that have been documented surrounding random arrays.

This research promotes a new methodology and framework to address positional awareness, detection, and node tracking in a random cluster. The primary goal of achieving this is to (1) provide a functional random array test framework to test and design various configurations of random arrays (2) use spatial awareness as a primary research tool to support further research in synchronization of wireless communication systems. The proposed mechanism detects object's or node's position through spatial recognition and visual analysis. Low-cost vision systems and open-source computer vision libraries have emerged as usable tools for development of these spatial recognition techniques. The advantage of these tools derives from tightly coupled color and depth images that link visual environmental properties with a respective physical location. Leveraging this relationship provides necessary information needed to control beamforming and beam steering properties of random arrays. The overall outcome of this research produces a framework capable of autonomously controlling random arrays for beamsteering, phasing, and collaboration applications and provides deeper insight into the functionality of random arrays as a whole.

CHAPTER II

LITERATURE SURVEY

The important literature surrounding this framework concerns two focus areas: random array theory and computer vision in autonomous object frameworks. Random arrays have been studied extensively in the past few decades from theoretical and mathematical standpoints [7-12]. Y. T. Lo is one of the most heavily cited authors surrounding this subject based on his original mathematical paper developed in 1963. Lo derived expressions to better quantize an aperiodic array with respect to its gain, power, fields at an observation points, and several other variables that are considered important in random array theory [10, 11]. Additional studies focused on analyzing the expected value of electric fields, power, and intensities given various distributions from a theoretical standpoint in order to gain better understanding of random array behavior. The outcomes of these studies gave rise to establishing characteristic functions, mean variance, and other array characteristics based on statistical probabilities and approximations [7, 10, 11]. These studies established fundamentals for improved theory derivations using the Fourier transform to predict the behavior of random arrays configured in various topologies [7].

After the mathematical fundamentals surrounding aperiodic arrays were developed, further research looked into performance behavior versus standard phased arrays and their design optimization [3, 4, 7, 12-16]. Significant factors such as vibration of elements, array factor tapering optimization, and other optimization routines have

been developed to address the radiation behavior differences that arise in random arrays in comparison with phased arrays. Random arrays were shown to have distinct advantages in certain application spaces over phased arrays warranting the research and development of design optimization routines and algorithms [1-4, 7, 8, 16]. For example, Vigano et. al. demonstrate the capability of using sparse antenna arrays for earth coverage satellite applications that notably leverage the functionality of random arrays [16]. While these research efforts provided useful simulation tools and results that demonstrated the potential benefits of random arrays, the fabrication of said arrays had not been conducted to verify the results. In fact, the first studied random array and respective radiation pattern results were not obtained until around 2010, although Lo's experiments could be considered to be the measure of the first aperiodic array [10, 11].

In order to better study these designs, it is imperative that literature supports an effort of fabricating and testing these arrays and theories, as some literature speculates that coupling and dispersion of elements can be a factor in array behavior concerning predictive side lobe levels and array performance [7, 8, 13, 15]. As aforementioned, the latest study on fabrication and testing of aperiodic arrays was performed in 2010, where the authors demonstrated differences in linear, planar, and volumetric aperiodic arrays from both theoretical and practical standpoints. In this study the authors constructed 2-D and 3-D random arrays from microstrip patches and wire monopoles [7]. An array controller leveraging a Smartphone as a control interface provided phasing information for a control system that generated beamsteering radiation patterns [17]. The authors also demonstrated the ability to generate sum-difference radiation patterns and an ability to

steer these sum-difference radiation patterns off of broadside radiation. The authors in these cases note that the position of each antenna or node in both the 2-D and 3-D distributions was hand measured [7]. Additional literature surrounding distributed arrays demonstrates the capabilities of a navy ship to contain wirelessly synchronized elements to provide radar capabilities [3]. In particular, the authors use wireless transceivers with multiple wireless channels to synchronize local oscillators, schedule information propagation, and synchronize frequency. The location and dynamic repositioning of elements is ignored as the operating frequency band is 3-300 MHz and movements amount to fractions of a wavelength. The authors here mention similar challenges seen with LO and time synchronization while thoroughly covering distributed array and radar theory to support their challenge statement [3]. It is clear from surveying the literature that both element position detection and synchronization of communication systems is quintessential for randomly distributed array control networks. Additional reading surrounding the local oscillator problem can be found in several other research efforts; however, the goal of this work does not include the synchronization of LOs [2-6, 18-20]. To alleviate the position estimation problem, it is proposed that the inclusion of a mechanism like computer vision offers potential solutions based on surrounding literature.

Low-cost vision systems and open source computer vision libraries have enabled development for spatial awareness techniques for intelligent and autonomous systems. Applications leverage color and depth image streams to filter through images to perform object detection, tracking, and avoidance tasks [21-43]. Specifically, one application has

used a Microsoft Kinect to track UAV colored objects using several image processing techniques and transforms, such as the HSV transform, to predict its position and trajectory through space [28]. These authors developed algorithms to account for a UAV's velocity, thrust, and hover states based on color and depth information gathered by the Kinect. Tests show accurate projection modeling and tracking of the UAVs through space based on color and depth analysis. In another example, the authors used depth imaging from a Microsoft Kinect and generated cumulative surface normal vectors to detect doorways for mobile land robotic movements [41]. The algorithm spatially separates different depths of field and develops a model for the total observed distances the robot measures. This method strictly uses depth of field (no RGB data) recognition from infrared light to determine physical objects and their positions allowing the land robot to successfully navigate through several doorways in an enclosed dwelling. But the computer vision libraries are not limited purely limited to Microsoft Kinect images. In an application using autonomous cars a LIDAR camera is affixed to an autonomous vehicle and a computer utilizes open source image processing libraries to perform object detection and avoidance algorithms for the vehicle [30]. The LIDAR images are processed using the same code set and algorithms but estimate distance and object detection within the vehicle's field of view with greater accuracy due to the LIDAR's resolution and ranging capabilities in comparison with the Kinect. Additionally, many researchers have analyzed using RGB light communication and detection as a way for machines and autonomous systems to exchange information with one another. Specifically, detection of LED traffic lights have been studied using high speed cameras

to study the possibilities of using visible light communication at stoplights to interlink control subsystems together [37]. Overall, survey of literature indicates promising potential for a computer vision assisted random array control framework.

Other studies related to detection of element position specifically in random arrays have focused on using GPS or additional mechanisms to determine element positioning [3, 4]. In one specific study, the MUSIC algorithm has been used in combination with a Euclidean space direction of arrival reading to estimate cooperative node location in a cluster [44]. Neither of these methods proposes using computer vision in any way to further enhance their studies and is confined to using existing embedded technologies to determine node location within a cluster to support ad hoc beamforming.

CHAPTER III

ARRAY THEORY

Basic array theory is presented to gain further understanding into the importance of element positioning and its relation to the radiation pattern provided by the array. Understanding array properties and governing equations allows the control framework to synthesize array theory into controllable software.

To begin, analysis of a single y-directed point source provides adequate understanding and derivation of basic array theory properties. A general form of Helmholtz equation is seen in (1).

$$\nabla^2 \vec{E} + k^2 \vec{E} = 0 \tag{1}$$

Equation (1) describes the solution of a free space plane wave given no sources present. Next, consider the magnetic vector potential solution of a point source beginning with (2).

$$\nabla^2 \vec{\psi} + k^2 \vec{\psi} = I_0 l \delta\left(\vec{r}_n - \vec{r}\,'\right) \tag{2}$$

Equation (2) represents the solution of a single point source with moment $I_0 l$. The solution to this equation is represented in (3).

$$\vec{\psi} = I_0 l \frac{e^{-jk|r-r_n|}}{4\pi|r-r_n|} = I_0 l \frac{e^{-jkR}}{4\pi R} \tag{3}$$

From (3), the variable ψ represents the magnetic vector potential of the point source. Next, it is well known that the electric field solution from a magnetic vector potential is equivalent to (4).

$$\vec{E} = -\frac{j\omega\mu}{k^2}(k^2 + \nabla\nabla\bullet)\vec{\psi} \tag{4}$$

Inserting (3) into (4), the far-field expression for the electric field is seen in (5).

$$\vec{E} = \hat{a}_\theta j\eta \frac{kI_0 l e^{-jkr}\cos\theta}{4\pi r} e^{j(kd_z\cos\theta+\beta_z)} \tag{5}$$

In equation (5), it can be assumed that the point source is placed at the origin resulting in $d_z = 0$. Now take the case of two y-directed point sources of similar moment $I_0 l$ separated from the origin by a distance of $d/2$ as shown in Figure 1.

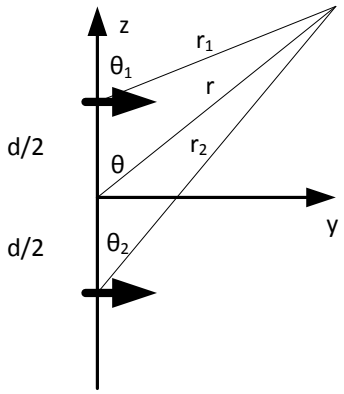


Figure 1. Two y-directed point sources

Realizing the solution obtained in (5), a combination of both point sources yields (6).

$$\vec{E} = \hat{a}_\theta j\eta \frac{kI_0 l e^{-jkr}\cos\theta}{4\pi r}[e^{j(kd_z\cos\theta+\beta_z)/2} + e^{-j(kd_z\cos\theta+\beta_z)/2}] \tag{6}$$

The resultant (6) yields two specific terms as defined by the brackets. The first term is what is described as the element pattern or the single element radiation envelope. This term is multiplied with a combination of phase terms noted as the array factor or *AF* as seen in (7) to provide the total radiation pattern.

$$E_{total} = [ElementPattern]*[ArrayFactor] \tag{7}$$

From (7), the total resultant E field and possible variations of this E field are governed by the total radiation pattern of the element and a dependency on steering angle θ. This indicates that a highly directional antenna array may have limited scanning capability in comparison with an array comprised of omni-directional antennas. The array factor for this uniformly excited array can be generally written for *P* number of elements in (8).

$$AF_z = \sum_{p=1}^{P} I_0 e^{j(p-1)(kd_z \cos\theta + \beta_z)} \tag{8}$$

The distance of separate d_z in (8) shows integer spacing of elements that linearly reside along the z axis. The total array factor for the 1-D case may be more concisely written as (9).

$$AF_z = \left[\frac{\sin(\frac{P}{2}\zeta)}{\sin(\frac{1}{2}\zeta)} \right] \tag{9}$$

where the symbol ζ is represented by

$$\zeta = kd_z \cos(\theta) + \beta \tag{10}$$

The realization of (9) is the compact form of the array factor for an *N* element linear array. To normalize the array factor, (9) is divided by the total number of elements shown by

$$AF_z = \frac{1}{P}\left[\frac{\sin(\frac{P}{2}\zeta)}{\sin(\frac{1}{2}\zeta)}\right] \tag{11}$$

Multi-dimensional arrays follow a similar formulation and approach with the exception of the element placement reference of the phase term. Elements distributed along the x and y axes have the respective array factors shown in (12) and (13)

$$AF_x = \sum_{m=1}^{M} I_0 e^{j(m-1)(kd_x \sin\theta\cos\varphi+\beta_x)} \tag{12}$$

$$AF_y = \sum_{n=1}^{N} I_0 e^{j(n-1)(kd_y \sin\theta\sin\varphi+\beta_y)} \tag{13}$$

For elements contained in Euclidean space which are off the primary x, y, and z axes, the total array factor is realized in (14).

$$AF(\theta,\phi) = I_0 \sum_{m=1}^{M} e^{j(m-1)(kd_x \sin(\theta)\cos(\phi)+\beta_x)} \sum_{n=1}^{N} e^{j(n-1)(kd_y \sin(\theta)\sin(\phi)+\beta_y)} \sum_{p=1}^{P} e^{j(p-1)(kd_z \cos(\theta)+\beta_z)} \tag{14}$$

From equation (14), it is clear that the AF has a maximum when all of the exponents are equal to 0. This maximum represents the steering direction of a main beam that is dependent from angles (θ,φ). With the exponents set equal to 0, the primary phase shifting equations for the x, y, and z axes are seen in (15), (16), and (17) respectively.

$$\beta_x = -kd_x \sin(\theta)\cos(\phi) \tag{15}$$

$$\beta_y = -kd_y \sin(\theta)\sin(\phi) \tag{16}$$

$$\beta_z = -kd_z \cos(\theta) \tag{17}$$

Equations (15-17) produce the necessary phase information to control the primary radiation direction of the array. In the random array scenario, the distance *d* will represent a non-integer multiple of the wavelength for elements of the array leading to an *aperiodic* or random configuration of array elements. Further investigation into the distribution of elements, the statistical estimation of the distribution, estimation of side lobe behavior, and other various mathematical considerations are not taken into account within the scope of this work. In this case, the mathematics surrounding the operation of the aperiodic array are focused on generating the necessary phasing information needed to control behavior radiation patterns from the aperiodic array, and although other advanced radiation steering techniques can be explored using these mathematics, the author encourages the reader to explore previous work highlighted in Chapter II.

CHAPTER IV

SYSTEM THEORY AND DESIGN

This chapter focuses on the development of the theory behind the proposed framework in Chapter V. This chapter explains the detail behind the control systems that coordinate information about the aperiodic array. First, a discussion on swarm intelligence will help setup the argument for a proposed system architecture for the framework. This will be followed by a previous control network created by authors discussed in Chapter II, as this work will build off of the existing platform that has been tested and built to work with aperiodic arrays [7, 17].

Swarm Intelligence

Although not a critical aspect of this work, swarm intelligence (SI) provides a solid reasoning basis for proper construction of this proposed framework. In its basic sense, swarm intelligence has related studies of ants, birds, bees, and other wildlife to the behaviors that they perform as a group in accomplishing tasks. When approaching the swarming array theory model, it is important that the network, communication protocols, communication mediums, and additional infrastructure are designed accordingly to how they can best support the underlying framework. In the case of this work, the swarm intelligence community has concluded through many outstanding research projects that the ability of ants, bees, or elements to act as a colony, hive, or group respectively, outweighs the performance benefit of a single ant, bee, or element to perform a single task by itself [45]. The idea of a hive or colony takes on both physical and theoretical

senses in that a hive serves as a physical congregation location for bees as a common meeting point, and further, a colony can serve as a congregation point for ants; however, the theoretical sense can also be taken in that ants and bees do not merely work together inside of the hive, but rather this extends to their common tasks. This is a similar case for object swarm theory. Objects can draw relational data about one another from a common congregation point and then take this information as they pass through different spaces. The objects may still exchange information directly with one another, but the common meeting and convergence point for their system will provide a synchronization of these objects for coordinated tasks.

In this work, the systematic synchronization and proposed framework to choreograph the radiation properties of an array is supported by the theory to develop a central 'hive' or 'colony' for the elements. The hive will provide the necessary means to gather and disseminate information at a holistic level to all elements or nodes participating on the network at any time. This "a whole is greater than the sum of its parts" will provide a common meeting point for information from all elements of the aperiodic array and will provide better understanding for the array and its behavior. The concept of having a central meeting point for information will also incorporate the framework's ability to leverage the power of computer vision. Instead of implementing computer vision at the object level, the 'hive' or base station will use computer vision to track the information about the objects of the array and subsequently inform all elements about the current status of the entire system.

Communication Theory

A natural occurrence within any coordinated system is the need and ability to exchange data quickly and effectively. Swarm systems and distributed systems have varying needs of information and the throughput of information in order to operate correctly at a system level. Based on the theory of having a central base station there are a couple of options for communicating information to the objects contained within the swarm. The styles of communication in the approach of this research are compared from a networking perspective. The first approach is to use *unicast* packet data exchange where the central base station simply tells a uniquely identifiable object the information it needs to know. In comparison with IP addresses, this is similar to a router at 192.168.0.1 sending information specifically to a computer at address 192.168.0.2. Even if other nodes exist on the network, the only object to actually receive the information is the intended object of the network address from the central command station. The second address is to use *multicast* packet data exchange, where one intended sender distributes a message to a group of intended listeners at a single time. In the case of a networking example, a router at address 192.168.0.1 could distribute a message to a group of computers at 192.168.0.1-4, even if the only listeners that will do something with the information are located at addresses 192.168.0.1-2. The final approach is to use *broadcast* packet data exchange where all nodes connected to the network hear the intended message whether they have use for the information or not. All these modes of communication have positive and negative aspects. Multicast and broadcast are very similar forms of communication differing only from the fact that multicast packets do

not need to include every node attached to the network. The argument between unicast and multicast data exchange is that unicast only delivers information to those node that need to hear the information and multicast delivers information to more nodes simultaneously, but this creates the intentional consequence of more traffic, information, and potential useless data exchange.

This network takes the approach similar to the study of swarm intelligence of ants. Ants use chemical communication signals (pheromones) to alert multiple groups of ants at a single time about a particular task, danger, or other situations. It is arguable that because not all ants need to know specifically about the situation, all the ants are at least informed of the situation as it happens. In the same way, using a central base station in this framework will use broadcast and multicast styles of communication to inform all nodes on the network about particular states of the base station or even other information about nodes within the network. In the design phase, this will help the system as the individual elements can cache present 'useless' data, but eventually use this data in the future to inform the central station about potential conflicts involving physical, network, or electromagnetic state. This coordinated set of communication will create several simultaneous conversations with the base station at once and therefore, the base station must be intricately designed to handle such communications and be aware of the states of the nodes in the system at all times. Broadcast communication is commonly used in networking for discovery of nodes within a network as a single packet of information can prompt an entire network to respond to a question. As an example, a router uses ARP to discover new MAC addresses that join an Ethernet network to gain information

about the new node that joined. In the same way, the hive or base station will use a similar protocol to discover elements participation in the aperiodic array. Unicast style will not be used in this particular implementation as the system does not expect to have a comparatively large amount of traffic as would be found with an enterprise IT network. Thus, the argument for information flooding by the use of multicast packets instead of unicast packets is less considered within the design scope of this framework.

Computer Vision

As aforementioned, low cost computer vision components have become incredibly prominent in the past decade due to consumer demand for innovation in gaming, avionics, robotics, and several other applications. Beyond the implementation of a single webcam, the industry developed new technologies that could enhance the information of images by coupling depth information. Microsoft was one of the first primary developers of a low-cost, three dimensional camera that could receive both RGB and depth information from multiple sensors.

Figure 2. Microsoft Kinect camera version 1

The Kinect, shown in Figure 2, is capable of reading RGB frames at a resolution of 640x480 at a frequency of 30 Hz. Additional documentation has stated that the Kinect can be over clocked to provide 60 Hz resolution if desired [25, 33, 46, 47]. In the case of this framework, 30 Hz will be fast enough as the framework will not actually be surveying motion-dynamic UAVs that would require a faster update rate and information processing speed. Additionally, at the time of writing, two versions of the Kinect are available. This framework uses version 1.

The Kinect provides color and depth information that are linked by the respective color pixel as shown in Figure 3.

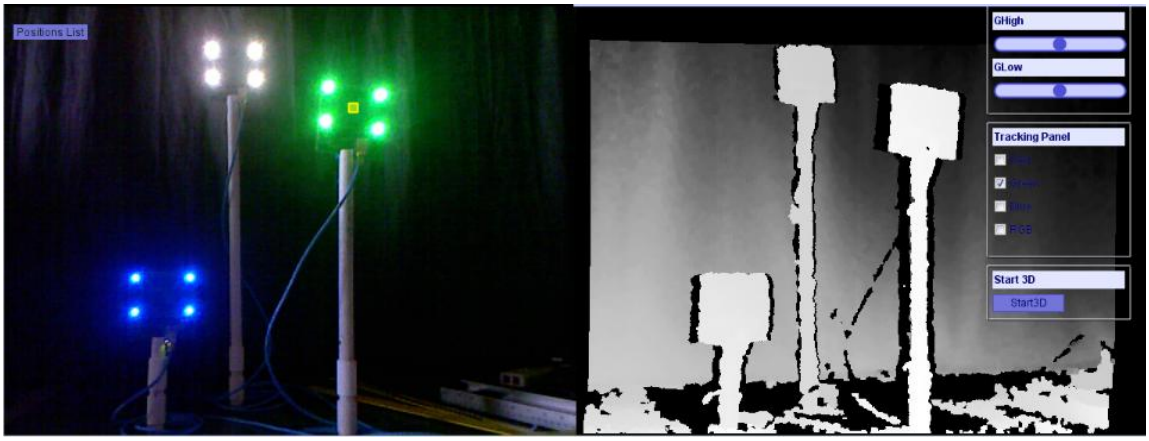


Figure 3. (Left) Kinect color image (Right) Kinect depth image

The Kinect uses scattered infrared light to determine the distance of a particular object. This 'scattering' therefore does not immediately co-align with the RGB matrix that is received by the CMOS camera and therefore, requires post processing of data in order to link the color and depth images together [47]. It should also be noted that in the case of this framework, using the Kinect camera outdoors will not work due to the infrared technology being used.

Because of the many uses of this information and device across several application fields, the open source software community has spent many years creating drivers and APIs that take the information generated from the Kinect and turn this information into accurate, meaningful data. In this framework, the primary open source software libraries used to support reading information from the Kinect are OpenNI, Processing, and OpenCV.

OpenNI, or open natural interaction, was a non-profit organization first setup in 2010 to support the development and sustainability of the Kinect. In the libraries developed, much of the math, algorithms, and processing of depth information coming from the Kinect is handled through the Kinect sensor itself and OpenNI libraries.

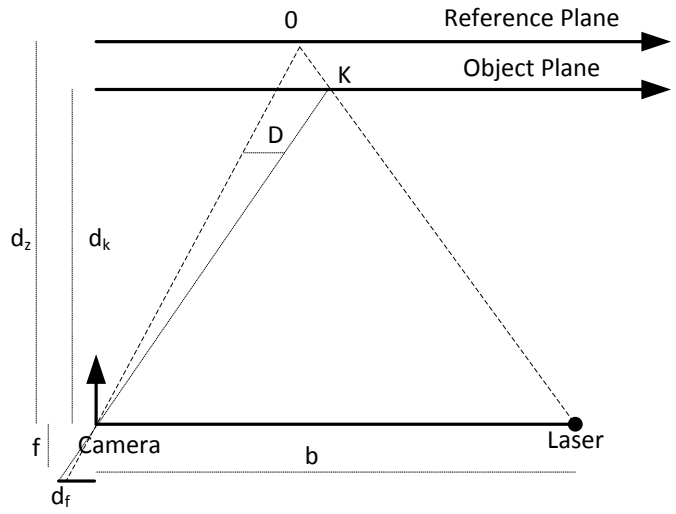


Figure 4. Mathematical calculation for determining distance from Kinect

Figure 4 demonstrates how the Kinect is able to determine distance using scattered infrared light. Studies on the accuracy of the Kinect and this method of depth perception have been studied by numerous authors and have proved that the accuracy of the Kinect measuring at a distance of 1-3 meters is around 0.5 mm [48]. Accuracy plays an important role in this framework as improper or bad measurements of distance of the elements can result in errors that propagate to the phase shifters of the antennas leading

to improper radiation patterns of the aperiodic array. Based on the accuracy of the Kinect in several other experiments, including those in Chapter II of the literature survey, it is appropriate that this computer vision utility provides the necessary characteristics to work within the framework.

Image Processing

Once the Kinect receives the RGB and RGBD information from the surrounding environment, this information must be forwarded for further processing to distinguish colors, objects, distances, and other information from the coupled images. Image processing plays a heavy role within this framework as it serves primarily to uniquely distinguish antenna elements in the aperiodic array from other elements, objects, and the background noise of the image. The algorithms and methods of image processing have long been studied due to their use in color images dating and have primarily been focused on filtering RGB images [35-37]. Because of the depth information generated from the Kinect, new image processing techniques using point clouds has been of recent interest. Point clouds simply take an RGB pixel and the distance associated with that pixel and places the pixel into 3-dimensional space. This type of associate allows the computer or system using the Kinect to essentially envision the 3-dimensional environment around them.

Using both color and depth images in the framework provides more information than simply color images and will allow the framework to determine distances with good accuracy and precision. The nodes or antenna elements in the system will need to be uniquely identifiable in order for image processing techniques to work. Therefore, the

focus of this framework will be to first distinguish a node by its RGB color and then determine physical characteristics about the node based on the filtered color image.

Filtering images by color can be simple or complex. A simple approach to working with color images is to apply simple color thresholds to RGB images. For example, an algorithm can go pixel by pixel determining whether the pixel's RGB color lies within a permissible threshold range. If the system were looking for a light red color, the condition for an RGB threshold may look like (18).

$$RGB(100,0,0) < color < RGB(255,0,0)\,?\; pixel = 1 :\; pixel = 0 \tag{18}$$

Note that the statement takes an original RGB pixel and converts it from a 256-bit color to a binary 1 or 0. Applying this particular threshold technique produces a black and white image: black pixels where the statement was false and white pixels where the statement was true. This method works well when the colors never change and the environment consistently produces proper lighting, color emission, and low white noise; however, most scenarios do not maintain these color impervious environmental principles and thus demand more advanced techniques to filter color images.

The first technique this framework will use to perform image processing is to convert the RGB 640x480 pixel frame from the Kinect to an HSV color space image. There are several advantages for doing this. First, the HSV color space is better recognized by computers as normal RGB colors are created by varying the intensity of red, green, and blue pixels. These varying intensities are heavily affected by the ambient light and do not reflect the accurate intensities of light from each color. Furthermore, to change the intensities of particular colors in the RGB color spaces requires almost

arbitrary adjustments of RGB in order to maintain a color transformation. The HSV or HSL color space has been heavily documented to provide better performance for image processing and analysis.

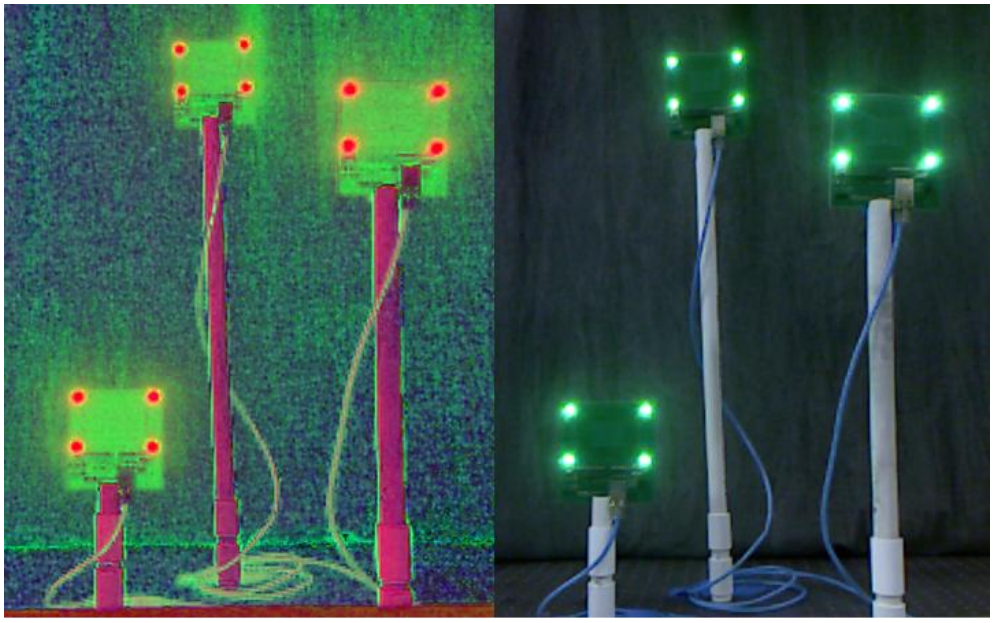


Figure 5. (Left) HSV color image (Right) normal RGB image

Two primary transformations exist for converting RGB to HSV; however, this framework will use equations (19)-(21) to transform the color images.

$$\begin{aligned} \alpha &= 0.5*(2R-G-B) \\ \beta &= \frac{\sqrt{3}}{2}(G-B) \\ Hue &= a\tan 2(\alpha,\beta) \\ Chroma &= \sqrt{\alpha^2+\beta^2} \end{aligned} \tag{19}$$

$$V = MAX(R, G, B) \tag{20}$$

$$C = 0 ? S = 0 : S = \frac{C}{V} \tag{21}$$

The next process the framework will use is the use of color thresholds based on the HSV color space. It should be noted that this is not a comparison of HSV values but a comparison of RGB values in the HSV space. Thus, the process of using equation (18) satisfies the condition for a threshold image contingent upon the upper and lower bounds.

After an image has had a threshold applied, it is converted from a 4-dimensional dataset to a 1-dimensional data set in that each pixel is either represented by a 1 or a 0 (i.e. white or black) based on (18). Because ambient light noise may exist in the system, it is necessary to apply further filtering techniques in an attempt to smooth the image so that it may be further processed with ease. Two primary functions in image processing exist to accomplish this task: dilation and erosion. Dilation in morphology can be best described as the expansion of the positive binary of an image. Thus, the dilation of a binary image is taken in the Euclidean space R^d or an integer grid of Z^d for some dimension of d. In this case, the dimension of d will be 4, as the total space is expanded by red, green, blue, and alpha channels. Therefore, let E be Euclidean space (or an integer grid) and let A be a 640x480 binary threshold image with matrix B as a structuring sub element in the space of R^d. Under these conditions, the dilation of binary image A by a structuring element B can be defined by (22).

$$A \oplus B = \bigcup_{b \in B} A_b \tag{22}$$

In a graphical form, the expansion of the binary image according to (22) looks similar to Figure 6. Note that the size of the black holes contained within the white outline become significantly smaller after the dilation due to the expansion of the binary subset matrix. Additionally, it is known that the size and structure of *B* can be defined in order to best suit the needs of the application or the specific algorithm. A common structure of *B* that will be used in this work is a 3x3 identity matrix.

Figure 6. (Left) Original threshold image (Right) dilation of binary image

Conversely, erosion performs an operation that eliminates binary positives within an image due to a negative structuring element. Given the same presets described for the dilation principle, erosion takes a binary image *A* in a subset of Euclidean space *E* and is transformed by the structuring matrix *B*.

$$A \triangle B = \bigcap_{b \in B} A_{-b} \tag{23}$$

Equation (23) shows that the symmetric difference of matrices *A* and *B* results in the erosion of binary image *A*. In this framework, the structuring matrix *B* is composed of a

3x3 matrix where every position in row and column is filled with a ‘0’ resembling a black pixel.

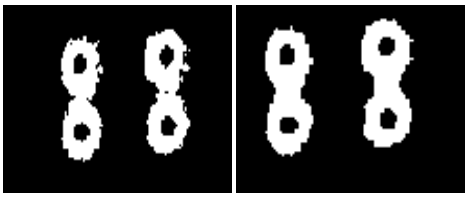


Figure 7. (Left) Threshold binary image (Right) eroded image

Note that in Figure 7, the threshold binary image contains noise, seen as white ‘specs’ that float off of the main white bodied image. After the erosion however, these instances of floating white spaces have been eliminated due to the structuring matrix that has eroded the image.

The processes of dilation and erosion provide semi-filtered images that eliminate ambient noise and provide more useful data to analyze through further processing techniques. In particular, these solid binary images (with the assumption all ambient noise has been eliminated) can provide simple access to other algorithms such as edge detection. Edge detection provides use in that objects can be identified through these filtered images and their space in terms of pixel location can be identified. Edge detection and generation can also provide data for other algorithms such as contour detection, Huygens algorithm, and other shape detection algorithms. This framework will use edge detection and contour definition to identify antennas once the respective

images have been properly filtered. The contour algorithms are provided by the OpenCV library and documentation surrounding these algorithms can be found at the given reference [49]. In general, the process for edge detection looks at each pixel and the neighboring pixel to determine if there is a change in value. For a binary image, this process becomes rather straightforward as each pixel is compared to a directed neighboring pixel. For example, in a 'look-left' edge detection scenario, the algorithm would evaluate the current value of the pixel and then examine the value of the pixel to the left in the matrix. If the previous value differs from the current value, the current pixel is counted as an edge. As aforementioned, more advanced processing techniques for edge detection have been explored and the reader is encouraged to observe the OpenCV reference documentation [49].

Control System

The aforementioned sections contained within this chapter have highlighted necessary components that the framework will use to leverage the power, processing, and capabilities of computer vision. This project extends beyond the scope of merely finding objects with a sensor and reaches the field of Electromagnetics in that the data obtained from the Kinect sensor, image processing, and nodes on the network must be integrated into an aperiodic array. Previous work on this project has already established a smartphone enabled phased array controller. A smartphone links phase, geolocation, and sensor information to an embedded phased array controller that generated phase shifts for elements within an array [17]. The architecture of the existing controller is shown in Figure 8.

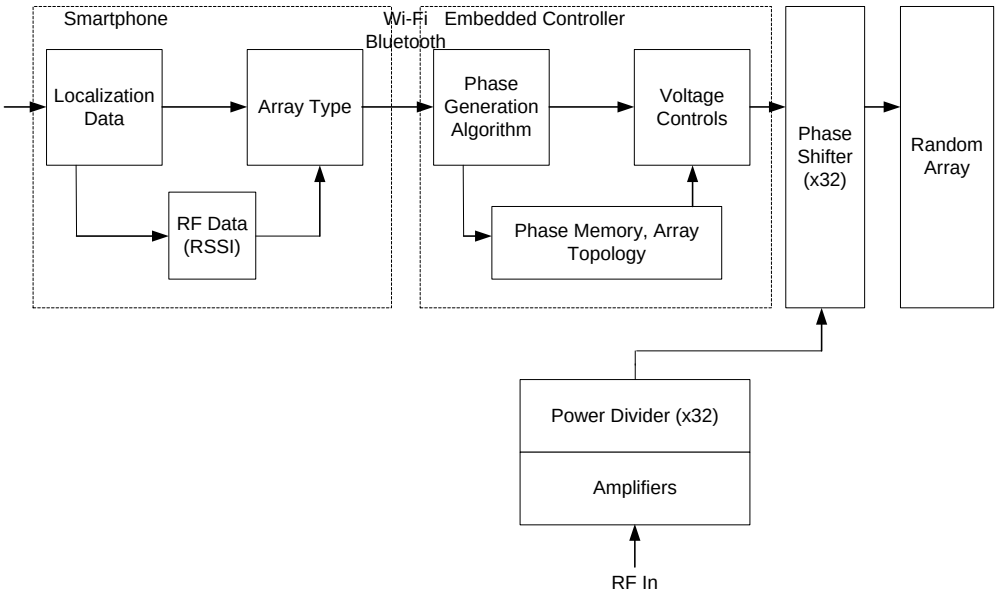


Figure 8. Existing smartphone enabled array controller architecture

Shown in Figure 8, the embedded array controller receives pertinent array information such as the array type, element position locations, and primary steering angles (θ,φ) from the smartphone. The user may select several different modes of operation depending on the desired radiation outcome of the array. Information sent from the smartphone is then analyzed on the embedded controller to generate phase information for each of the elements. A mapping algorithm converts the phase information to an analog voltage in reference to Figure 9 to produce an overall desired phase shift for individual antenna elements.

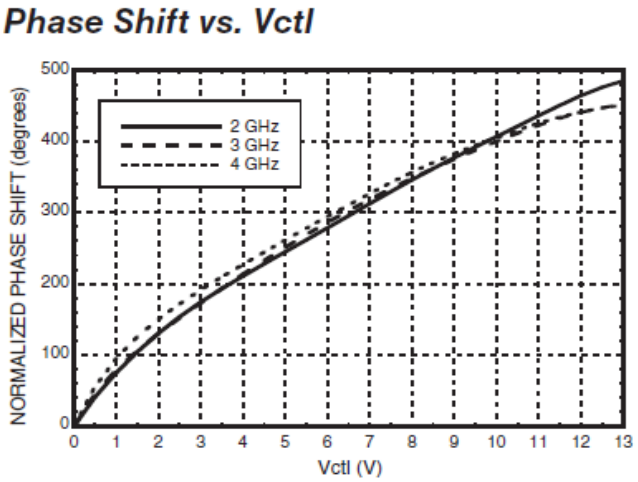


Figure 9. Hittite phase shifter voltage control versus phase shift

The first generation embedded control board could simultaneously control 16 different radiating elements and is shown in Figure 10.

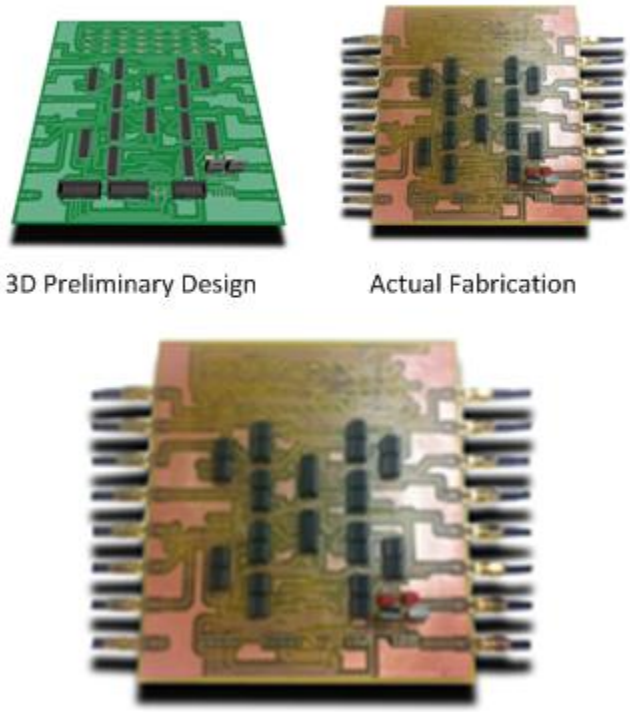


Figure 10. First phased array embedded controller design

The first generation board provided 12-bit phase shifting capability for array elements and successfully controlled a 16 element phased array. Further experiments and need for 3-D controllers required a redesigned embedded controller shown in Figure 11.

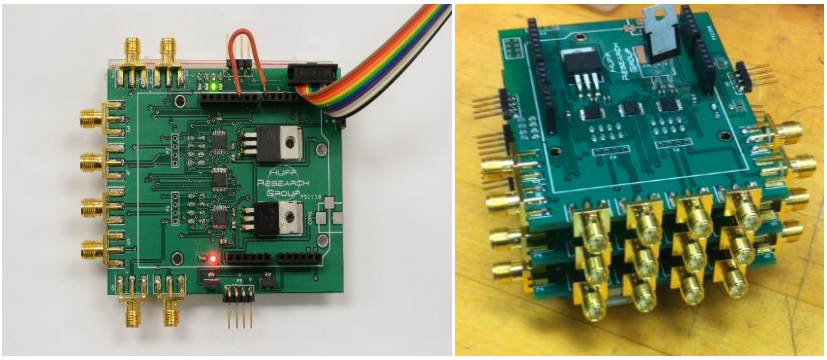


Figure 11. (Left) Single layer second generation control board (Right) stacked three layer controller boards

The second generation embedded controller contained several improvements including: functionality from a single controller, reduced size (60%), 3-D phase control, and support for 48 total elements. The controller features both a Bluetooth and hardwired serial interface to communicate with external controllers such as a Smartphone or a central base station. In this framework the central base station will leverage the existing functionality of this control system in order to generate phasing information for the nodes in the aperiodic array. The controller's capabilities will also be expanded to handle dynamic reconfiguration of element positioning in real time. Previously, the element positions were however hardcoded into the embedded controller and required meticulous hand measurement of each element *a priori* to successful array operation. This method proved to be successful in previous work where several volumetric random array experiments were conducted to produce beam steering array patterns as shown in Figure 12 and Figure 13.

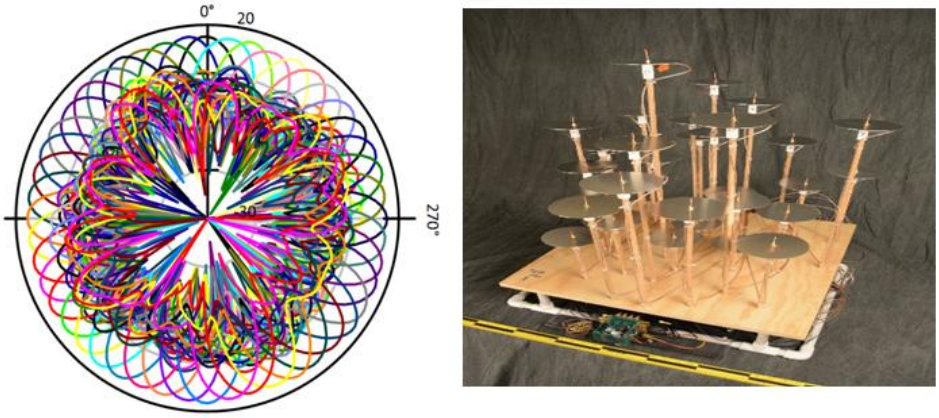


Figure 12. (Left) 360 degree scan radiation pattern of volumetric monopole array (Right) 32-element spherically bound monopole array

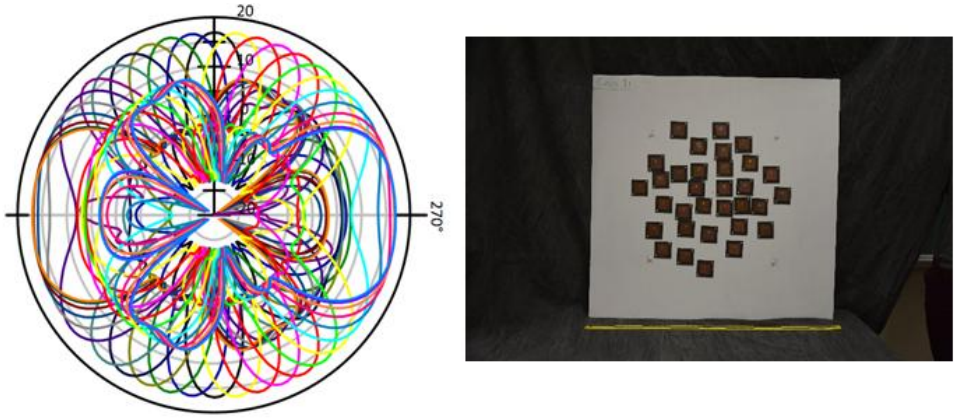


Figure 13. (Left) 360 degree scan of 32 element patch array (Right) 32-element circular microstrip patch array

Both test cases from indicate successful operation of the array controller through demonstrated beam steering capability of 2-D and 3-D arrays. Therefore, the framework will move forward with this embedded control system and expand the functionality to meet the new demands of a reconfigurable, dynamic system.

CHAPTER V

PROPOSED FRAMEWORK

The proposed novel architecture leverages the existing platform from Chapter IV to calculate element phases and wireless GPS localization capability through the smartphone. The fundamental addition is a spatial recognition and tracking component as seen in Figure 14. The spatial recognition and detection component presented in this work provide element position data to generate phasing information for the random array. Discovered elements are localized within the cluster, and can be cached to memory or communicated to the control system. Cached elements are tracked for any positional changes during array control operations and any positional change can be communicated to the controller for phase adjustment. The previous tests results [7] include an additional intrinsic delay of the RF path that had to be measured for each element. This phase error must also be included in the individual element phasing calculation as shown in (24).

$$\beta_{tot} = \beta_x + \beta_y + \beta_z + \delta_{path} \tag{24}$$

This resultant phase delay for each individual element is a combination of element position and the intrinsic RF path delay due to the amplifier, power divider, phase shifter, and cable connections.

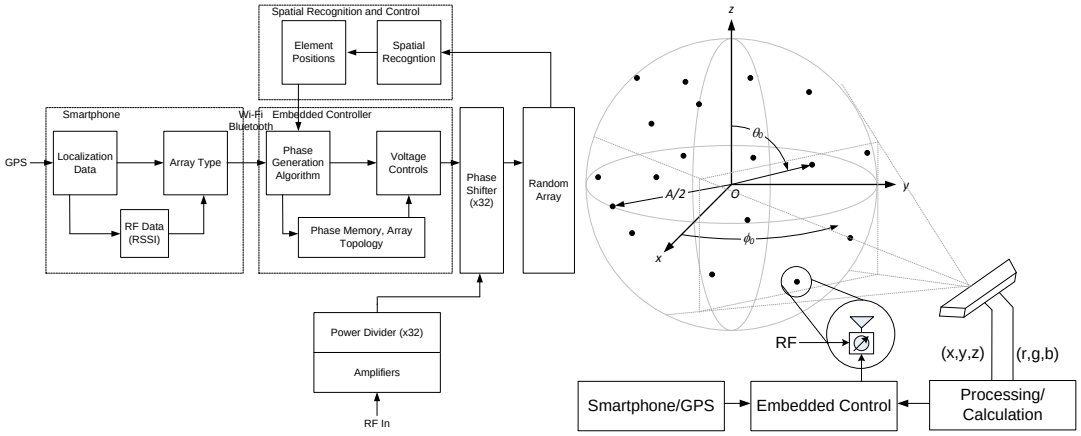


Figure 14. (Left) Novel control architecture block diagram (Right) semi-physical representation of novel control architecture

The implementation of the computer vision assisted phased array architecture requires hardware for spatial detection and recognition. As aforementioned in Chapter IV, the Microsoft Kinect has been selected as a peripheral sensor because it provides 640x480 pixel color and depth (denoted RGBD) data streams for computer-human interaction at a sampling rate of 30 Hz. The use of its proprietary pseudo-random infrared detection technique for spatial displacement measurement will provide the necessary spatial information for the framework and previous embedded control system to generate phasing information for the nodes in the aperiodic array. This final combined camera data stream provides a 640x480 matrix of data which contains a RGB 256-bit color code and a real world (*x*,*y*,*z*) coordinate for that specific pixel. This enables simplified environment awareness and object detection from both color and depth data streams. Color filtering is of particular interest due to minimally complex threshold filtration

algorithms and a natural occurrence in both color and depth images as mentioned in Chapter IV. Color filtering is also agnostic of the image generation source and can be adapted to platforms such as Microsoft Kinect, LIDAR cameras, and other high resolution imaging hardware. This also allows this framework to be used in other research arenas such as robotics, avionics, autonomous systems, and other control systems surveyed in literature in Chapter II.

Node Design

Using the Microsoft Kinect's color and data information for spatial detection requires uniquely identifiable colored targets. Specifically, antennas are predominantly constructed from metal materials like copper and lack easily identifiable features. Therefore, color enhancement to an antenna must be provided in order for the computer vision part of the framework to uniquely recognize a target. Furthermore, it is the intent of this research to explore the radiation behavior of swarming antenna arrays often found in UAV and autonomous ground vehicle application. It is also a design goal to emulate the embedded electronics and communication schemes typically found on these vehicles. However, realistic targets for this system require a functional antenna design and therefore must be designed in a way that features the electromagnetic behavior while maintaining the functionality of an autonomous vehicle.

Antenna Design

This framework will use a rectangular microstrip patch antenna design in order to study the radiation behavior of aperiodic arrays. Microstrip patches are well studied antennas and the design parameters for a microstrip patch are straightforward. The

intended design frequency for the operation of the antenna and subsequent array is 2.4 GHz. This frequency was chosen due to the convenience of operating in the ISM band and the availability of off the shelf components such as routers, radios, and modems. Furthermore, this design frequency does not permit the microstrip patch to be structurally large so that the antenna array may be tested within the anechoic chamber at Texas A&M University. The substrate chosen for the antenna design is FR4 for a couple of reasons. First, FR4 is easily available, inexpensive, and allows circuits to be manufactured quickly. Second, the antenna must share a substrate with the embedded electronics that will control lighting, facilitate communication to the base station, and read physical properties about the antenna itself.

The design properties for this antenna were calculated using well known equations and are shown in Table 1.

Table 1. Microstrip patch design parameters

Frequency (GHz)	Width (mm)	Length (mm)	Probe Inset (mm)
2.4	38	28	6.5

In order to negate surface waves and scattering off of the embedded electronic components, a quasi-PEC wall will be inserted in the form of through-hole vias surrounding the substrate area of the microstrip patch. This PEC wall has been included in the simulation to determine any scattering effects that may potentially arise.

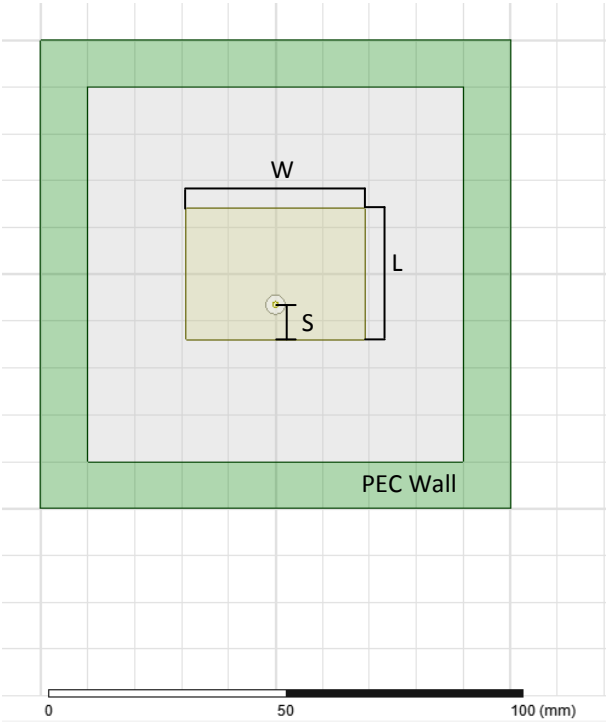


Figure 15. Microstrip patch with quasi-PEC wall

The simulation results show that the microstrip patch resonates slightly higher than the intended design frequency at 2.44 GHz, but the overall radiation operation of the patch is similar to that of a microstrip patch on a different substrate such as Rogers 6880 or Rogers 6010 as explored in previous work mentioned in Chapter IV. The radiation pattern shows typical behavior of a normal microstrip patch in Figure 16.

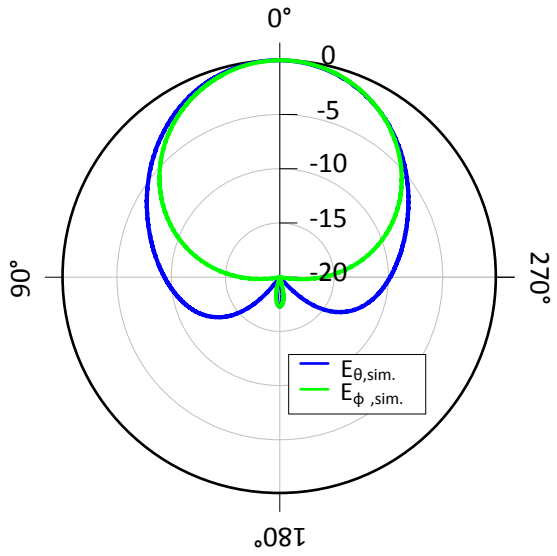


Figure 16. Simulated radiation pattern of microstrip patch

The simulation results demonstrate that the patch and quasi-PEC wall will serve as a suitable combined antenna design for the node platform.

Embedded Design

The design of the microstrip patch impacts the layout and design of the embedded electronics that will share the FR4 substrate. Primarily, the design of the embedded electronics will need to solve several problems. First, the embedded electronics will need to receive power from some source. Typically a UAV would supply power with LiPO batteries; however, since the framework is not intended to use actually flying modules, the embedded electronics will rely on wired power. Additionally, the module will need to exchange data with the central base station as it will receive commands, state information, and other network information frequently. A wireless radio could be used, but due to costs and wireless interference at the 2.4 GHz,

an optimal choice for communication is determined to be through an Ethernet medium. Because both power and data need to be delivered over a wired medium, Ethernet can provide both of these necessities through a single cable. Powering embedded electronics is commonly found in the IT arena as intelligent desk phones, switches, hubs, cameras, and many other network peripheries can be powered over Ethernet. There are official power-over-Ethernet (POE) standards, but these methods require current signaling to the power source. An easier method to deliver power over Ethernet without signaling is through power injection. The only downside to this method is that the wires used for power may not carry data in parallel.

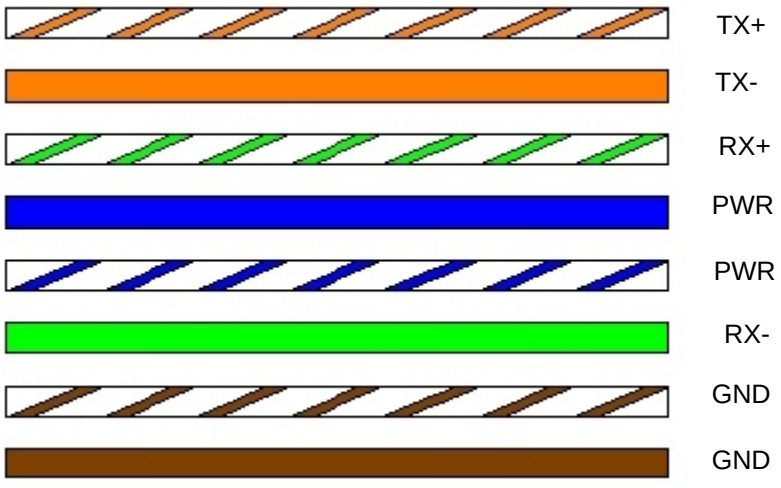


Figure 17. Wiring diagram for Ethernet cable to nodes

Figure 17 shows the blue twisted pair will deliver power to the modules and the brown twisted pair will serve as a system ground. Should the nodes or modules ever need to communicate at 1000 Mbps, power injection cannot be used to power the modules.

The most important part of the electronics are the LEDs that will make the node or antenna unique identifiable and available to the Kinect camera. The rectangular shape of the microstrip patch gives a natural suggestion to the configuration of the LEDs onboard. Placing the LEDs in a rectangle will provide the framework a couple of advantages. First, the LEDs at the four corners of the rectangle give way to simple equations to calculate the center of the microstrip patch antenna. This will be the same calculation performed on all the elements so that the calculation can be replicated easily across all nodes in the system. Second, the four LEDs allow various measurements to be calculated such as the width, length, and even rotation of the element. Because the Kinect is reading depth information, the framework will be able to read the yaw, pitch, and roll of each node. Third, because some aperiodic configurations can be tightly bound, the case can arise where nodes closer to the camera could shadow elements that are farther away preventing the Kinect from reading all four LEDs of the node. In this case, if two LEDs can be viewed, calculations can still estimate the position of the node in space.

The proposed platform combines two requirements to produce a highly configurable and extendable embedded antenna platform to emulate autonomous swarm systems shown in Figure 18.

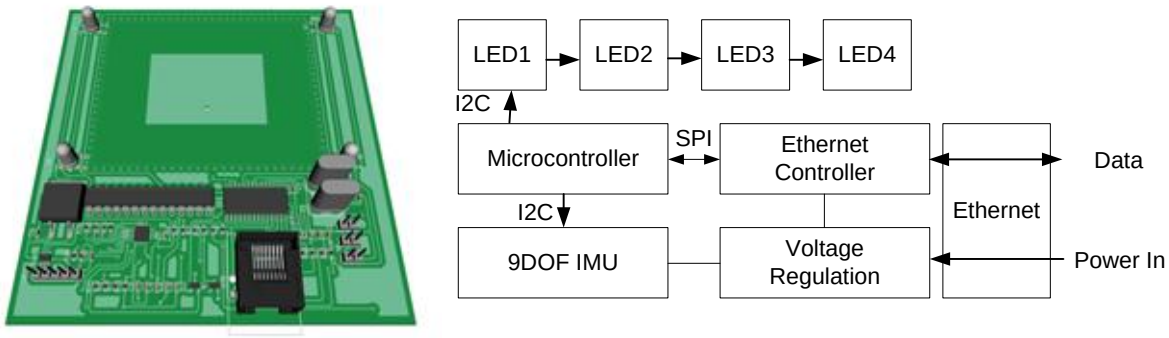


Figure 18. (Left) 3-D antenna module design (Right) block diagram for antenna module

A patch antenna on FR4 substrate (ε_r = 4.4) sits in the upper middle part of the substrate surrounded by a quasi-PEC via wall. The substrate is shared with an embedded microcontroller, Ethernet controller featuring power over Ethernet (POE), a nine degree-of-freedom (9DOF) inertial measurement unit (IMU), and four (4) RGB light emitting diodes (LEDs).

Table 2. Node embedded components and descriptions

Component Type	Component Name	Description
Microcontroller	Atmega328p	A microcontroller used to control LEDs, exchange information with the Ethernet controller, and collect data from the IMU.
Ethernet Controller	ENC28J60	An Tx10/100 Ethernet control module. Communicates with Atmega using SPI.
IMU	LSM9DS0	IMU to measure displacement, yaw, pitch, and roll.
LED	Neopixel	A small single RGB LED that requires only one data line as input.

The IMU can be used to provide additional rotation information and displacement position. This component is also found on most UAV platforms. The voltage regulator onboard receives the power from the power injector module directly through the RJ-45 Ethernet connector and converts any voltage from 12-25 Volts to 5 Volts.

Communication Design

The embedded hardware for the nodes has been presented with the inclusion of a microstrip patch antenna. It is important that a communications protocol is developed in order for the framework to communicate with the nodes. Previously, a communications protocol was developed to communicate with the embedded array controller. A similar protocol emulating the protocol used to exchange data with the embedded hardware platform keeps uniformity and simplicity within the framework.

Figure 19. Framework packet structure

The structure of the datagram allows for bi-directional information exchange using a series of commands that may be originated from either the base station or the client nodes as shown in Figure 20. A description of the individual structural components is described in Table 3.

Table 3. Description of framework packet

Name	Length (Bytes)	Description
Start byte	1	A unique byte used to indicate the start of a data frame
Length	2	Two bytes that indicate the length of the data in this datagram
Command	1	A unique identifier that indicates an action on the base station or client device
Data	Max 1440	The data being sent
Checksum	1	A checksum that indicates the validity of the data transmitted

The startbyte, length, command, and checksum have static length values that determine how much of the datagram is used. The data, or payload, may be variable in order to meet the data requirements of the various commands. The checksum is based on a modulus 256 checksum and is calculated according to (25) where χ is the checksum and L is the length byte of the datagram.

$$\chi = \mathrm{mod}\left(\sum_{Data[m]}^{Data[L-1]} (0x00FF \,\&\, data[m]), 256\right) \mid m = 0..L-1 \tag{25}$$

The command byte requires that the framework supply a unique command to indicate a desired configuration, update, or execution from either the base station or the client node. The commands and their descriptions are listed in Table 4.

Table 4. List of commands for nodes and base station

Command Name	Base	Identification Code	Description
Lights Solid Color	Client	0x00A0	Lights on client will turn solid color
Lights Sparkle	Client	0x00A1	Lights will sparkle desired color indicated in data
Lights Individual Color	Client	0x00A2	The four LEDs onboard client will turn individual colors according to data
Get All Information	Base	0x00E0	Get all client information on network
Get MAC Information	Base	0x00E1	Get MAC address information from specific node
MAC Response Ok	Client	0x00E2	Client responds that it received the specific request for MAC
MAC Response Error	Client	0x00E3	An error happened on the client in response for MAC request
Get IP	Base	0x00E4	Get IP address of specific client
IP Response OK	Client	0x00E5	The client successfully received the request for IP address
IP Response Error	Client	0x00E6	There was an error in processing the request for IP address
Get Pixel Information	Base	0x00E7	Request the colors of specific pixels on a node
Get Yaw, Pitch, and Roll Information	Base	0x00E8	Request the yaw, pitch, and roll information

The commands allow the base station and nodes to communicate with one another to determine various system states. These system states will be discussed further in the software development of the base station and nodes.

The communications within this framework references the standard seven layer OSI communications model. The first layer, or physical layer, for this framework is Ethernet. The second layer, or link layer, heavily utilizes ARP and MAC address discovery to become aware of active nodes within the network. The third layer utilizes standard IP header information for routing layer 2 frames into layer 3 packets. A router is not needed on this system as the multicast framework distributes the information and messages to all nodes within the framework; however, a router is useful for attaching third party peripherals and other non-standard nodes (like Smartphones) to the network. The fourth layer utilizes UDP communication instead of a common TCP style of connection. The reason for using UDP in this framework is to leverage the speed that UDP provides in contrast with TCP. Because most of the communications will occur on a local subnet and will not be routed across some type of WAN connection, UDP will provide simple layer 4 mechanisms to exchange data between nodes, the base station, and other peripheral devices. The base station acts as the UDP server and all other nodes and periphery will subscribe to the server as clients to exchange data.

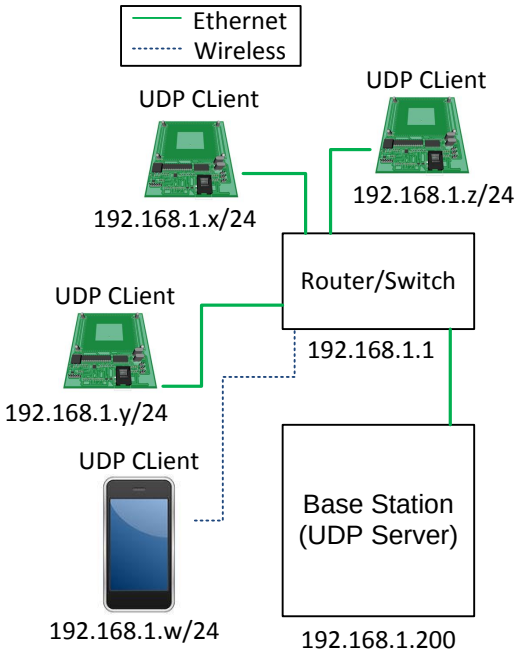


Figure 20. Network communications setup for framework

The base station will have a fixed IP address that all the nodes look for when joining the network. The base station also has the default gateway for the network set to the primary router or switch that connects the nodes together. It is preferable to have a wireless router in order to attach additional periphery that was utilized in the previous work as shown in Figure 20. The *w*, *x*, *y*, and *z* shown in the picture are placeholder variables for unique integers that span the range from 2-254, with the exception of the number 200, as this IP address is already being used for the base station.

Node Software Design

The software of the node needs to continually recognize commands from the base station while observing the state of operation locally from the IMU, LEDs, and

general information that it is collecting and receiving. The microcontroller fundamentally runs on a single thread of software that fundamentally bounds how many states the microcontroller can simultaneously process. Because of this restriction the states will be achieved by caching memory states and processing series of "if" statements throughout the thread of the controller.

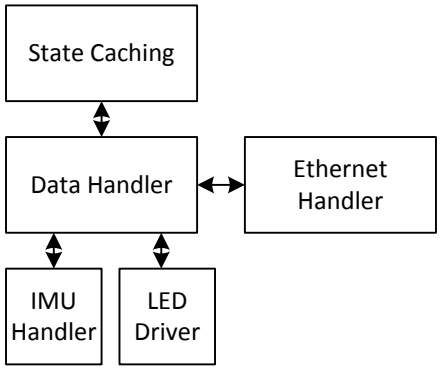


Figure 21. Software architecture of node

The Ethernet handler shown in Figure 21 receives all commands and packets from the base station and other nodes. It processes the packet, ensures data integrity, and then passes the data onto the data handler. The data handler caches the packet data and talks with the local system to determine the state that the microcontroller is in.

Default State

The default state of the controller is to listen for commands from the base station and to turn all four LEDs to their start state (color red). In this state, the nodes are not

talking with other nodes and listen for commands from the base station to determine what state the framework is in.

Scanning State

The framework is attempting to find the position of the node and the node is either showing its unique discoverable configuration or all the LEDs are turned off. This state is only temporary and the node returns to the default after the scan state of the base station has completed. The base station informs all nodes to return to default state after the scan state of the framework has been completed.

Autonomous State

This state allows the node to be self aware. The microcontroller continuously is reading commands from the base station and also interrogates the local IMU about any movements that the node is making. This interrogation can prompt data to be sent back to the base station to inform the base station about its movements or internal state changes that have happened.

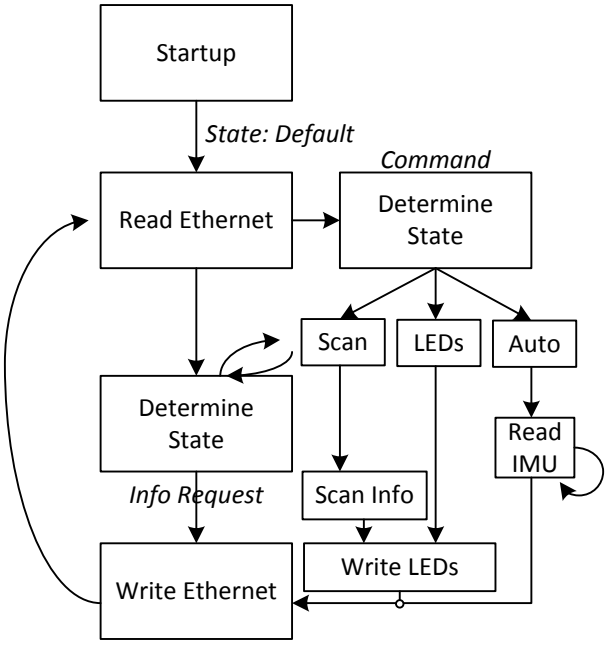


Figure 22. State diagram of node

Figure 22 shows how the node can process information from the base station and achieve the aforementioned states. Note that the incoming packets from the base station are always read after processing either a command or an information request. A command has a specific header that was mentioned in Table 4, and is used to update a physical parameter on the node. For example, all command packets change the state of the node, change the current LED configuration, or enable the node to read the IMU continuously. An information request does not change the current state of the node but rather fetches the desired information from the base station, writes this to the Ethernet, and then returns to the incoming Ethernet buffer to look for new packets from the central base station. Any “scan” or “LED” command ultimately ends by reconfiguring the

current state of the LEDs. This is because scan mode requires the LEDs to turn on for the Kinect to find the node or to turn off so that the Kinect does not misread the current node. The “LED” command changes the configuration output of the LEDs regardless if the module is in a scan mode. The autonomous mode recognizes any physical movement of the node and reports this back to the base station automatically, but does not reconfigure any of the LEDs.

Base Station Software Design

A software base station is essential to overall array behavior and performance. Randomly distributed node networks must have one master or designated controller that assign system wide functions and behaviors for the individual elements. Thus, the software architecture is comprised of four separate stages: calibration, acquisition, calculation, and reaction. The first stage requests system calibration of the visual device to minimize measurement error in high noise environments. Subsequent spatial calibration of the Kinect can be run to remedy erroneous results or to reset error bounds used in position detection and estimation.

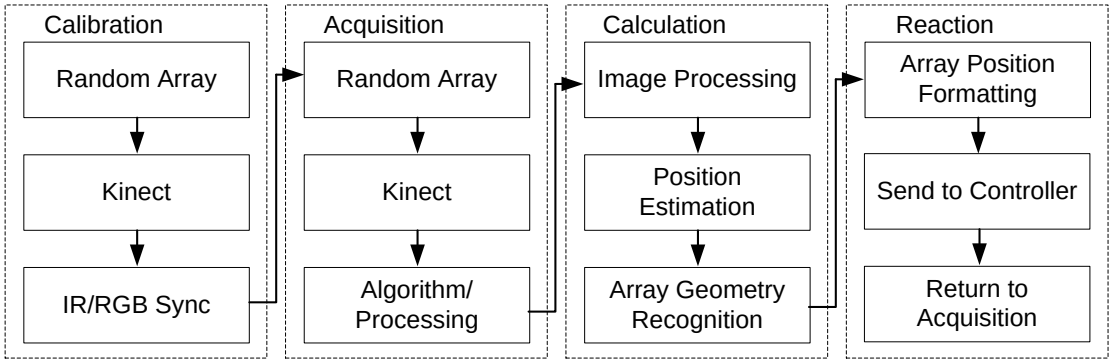
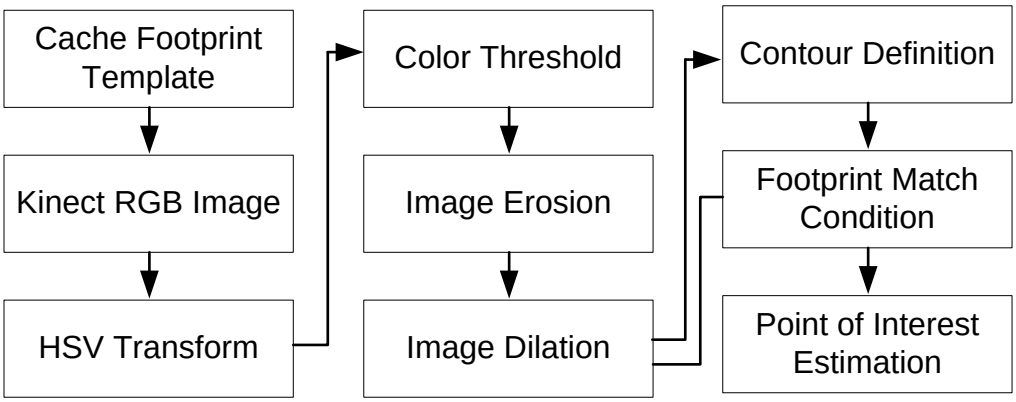


Figure 23. Block diagram for system software architecture

The acquisition stage defines an interface to gather visual information from the visual source to be processed in the object detection algorithm contained within the calculation stage. Multi-threaded calculation processes also spawn to extract information from color and depth images to be analyzed in the calculation stage. Threading is needed due to heavy demand of computing resources used during image processing techniques. The calculation stage processes image data acquired by the acquisition stage using image filtering techniques such as color threshold, dilation, and erosion. The position detection and estimation calculation is dependent on pattern recognition in the RGB image in this system. The successive image processing steps are seen in Figure 25.



Pseudo Code

```
Footprint = loadImageFromMem()
Mat[640*480] = kinect.Image()
HSV = RGB2HSV(Mat,HSV)
Opencv.threshold(HSV,high,low)
Opencv.load(HSV)
Opencv.erode()
Opencv.dilate()
While(!footprintMatch):
        opencv.contours()
        if(countour.width < 7mm)
                break
        else
        Opencv.dilate()
```

```
Footprint2 = Opencv.getImage()
Compare(Footprint2,Footprint)

If(match):
depthField = kinect.depth()
Pixels[] = mapPOItoDepth(Footprint2)

//Go to antenna estimate antenna
//Location
```

Figure 24. (Top) Image processing progression steps (Bottom) Pseudo code for image processing

Pattern matching is essential to object discovery and tracking. The proposed image processing algorithm in Figure 25 caches a desired footprint or pattern that is continuously analyzed in each RGB frame by the system. The proposed pattern footprint contains two large white areas housing four small areas that represent the four LED locations on the modular antenna shown in Figure 25. The algorithm caches an RGB image and performs a common HSV color space color transformation. The HSV color space is a cylindrical transform that represents the luminance of colors that are observed

in an image by incorporating the value (V) or brightness of the color. This algorithm uses color thresholds that have been empirically discovered due to varying brightness of LEDs on the antenna module. Unique color configurations for each antenna module require unique color thresholds bounds. Small threshold bounds permit greater use of colors of the same spectrum and reduce ambient color noise in the binary image. The produced binary image may still contain undesired noise from the surrounding environment that can be eliminated by eroding the image. A loop then continues to dilate the image until the black spaces representing the pixels have a matched depth location of 7mm or less. An average footprint with standard deviation is shown in Figure 25. This measurement is within 1mm of the actual physical LED size of 5 mm.

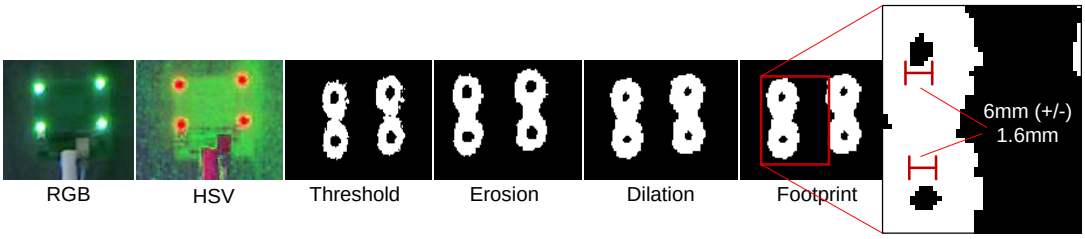


Figure 25. Image progression for detection algorithm

Successful recognition yields a specific point of interest (POI) to be analyzed from the depth image. Cross correlation of the POI in the depth image finds the estimated location from the depth image. The position in Euclidean space is saved in storage for processing in the reaction stage. This stage is comprised of a control and data layer which

communicates with the existing controller to calculate phases for each array element. Using this method for node detection, the framework can recognize five unique points on a single node to determine the final positioning of the element.

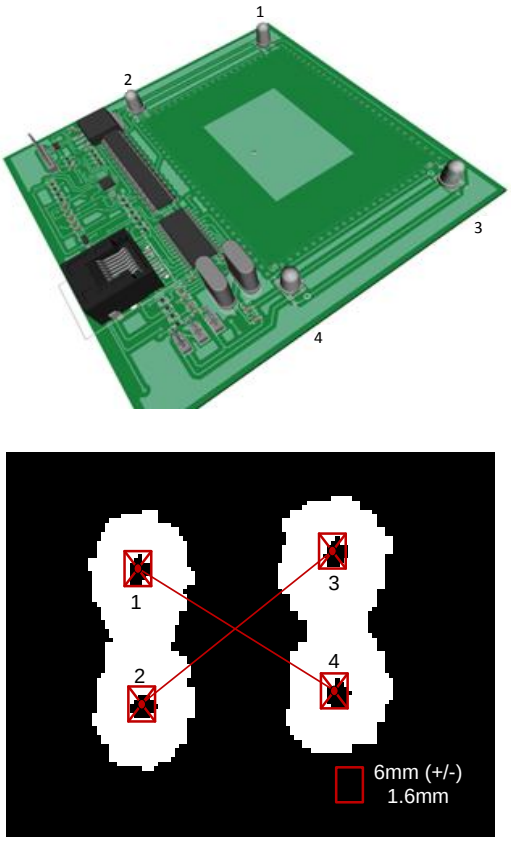


Figure 26. (Top) 3-Dimensional antenna representation with pixel numbering (Bottom) filtered image of antenna with pixel numbering

As shown in Figure 26, each of the 4 pixels of the node produce a point in Euclidean space and a final averaging of these points produces the center of the antenna element.

The pixels can be identified uniquely by a color that is flashed or by the 3-Dimensional position in space.

Figure 27. Unique color configuration identification of a node

If the color image is not used then a simple algorithm to determine the locations of the pixels is used. Because the Kinect treats the center of the RGB image as the origin or *(0,0)*, the pixel locations can be easily determined. Because the Euclidean Y value increases as the pixel value of the RGB image decreases and the Euclidean X value increases as the pixel value increases, it can be determined that pixel 2 should always have the smallest value and pixel 3 should always have the largest value.

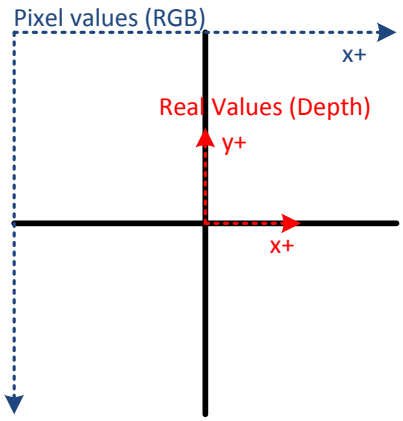


Figure 28. Kinect coordinate system and mapping

In mathematical form, the pixel 2 and pixel 3 can be defined with convenient expressions in relation to their real Euclidean coordinates. The functions provided in (26) are in many well known C++ and java libraries.

$$\begin{aligned} p_2 &= \min(X, Y) \\ p_3 &= \max(X, Y) \end{aligned} \tag{26}$$

X and *Y* represent vectors of all four respective Euclidean pixel locations. Thus, the maximum and minimum of all *x* and *y* easily determine the locations of pixels 2 and 3. To determine pixels 4 and 1, a simple if statement is needed based on the real value coordinates of the pixels given in (27).

$$if\,(pixel1.y > pixel4.y)?leave:swap \tag{27}$$

Since the pixels are prearranged in vectors, the *if* statement compares the y-values of the pixels in the vector. If the *y* value of the pixel currently located in position 1 is greater than the *y* value of the pixel located in position 4, the pixels remain at their respective

locations. If the statement is false, the pixels switch positions in the vector. This organization of pixels then allows for a convenient calculation to determine the 3-dimensional center point of the antenna as shown in Figure 26.

$$L(x,y,z)_{center} = ((L_1(x,y,z) + L_4(x,y,z))/2 + (L_2(x,y,z) + L_3(x,y,z)/2)/2 \quad (28)$$

The equation above shows an averaging of the cross-diagonal vectors of the pixels at each corner averaged with one another. This average then represents a point that lies on the plane of the antenna regardless of rotation in space. This method is then applied for all antennas or nodes in the framework ultimately yielding positions for each element.

Phased Array Controller Design

After the positions for each element or node in the framework have been determined, the phased array controller system needs to receive this data and reprogram the phase shifter network to the correct values. As aforementioned and cited by the authors from previous work [7, 17], the values for elements used to be preprogrammed into the controller to calculate correct phasing for each element. The phased array controller already contained a built in protocol to communicate phase and steering information, and therefore can be utilized to create messages to dynamically program positions into the controller. First, a frame tells the embedded controller that it will be running in a "*dynamic coordinates*" mode.

Start byte (255)	Length (0)	Command Dynamic Coords	Data (0)	Checksum

Figure 29. Dynamic coordinates frame

Three vectors for 32 *x*, *y*, and *z* positions have been created to store the dynamic coordinates for each element. Once the command has been sent, unique frames holding the *x*, *y*, and *z* positions of each element may be downloaded into the vectors on the controller.

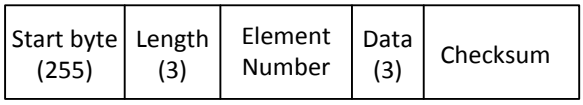


Figure 30. Frame for downloading dynamic positions to controller

Figure 30 shows the frame that is used to download the coordinates into the controller. An element number in the middle of the frame ensures that the coordinates are placed in the correct position on the embedded controller. Once the frames in Figure 29 and Figure 30 have been sent, the controller needs to receive a "disable" frame in order to stop using the vectors with the dynamic coordinates. The rest of the code and functionality of the previous phased array controller is preserved.

CHAPTER VI

SYSTEM FABRICATION AND TESTING

This chapter is broken into two major sections. The first section covers all information about the fabricated framework including verification tests of the Kinect, antennas, and respective software for the framework. These designs will support the previous sections by providing actual measured data and demonstrate successful fabrication of elements and software through figures and other tests. The second half of this chapter will cover a series of experiments that demonstrate the capabilities of the framework to measure aperiodic arrays, various configurations of aperiodic arrays, and aperiodic arrays acting as swarms.

System Design

This section contains the designed and fabricated parts of the framework proposed in chapters IV and V.

Antenna and Node Fabrication

A single antenna module (i.e. node) was first fabricated to determine the functionality of the microstrip patch on the FR4 substrate material. Three separate design iterations were performed in order to minimize the total footprint of the node to prevent pervasive shadowing in the aperiodic array. The results of the final iteration of the antenna are shown in Figure 31.

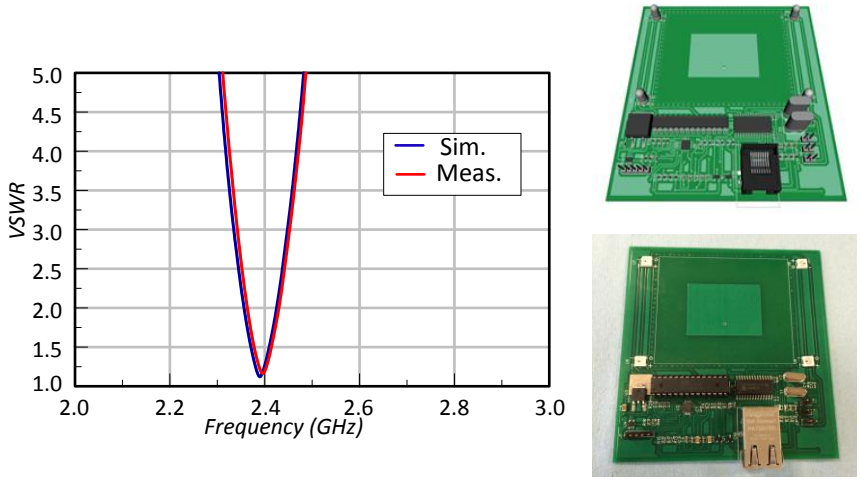


Figure 31. VSWR of simulated and fabricated antenna modules (Upper Right) Simulated antenna (Lower Right) Fabricated antenna

Measured antenna results compared to simulation show a shift of approximately 2 MHz in frequency. The impedance result shows a disparity outside of the resonant frequency due to an initial offset that was not subtracted in the calibration. However, the shift in frequency is still within our desired band (ISM band) and is therefore still useful to move forward in producing the rest of the nodes that are needed in the framework. The antenna module's radiation pattern was then measured to verify proper microstrip patch antenna radiation behavior and to compare the radiation behavior with the simulations performed in Chapter V.

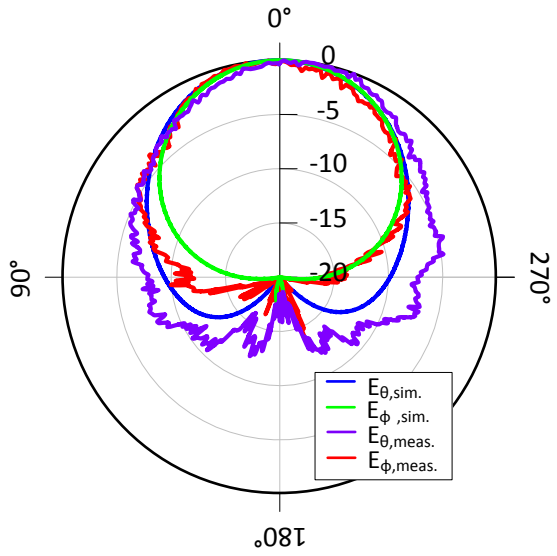


Figure 32. Normalized simulated and measured radiation pattern for antenna node

The pattern in Figure 32 shows radiation behavior consistent with a patch antenna. The results have been normalized to show alignment of pattern behavior rather than total gain of the antenna. Because the antenna was fabricated on FR4, which is a considerably lossy substrate, the gain of the antenna was not deemed as important for the framework in comparison with the pattern. It should be noted that the measured radiation pattern appears to have a substantial amount of noise that is observed on both the theta and phi angular measurements. This is due to an improper calibration of the gain in the anechoic chamber at the time of measurement.

The successful measurement of the antenna node then confirmed a batch fabrication of 20 elements. The successive fabrication of each element required calibration and measurement similar to the singular measurements of the first antenna

module. The goal in measuring all antennas is to ensure that all antennas perform approximately at the same frequency.

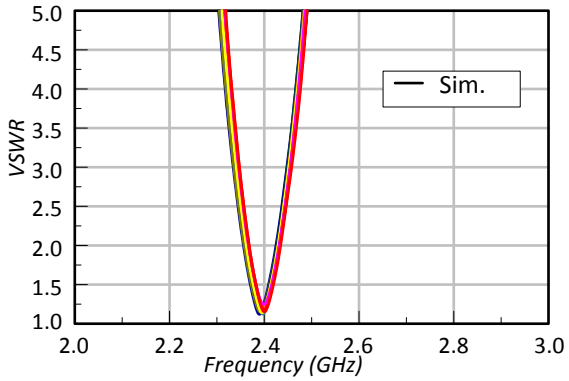


Figure 33. VSWR comparison of fabricated and simulated antenna nodes

All fabricated elements were tested for their VSWR. In comparison with the first fabricated element, the batch of 20 elements had a slightly shifted resonance down to 2.4 GHz. Because the framework will only use 16 elements, the elements with the closest VSWR measurements were selected to be used for array measurements. All sixteen VSWR measurements are shown in Figure 33, with the simulated measurement highlighted in black.

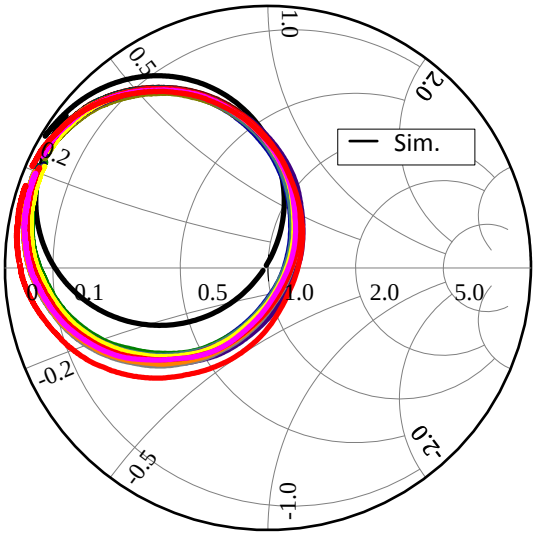


Figure 34. Smith chart of simulated and fabricated impedance measurements from 2 to 3 GHz

A measurement of the fabricated impedances over the 2 to 3 GHz range shows that nearly all 16 elements have the same impedance at resonance and then slightly deviate at higher frequencies. The measured impedance compared with the simulated impedance is more visible on this plot than with the VSWR plot as the simulated impedance is near 1.0 at resonance. Altogether, these measurements show good alignment among the fabricated antenna elements to be used within the framework and demonstrate good correlation to the simulated version of the antenna as the difference in VSWR is approximately 0.25 at resonance. The measurement of radiation patterns for all elements was not conducted based on the impedance and VSWR results. It was concluded that the elements aligned well with these measurements and that the radiation patterns would also align.

Base Station Design

The entire framework depends on the functionality of the base station or central control station. All the elements take commands from the base station about their current state, the base station uses the Kinect to find the elements in Euclidean space, it takes commands from 3rd party devices such as Smartphones, and it communicates the array configuration with the phased array controller in order to produce the desired radiation behavior. The proposed design in Chapter 5 uses a series of commands to communicate with the nodes in the framework and the embedded controller. In addition, the proposal included image processing functionality to find the nodes in Euclidean space. A base station was built in Java through the Processing interface using Processing, OpenCV, and OpenNI libraries that were mentioned in Chapter III.

Core Components

The base station software is comprised of several different interfaces and handlers to meet the system demands that are required in terms of communication, image analysis, and algorithmic processing. The software can be broken out into frontend and backend components. The backend handles most of the image processing and data handling, and the front end handles all of the user interface and interaction.

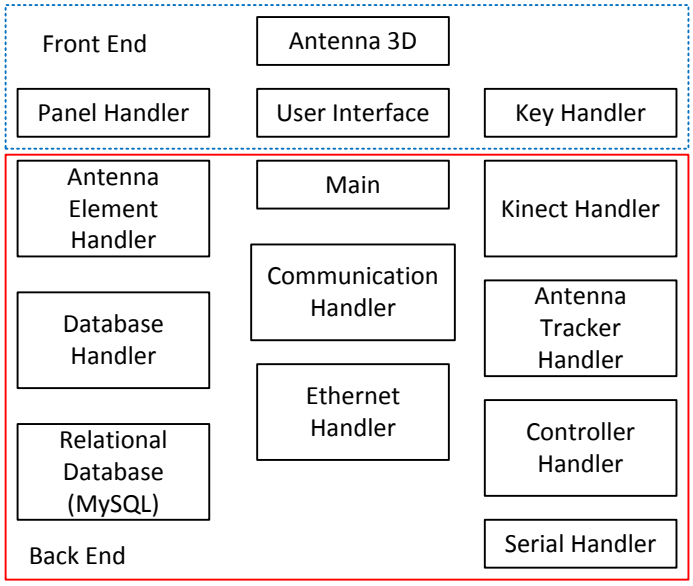


Figure 35. Core components of base station for framework

The core components of the base station are shown in Figure 35. The individual purposes for each component are listed in the table below.

Table 5. Core software components of base station

Component Name	Description
Antenna3D	A base class for the 3-Dimensional UI that shows the antenna nodes in space during runtime
Panel Handler	Part of the main class that handles all of the interactions with the primary user panels of the UI
User Interface (UI)	The main user interface of the program containing all user interactions and displays of information
Key Handler	A class for handling user input through keyboard keys for shortcuts and quick actions
Antenna Element Handler	The primary interface between the UI and the relational database that caches the all nodes (antennas) and their respective states
Database Handler	An interface and handler for communicating with the antenna element handler and the relational database implemented in MySQL
Relational Database	A database schema used for caching all data retentive memory
Communication Handler	The handler and interface for the UI to send Ethernet messages to respective nodes and to read messages from nodes on the network
Ethernet Handler	An interface to communicate with the UDP server implemented on the base station
Kinect Handler	The primary interface for the UI, main program, and other modules to the Kinect for real time image and depth information
Antenna Tracker Handler	A handler used to analyze real time images from the Kinect in order to find the position of nodes in space
Controller Handler	An interface to the phased array controller from the main user program
Serial Handler	A handler that communicates with the phased array controller over serial or Bluetooth and handles messages to and from the main program

The main program starts by starting several of the main classes on program entry. Naturally, the program starts the user interface and some of the other modules that display information on the user interface such as the panel handler and key handler. The program starts through the main class entry point. The main thread of the program opens 5 subordinate classes on the front and back end.

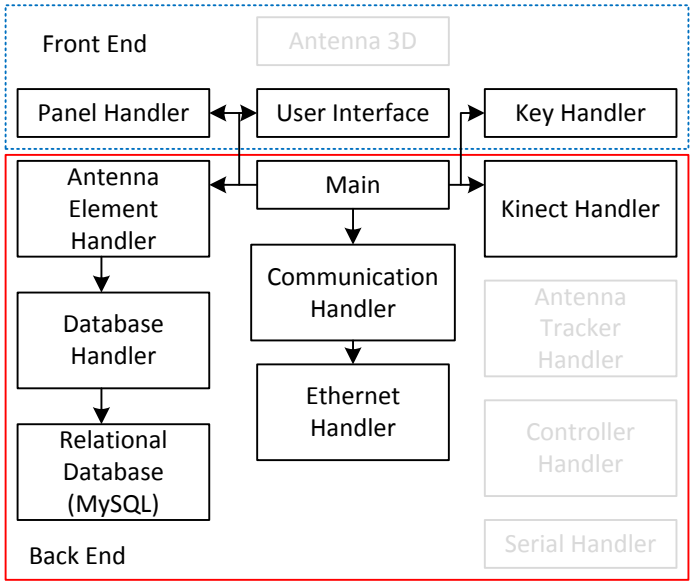


Figure 36. Classes active on program open

The main class calls the user interface first to load panels from the panel handler. The panels contain several different types of controls that will be explained shortly. The main class also calls the antenna element handler that calls an instance of the database handler that connects to the relational database implemented in MySQL. Once the database has

been loaded, the panel handler asks the antenna element handler for the most updated information from the database to display in the panels. The main display also calls the Kinect handler which checks to see if a Kinect is physically attached to the base station and informs the rest of the program if a Kinect exists. If a Kinect is not plugged in, the system is made aware of this and the user interface informs the user that no Kinect has been attached. In this case, the modularity of the core components can be changed to adapt to many different types of peripheral devices as discussed in the literature review in Chapter II. The core components of this base station in this framework allow for pieces to be added or removed without affecting the overall performance of the framework.

User Interface

Once the main thread of the program has called the user interface and loaded the core components, the system relies on the interactions of the user interface to control the framework.

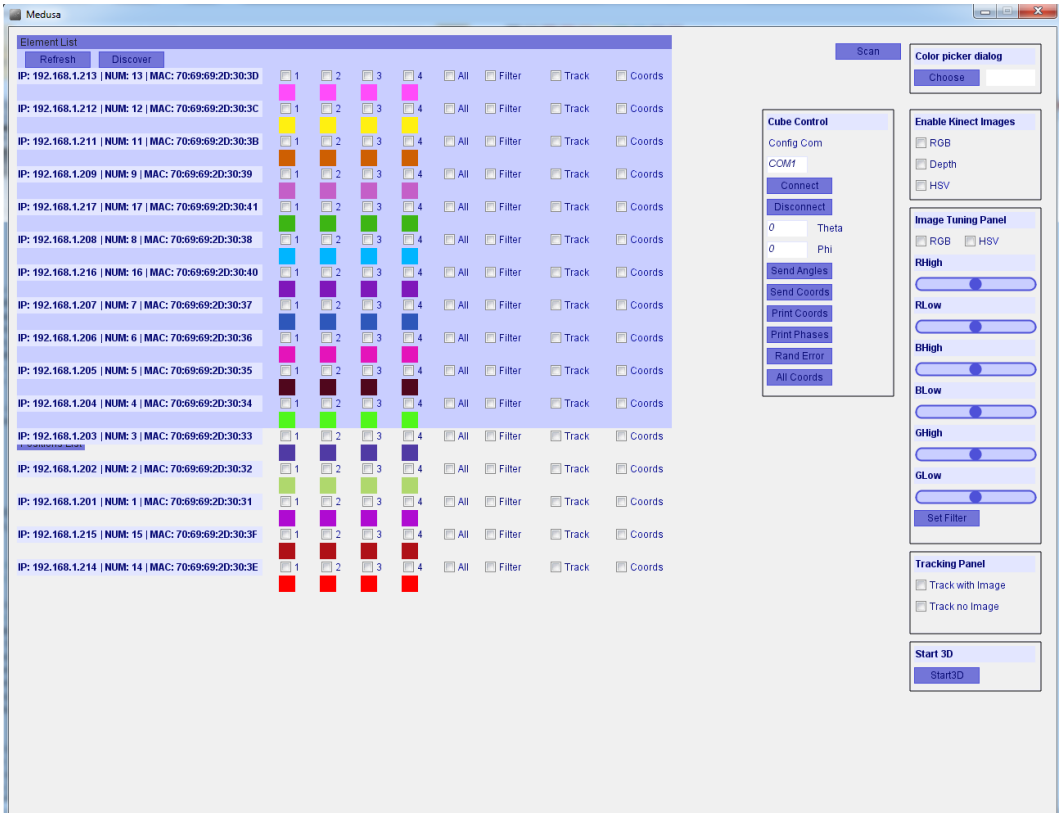


Figure 37. Opening of user interface program

The first glance into the user interface shows several different panels, and a list of information including IP addresses, MAC addresses, checkboxes, and color boxes. To begin, the first panel named “Element List” shown in Figure 37 contains an active list of all the elements or nodes that have been saved in the relational database. The IP addresses, MAC addresses, and antenna element numbers are all listed for the user to identify specific elements. The previous states of the four pixels on the antenna are listed under checkboxes. These checkboxes are followed by four additional checkboxes that will be explained in the next section called “user controls”. In the upper right corner of

the main window in Figure 37, there is a panel labeled “Color picker dialog” that allows the user to select specific colors to send to the antenna nodes. Below this panel is a panel labeled “Enable Kinect Images” that allows a user to visibly toggle the RGB and depth feeds from the Kinect if one is connected. Below the Kinect panel is an “Image Tuning Panel” that allows a user to tune the images for certain filter settings. Next, the panel below labeled “tracking panel” allows the user to enable a particular mode of antenna tracking. The panel below the tracking panel labeled “Start 3D” starts a new user interface with the 3D Antenna module enabled allowing the user to view the entire spatial distribution of the antenna nodes in 3 dimensions. To the left of the “Enable Kinect” panel is the “Cube Control” panel that allows the user to interface with the phased array controller. A button labeled by “Scan” sits next to the color picker panel and is used to initiate a full scan of the aperiodic array.

User Controls

This framework is assisted by computer vision and is a tool to better understand, measure, and interact with aperiodic arrays. The system is not fully autonomous and requires user control and interaction in order to function properly. The user-provided intelligence helps the computer vision handler scan, find, and track the antenna nodes as they move through space.

The first user control is through *unique node control* that is performed by varying colors of the individual elements. Although the system can randomly assign individual colors to the nodes, it can also be manually done by the user through the user interface.

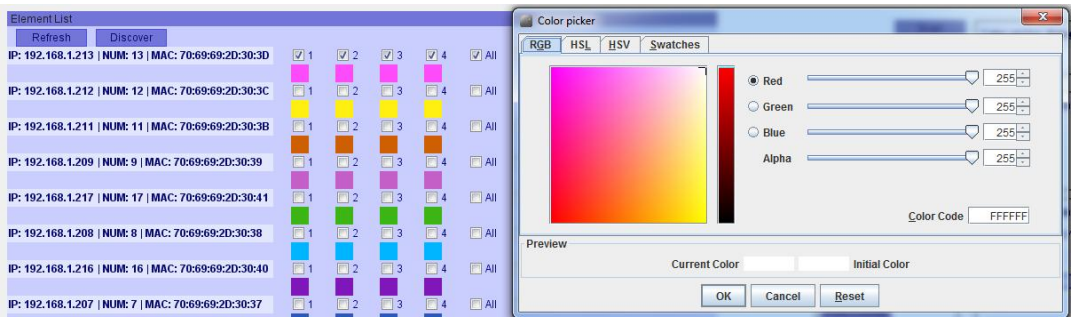


Figure 38. Individual color control over elements

As shown in Figure 38, the user has the option of individually selecting the pixels of an antenna node or selecting all of them. The user can then open the color picker dialog and choose a 24 bit color to send to the antenna node using the communication handler by selecting ‘OK’ after the color choice. This capability allows the user to select unique colors for performing the test and to visually determine which node exists in Euclidean space from the human eye.

The next user control allows the operator to visually see what the Kinect sees through the Kinect enable panel. If a Kinect is attached, selecting the RGB, depth, or HSV checkboxes will display a live feed of these images to the user through the user interface. Enabling one of these checkboxes calls the Kinect handler and enables the Kinect handler to provide input to the user interface. Once the Kinect has been enabled to communicate with the user interface, the user may then use either the tuning panel or the tracking panel to allow the image processing libraries and the algorithms developed in Chapter V to operate on the images. The user must first hit the checkbox for “Filter”

for a specific element in the “Element List” as shown in Figure 37, and then, using the tuning panel, the user can set individual filter schemes that the system recognizes during a scan to find and track a specific element. The tuning panel prompts the user to select a color space, RGB or HSV, and then provides a series of tuning sliders to filter out the image to a desired footprint. Once the footprint desired footprint has been achieved, the user can save this information into the database schema through the database handler.

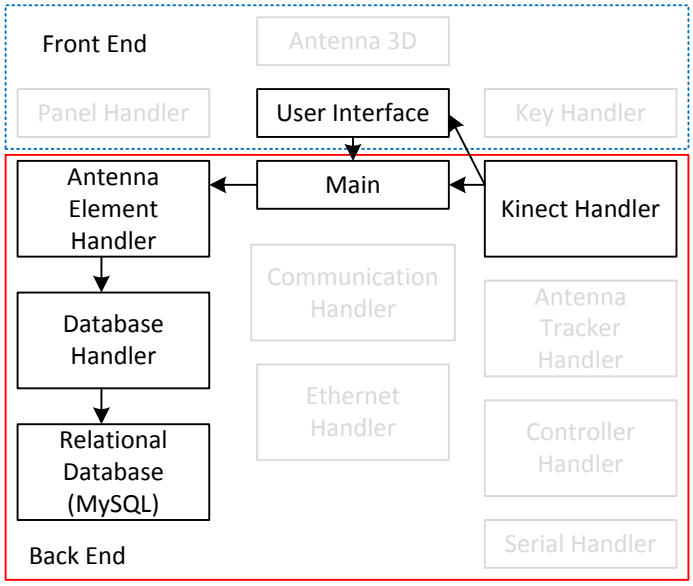


Figure 39. Components used to tune images for specific filtration settings

Note that in Figure 39, the Kinect handler provides both the Kinect image and simple filtration algorithms using the OpenCV library. The Kinect handler then coordinates with the user interface based on user input to inform the element handler about a specific

filtration setting. The element handler then stores this information in the local database to be used during an element scan.

Tracking is performed once all the elements have been configured for a proper filtration setting. The user may individually track the elements using the "Element List" shown in Figure 37 by selecting the "Track" checkbox, or the user may hit the scan button to automatically scan all elements that are currently in the database. At the beginning of a scan or individually tracking an element, the main program calls up the antenna tracker handler. This particular module contains all the necessary algorithms for tracking the antennas, as well as, a connection to the database to read out the filtration codes that were previously saved. The antenna tracker then communicates directly with the Kinect handler to feed live images of the aperiodic array.

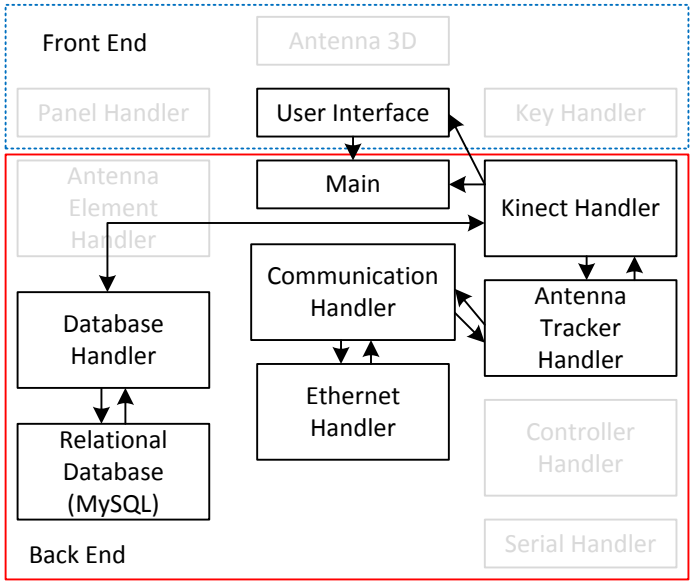


Figure 40. Components required to scan nodes

From Figure 40, many components are actively working together to filter the images provided by the Kinect handler. The antenna tracker communicates with the communication handler that promptly sends out color configurations for each node during a scan. These color configurations are read directly from the relational database through the Kinect handler.

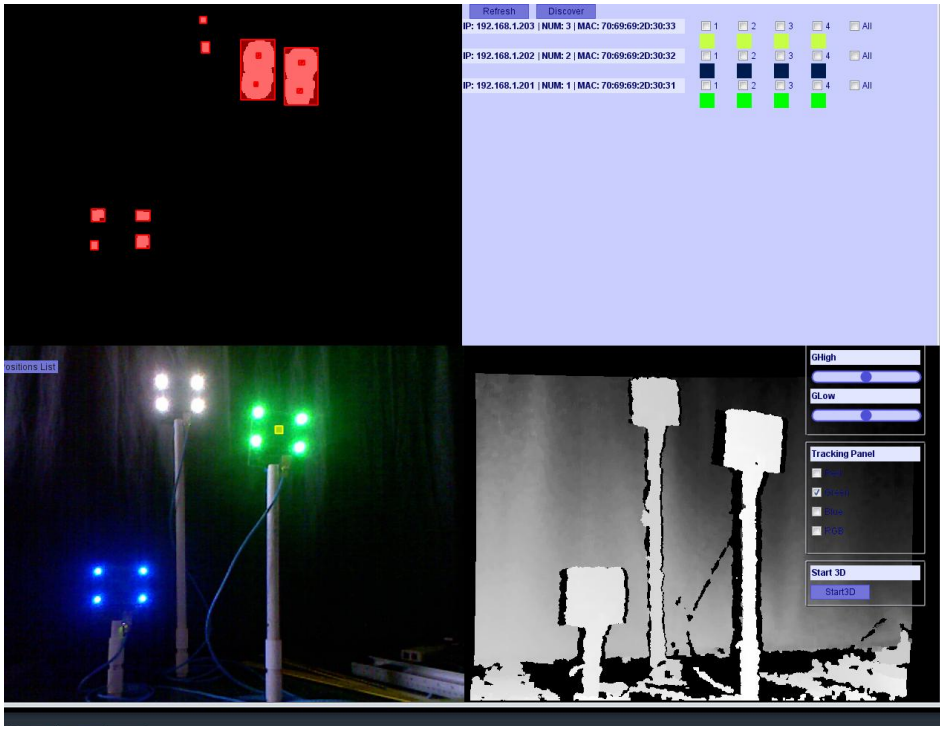


Figure 41. Active scan for nodes

Figure 41 shows the user interface during an active scan mode. The element being scanned is the one that has all four LEDs configured as green. The upper left box of the user interface shows a filtered image with red rectangles encompassing white ‘blobs’

that represent the filtered light as discussed in Chapter 5. The algorithm to determine these red blobs will be discussed in the "enhanced filtration" section of this document. In the lower left box, a small yellow rectangle marking the center of the antenna is shown. The width and height of the yellow box are determined from a calculation of uncertainty in the measurement that will also be discussed in the "enhanced filtration" section.

Once the scan has completed, the element positions are placed in a relational database that saves the positions of the elements with each scan through the antenna tracker handler. The 3D antenna module may now be opened to view the antennas in 3D to better help the user determine if any errors were received in processing the elements.

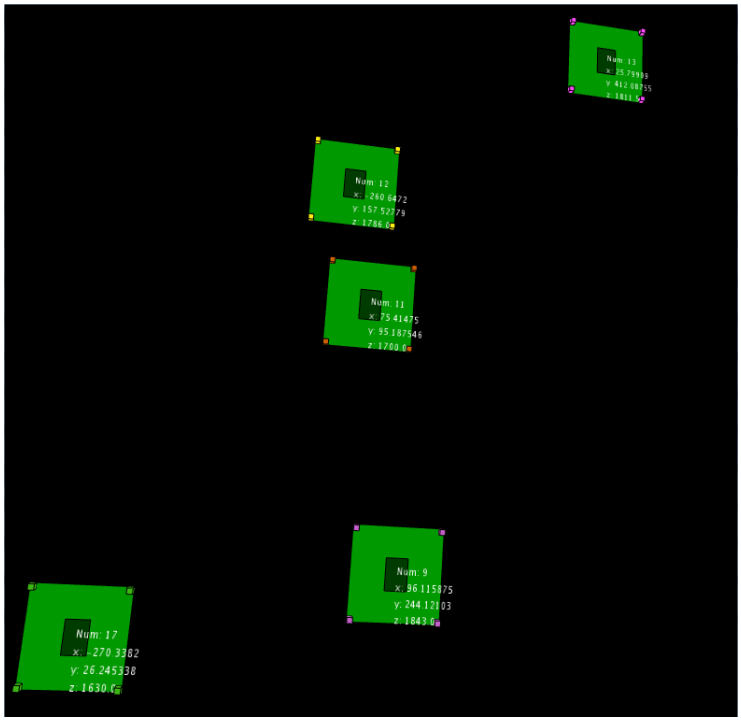


Figure 42. 3D Antenna element viewer

Each antenna node in Figure 42 is drawn in Euclidean space with the measured distances between pixels, the antenna number, and the antenna node's position in space. The user may interact with the model and measure any elements with perceived errors again.

Finally, the user can configure the communication settings for communication with the embedded phased array controller. The phased array controller is accessible over Bluetooth or a serial connection. The Bluetooth connection can also be mounted to the computer as a serial connection so that only serial connection parameters are needed in order to connect with the controller. After the user enters the proper serial settings, the user can interact with the phased array controller by downloading the positions, reading the current status of positions from the controller, download a desired steering angle, or read all current phases out of the controller.

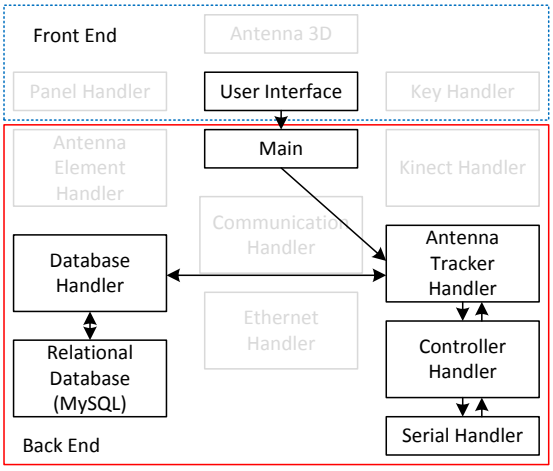


Figure 43. Components required interacting with phased array controller

Altogether, the variable uses of these core components allows the base station to successfully scan and track antenna nodes as originally designed in Chapter V. The components are highly scalable to adjust for different types of images, cameras, and other inputs from various devices. Smartphones and other internet connected devices can communicate with the main program through the Ethernet and communication handlers in order to retrieve data or control the system. This also leverages the previous benefits as demonstrated by the others of linking location information from a Smartphone.

Enhanced Filtration

The algorithm presented in chapter V to filter an RGB image required additional enhancements in order to properly determine the antennas location. As seen in the previous section, bounding boxes were placed around the filtrated light "blobs". These bounding boxes provided a simple mechanism to measure the center of the pixels and the antennas without having to perform an extended check on the contours found using the edge detection algorithms.

The bounding boxes were determined from the contour vectors using the pixel image data. The upper left hand corner and the bottom right hand corner of the bounding box were determined by (29).

$$\begin{aligned} C(x,y)_{upperleft} &= (\min(X_c), \min(Y_c)) \\ C(x,y)_{lowerright} &= (\max(X_c), \max(Y_c)) \end{aligned} \tag{29}$$

The method for finding the corners of the boxes iterates through the contour vector pixel positions and determines the minimum and maximum respectively for the upper left and lower right corners. This process yields an image as shown in Figure 44.

Figure 44. Bounding box filtered image

Thereafter, the enhanced algorithm essentially looks for two large bounding boxes that each contains two small bounding boxes within them. This check now requires that the smaller boxes for the pixels, as shown in Figure 44, only pass one statement instead of iterating through an entire contour vector to determine a shape. The footprint can also be adapted for variable levels of noise so that the algorithm looks for one big box with four smaller boxes contained within it. There are several unique examples of determining this 'box-within-box' technique that can be run in real time at a simple complexity.

Part of the reduced complexity comes from transforming the RGB image into the HSV color space. After much experimentation, it was found that the light produced at the very center of the LED always appeared as a white color on the RGB scheme and a red color in the HSV screen; however, the ambient light of the LED color would vary

away from this red color, allowing the light to be unique identifiable with a unique color code.

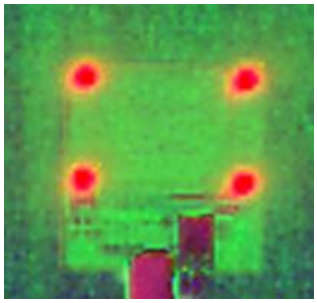


Figure 45. HSV image highlighting color of pixels

The HSV image in Figure 45 shows the bright red color of the white RGB pixel and the yellow color surrounding the red circle (which is actually green in the RGB color scheme). One could suggest that merely the red dots could be searched for in an algorithm; however, the identification of local pixels in comparison with other pixels from other antenna nodes could make algorithmic processing more challenging.

Database Schema

The information about antenna nodes is critical to the operation of the framework. Elements are first discovered on an Ethernet network and then are somehow referenced during the scan cycle of the framework using specific color filtration schemes and techniques. The databases that cache this information supply the framework with a state model that leverages scan cycles of the framework with respect to the user interactions through the software interface.

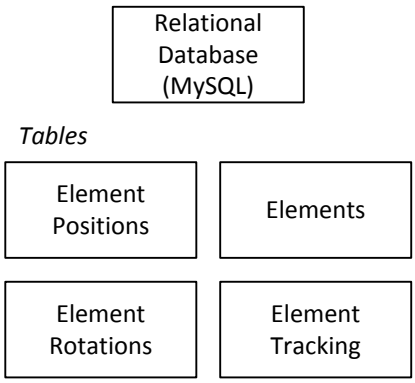


Figure 46. Tables in database for framework

Figure 46 shows the primary tables in the database. The *Element Positions* table contains information on the physical location of the element. This table is populated after a scan and links the element to the table *Elements* by a unique ID. The table *Elements* contains all of the network information and node information about the elements such as IP Address, MAC address, antenna number, and other factors. The table *Element Rotations* contains specific yaw, pitch, and roll information about a node and is linked to the *Elements* table through a unique ID. The *Element Tracking* table contains the cache information about the specific filtration codes for a specific element for the antenna tracker handler in the user program. This table is also linked to the elements table through a unique ID. The unique ID mentioned above is generated by taking the hexadecimal code of the MAC address and turning this into a string. The string then calls Java.lang.String.hashcode() which generates a unique ID based on the string parameters. Because every node on the network must have a unique MAC address, the hashcode function can also guarantee a unique numerical ID that is related to the specific

MAC address compiled as a string. A full reference document for the MySQL database schemas can be found in APPENDIX A.

Preliminary Distance Test

After writing the software for the base station framework and the nodes, an experiment was created to test the accuracy of the framework in a controlled environment. The goal of the experiment was to determine the accuracy of the system with the Kinect as a visual-spatial acquisition tool for random array geometries. Accuracy in these experiments is defined as the comparison of the Kinect measured positions in their relation to a resultant phase error according to (15), (16) and (17) using the measurement error. The Kinect was placed at varying distances from the closest element in the array to gain insight about optimal viewing distances of the camera to most accurately measure element position.

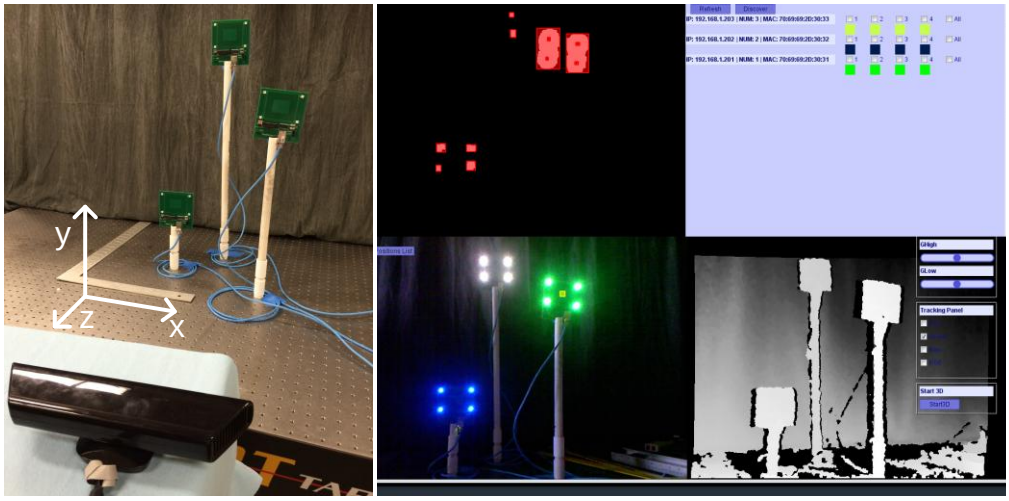


Figure 47.(Left) Three antenna experimental setup (Right) screen shot of software controller and Kinect tracking green antenna modules

The antennas were placed in a dark room to eliminate ambient light noise and were attached to three height-varying mounts on a measurement table. The Kinect camera connected to a computer which ran the system control software that performed all image processing and communication with the pre-existing phasing controller. The graphs in Figure 48 show (*x*,*y*,*z*) coordinates for the closest antenna to the Kinect for four varied distances. The experiments were run in successive order to calibrate all measurements according to the displaced distance of the Kinect camera. Doing this would minimize the amount of ambient noise error induced by the camera and provides an accurate estimate of the distance variation of the different measurements. The goal of this experiment was to align with the precision measurements that have been mentioned in the literature and to verify this particular accuracy of measurement with this particular end node. This prepares the system for accurate distance readings for tracking various elements.

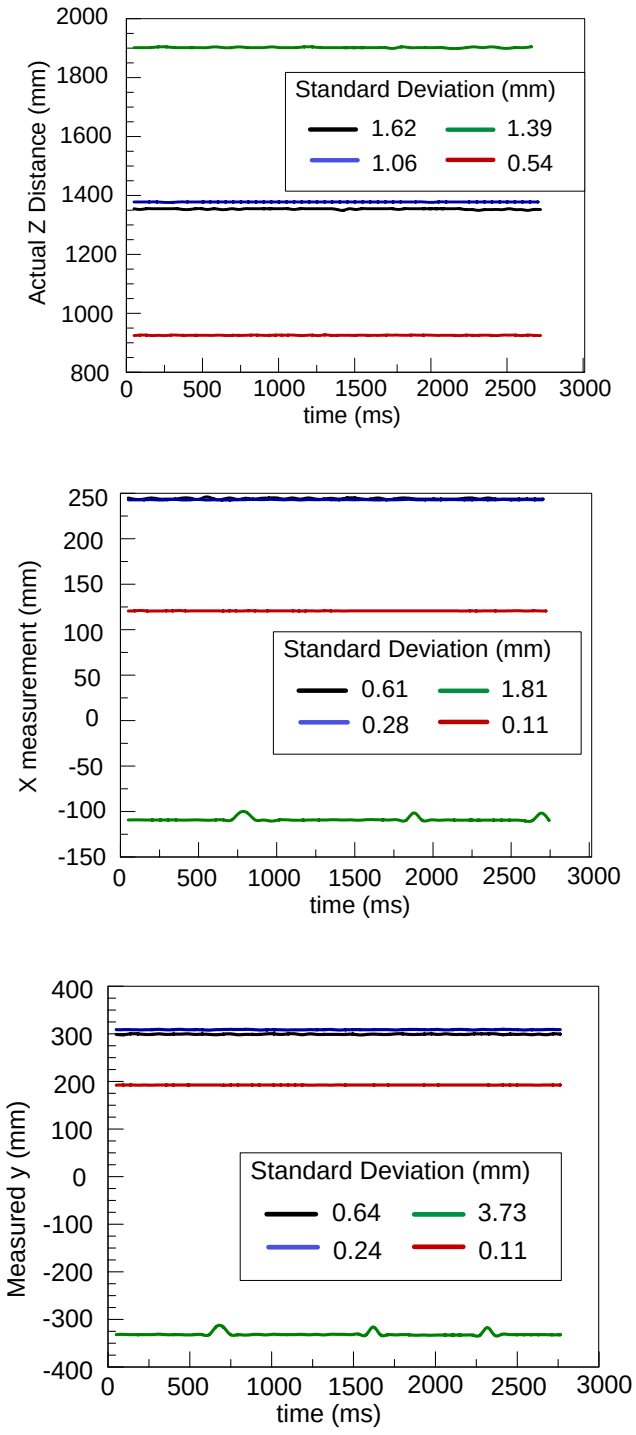


Figure 48. Standard deviation measurements for real world (middle) x, (bottom) y, and (top) z coordinates

Standard deviation increased as the Kinect moved away from the array to accommodate larger aperture size but had sub-millimeter drift in close proximity ($z < 1m$). Similar accuracies and reports of drift were also reported in some of the studies covered in the literature review in Chapter III. These tests indicated that the image processing algorithm could successfully detect and track an element location with minimal change during time. A tradeoff is that as an aperture size increases the Kinect will be forced to move farther away increasing the potential for measurement error. Positions from the Kinect were then compared with hand measurements. Kinect spatial detection error in millimeters averaged between 4 – 8 mm with a worst case estimation error of 14 mm in the y axis at distance of 925 mm from the closest antenna. The measurement error and error range present the possibility of using this spatial recognition technique as a 4-bit or 3-bit phase shifter.

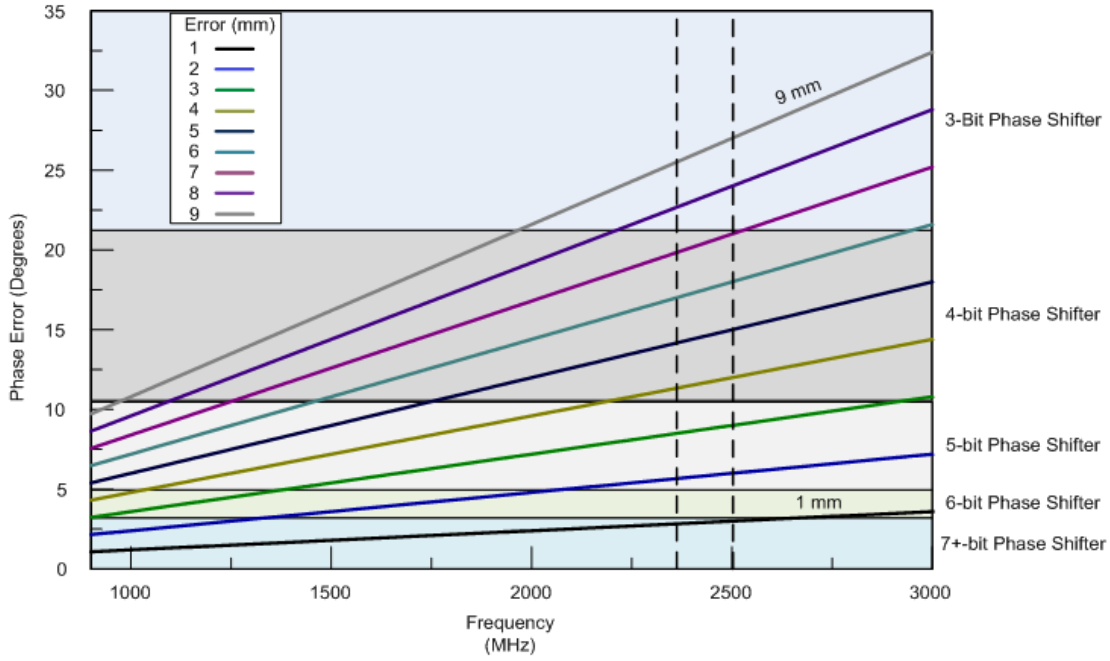


Figure 49. Relation between frequency, phase error, measurement error, and n-bit phase shifters

This result is important because it provides a unique metric for evaluating the system at the design frequency in terms of potential phase error. The measurement error produces a phasing error that additively contributes to the total phase delay the antenna element experiences. It is therefore important to reference an n-bit phase shifter for the respective frequency and measurement error to determine beam-steering capabilities of the system. This metric additionally reemphasizes the importance of minimizing measurement error with higher resolution cameras and better spatial detection techniques such as those used with LIDAR.

The estimation error of the Kinect and array system allows the phase shifter controller to operate at 4-bit phase shifter accuracy. This calculation was derived from equations (15) – (17) where the resultant phase error at a frequency of 2.4 GHz for 1mm error in measurement is approximately 2.88 degrees. To observe how measurement error would impact arrays that will be tested, simulations were conducted with a randomized 4-bit phase shift error of (+/-) 22.5 degrees.

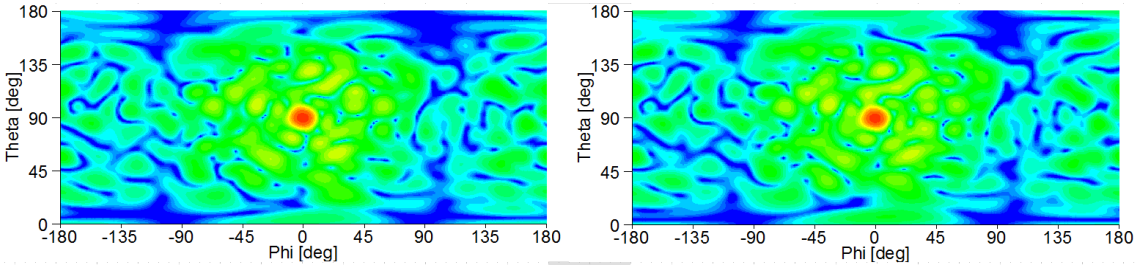


Figure 50. Normalized gain radiation pattern for (Left) exact phase shift 32 element array (Right) randomized 4-bit phase error 32 element array

Figure 50 shows a holistic and comprehensive view of the exact and randomized 4-bit error phase shifting radiation patterns displaying less than 1 degree of angular shift of the main beam. The side lobe behavior of the 4-bit randomized array shows unsymmetrical shifts and increased gains of the closest side lobe that is additionally seen in Figure 50 due to the randomized 4-bit error.

With these simulation results, it can be concluded that the framework should supply sufficient accuracy to support radiation behavior from the aperiodic array.

Accordingly, the main pattern should be observable and in an observable cut plane in the anechoic chamber to compare measurements against these simulations.

System Tests

A test apparatus for the framework was built given the outcomes of the depth accuracy test and the simulations of the 32 element array. A 16 element reconfigurable aperiodic array was constructed using the fabricated elements. Because of the many arms of the apparatus and the termination of each arm with an antenna node, the aperture was named Medusa.

Figure 51. Testing apparatus "Medusa" in anechoic chamber

The testing apparatus required two arms to support the each antenna node due to the spinning effects of the rotational table in the anechoic chamber. The arms, seen as the black twisted plastic, carry both the Ethernet cables and the RF feed cables directly to the antenna module. The phased array controller is shown sitting beneath the apparatus as the collection of green PCB boards. Limited distance in the anechoic chamber required that the Kinect be placed at two fixed distances away from the array.

Figure 52. Kinect setup with testing apparatus

The Kinect was covered with attenuator commonly found in anechoic chambers to minimize the scattering from the electronics. Additionally, it was placed on a small stand made of attenuating material to minimize any further scattering effects. Furthermore, the

entire apparatus was adjusted to be in the center receiving plane of the receiving antenna in the anechoic chamber. Only one scan plane could be measured for all of the following tests that were conducted with the apparatus.

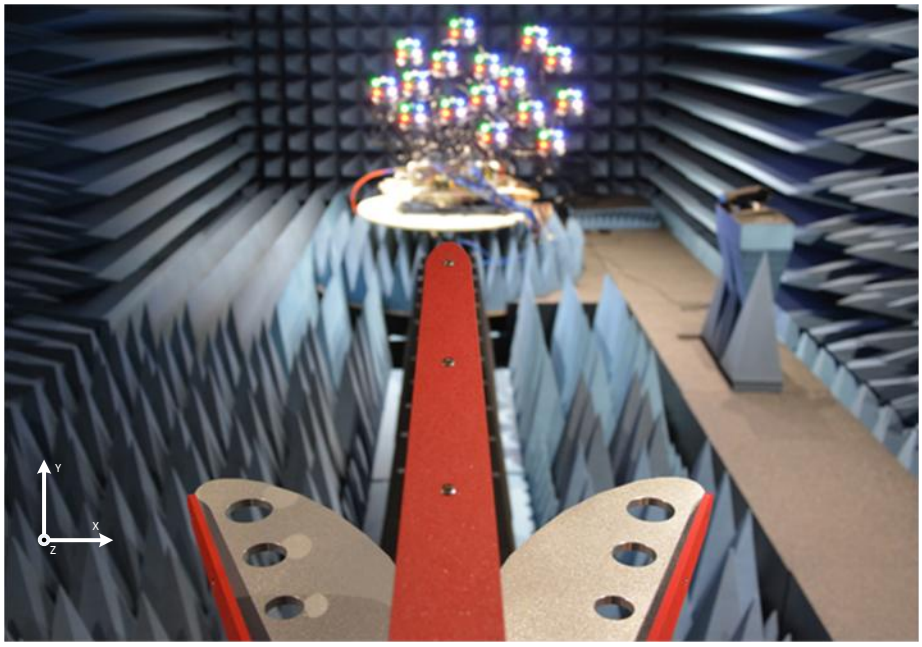


Figure 53. Receiving antenna aligned with the test apparatus

The distance from the receiving antenna to the apparatus is approximately 26 ft. Depending on the various reconfiguration sizes of the aperiodic aperture the receiving antenna is placed nearly in the radiating far field for measurement. As aforementioned, the vertical polarization as shown in Figure 53 was the co-polarized receiving plane and

the horizontal cut plane was the cross polarization receiving plane. This configuration is valid for all measurements taken for all of the following system tests.

Simulations were performed and compared to measurement data in all following subsequent tests of the apparatus. The goal of these experiments was to determine whether the apparatus and framework could provide insight and capability into studying radiation behavior of swarming aperiodic arrays. Because each measurement can be taken at discrete points in time where the antenna nodes are not actually moving, the data received in the measurements will be normalized to the starting position. This testing methodology will allow the framework to realize the effects of moving elements at a single observation point in space. In the case that gain is lost or the main radiation beam appears to reside in a different cut plane, an adjustment for beam steering has not been executed. Instead, it is important to observe how the radiation properties occur at a single observation point given a swarm of elements.

Single Apparatus Tests

The first tests on the apparatus were to observe simple radiation behavior at broadside radiation of the aperiodic array. Testing required the framework to scan all elements and configure the phased array controller with the respective positions for the elements. Three different configurations with their respective simulated and measured results are shown in this section to provide analysis of the framework before moving to more advanced configurations.

Table 6. Aperture dimensions of Test 1

X Distance (λ)	Y Distance (λ)	Z Distance (λ)	Radius (λ)
8.58	3.35	3.53	4.66

The bounding measurements of the first configuration for the aperiodic array are shown in Table 6. Note that the various x, y, and z distances provide a rectangular bounding geometry with respect to the coordinate system as shown in Figure 53. The radius is provided as a baseline measurement for the minimum bounding sphere to completely encompass the entire volume of the array.

Figure 54. Physical configuration of Test 1

The antenna nodes were configured into the formation as shown in Figure 54 using a simulation program written to view distances, movements and patterns in space for each of the tests. A broadside measurement was taken first to view whether a radiation pattern

from the aperiodic configuration could be observed. The measured pattern was plotted against the simulation pattern to view the differences in Figure 55.

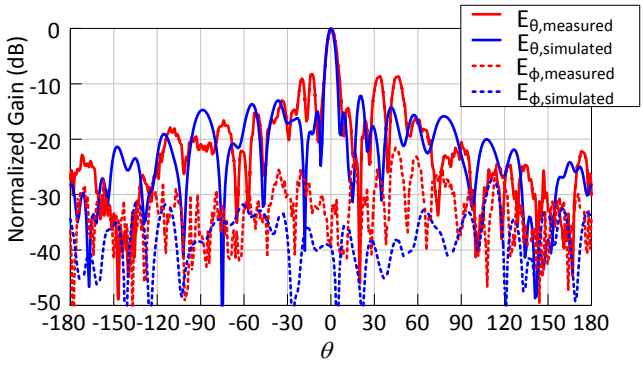


Figure 55. Normalized broadside radiation pattern simulated and measured Test 1

The results in Figure 55 demonstrate that a radiation pattern from the apparatus can indeed be observed, although the resultant radiation pattern didn't identically match the simulated radiation pattern. The pattern was normalized in order to compare the main beam with the side lobe levels and because the anechoic chamber at the time of testing was uncalibrated for proper gain measurements. It is noticeable that the side lobe levels are on average around 3 dB higher than the side lobe levels from the simulation pattern. The goal of this research is to not necessarily compare these metrics together inasmuch as it is to observe whether or not the framework can produce a resultant radiation pattern that can be studied. Additionally, it is uncertain exactly what plane of the measured

volume is being studied as the receiving antenna may not be at the aperture center of the array for every scan.

The phased array controller was then programmed to scan the beam from -45 degrees to +45 degrees in the scan plane.

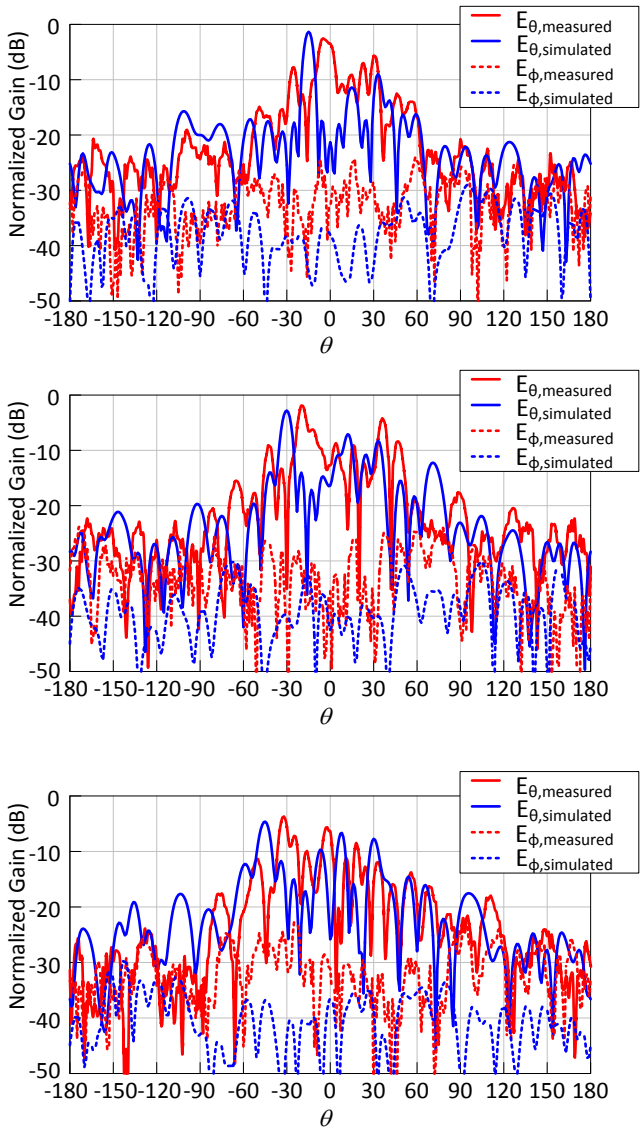


Figure 56. Scanning from (top) -15, (middle) -30, and (bottom) -45 degrees

From Figure 56, the steering angles are off on average by 12 degrees but demonstrate the ability to scan a main beam to these steering angles. The side lobe levels showed some correspondence to the simulated side lobe levels, but exact conclusions about the behavior cannot be drawn.

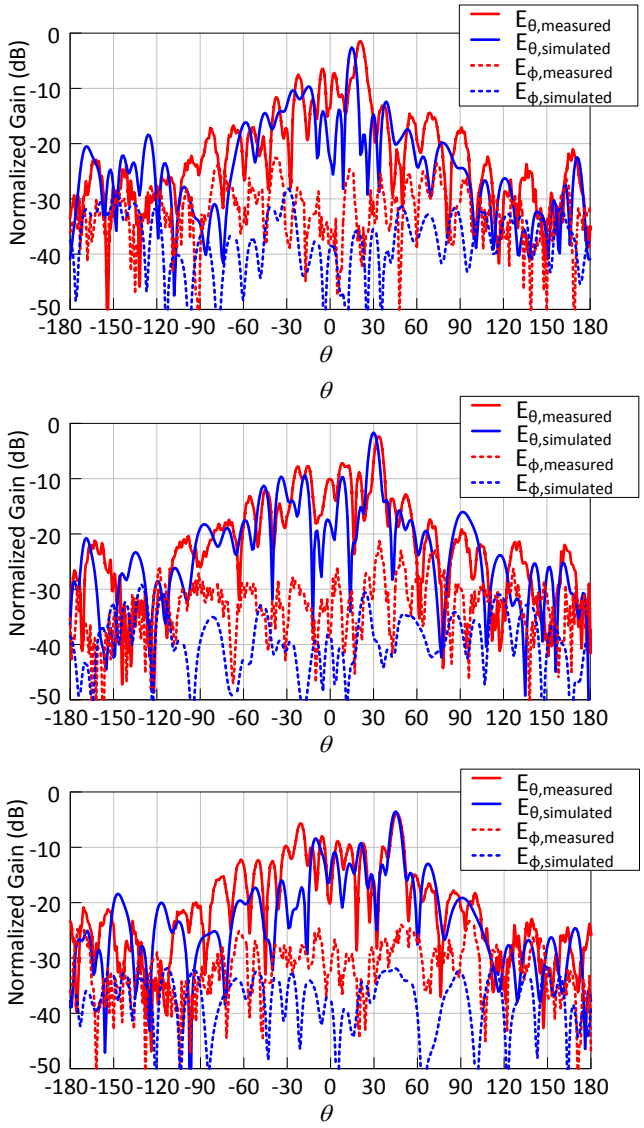


Figure 57. Scanning to (top) 15, (middle) 30, and (bottom) 45 degrees

The graphs shown in Figure 57 demonstrate a closer match to the simulations when scanning the opposite direction with all angular measurements within 4 degrees of the simulated versions. It is first unclear why this particular agreement occurs with this variation as nothing has been configured with regard to geometry or setup that would ultimately indicate this type of behavior. The 45 degree scan was nearly aligned with the simulation differing less than 1 degree. This was atypical for this particular geometry as the larger scan angles should have less accuracy due to the element pattern of the array elements. The element pattern of the patch antenna that was previously shown demonstrates the ability of the patch to have high gain at a broadside radiation pattern with decreasing gain as the angle of the radiation pattern tends towards 45 degrees. Ultimately, this behavior can also be seen in the array pattern due to the element pattern. The simulated heatmaps in Figure 58 provide further insights about the side lobe behavior during various scans of this particular aperiodic array.

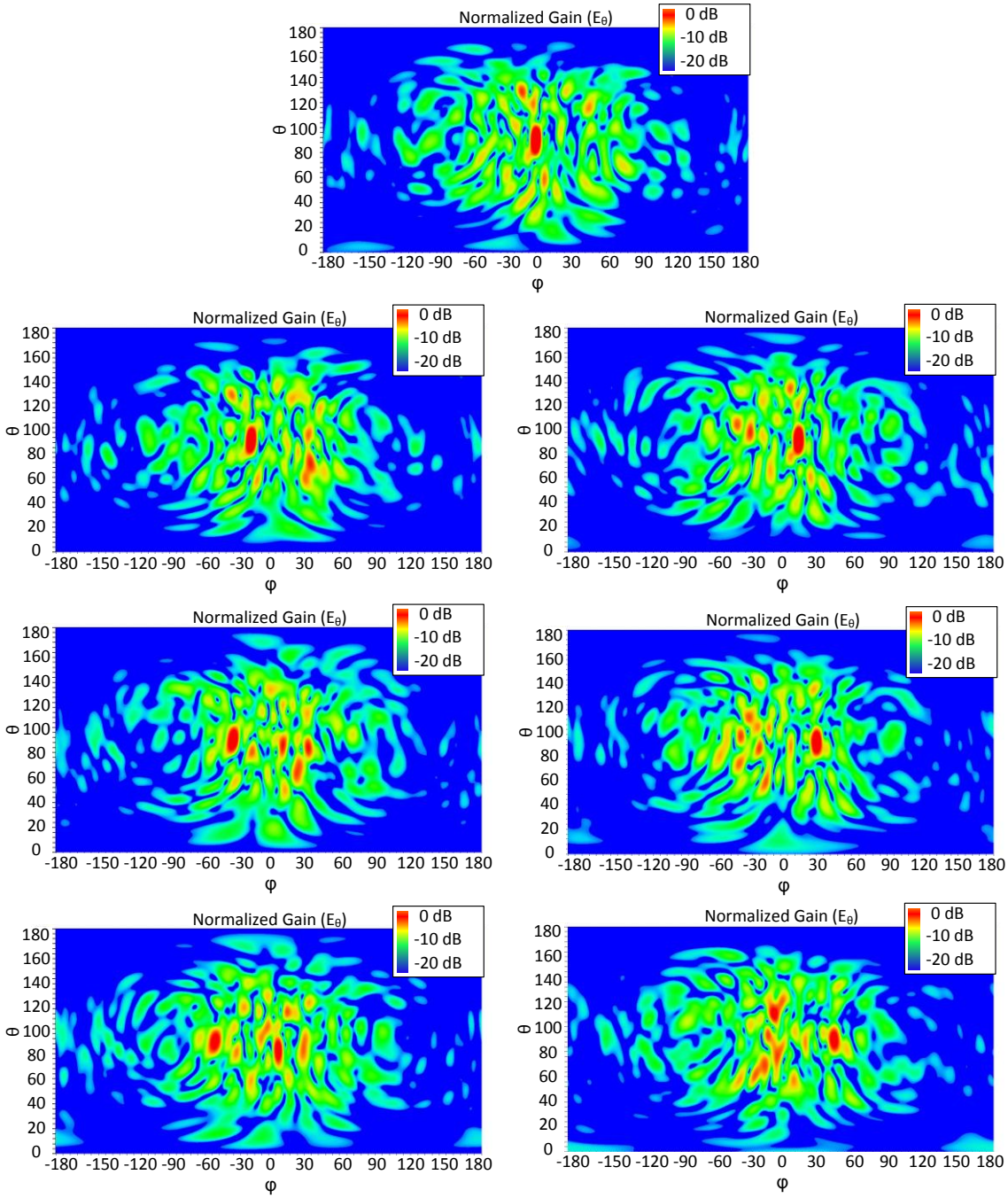


Figure 58. Simulated heatmaps showing side lobe change behavior for different scanning angles

The simulations have rotated the axis of measurement by 90 degrees in order to observe an overlaid heatmap for the array. Essentially, these heatmaps can be wrapped around the bounding sphere of the array above and represent the simulated radiation pattern of the array. Therefore, all heatmap plots will be plotted as phi versus theta for simulated results. For each scan there is a new reconfiguration of the side lobe behavior with the peaking behavior of the side lobes at -45 and + 45 degree scanning angles. What is also interesting about the heatmaps is they provide a glance into the side lobe behavior of the scanning plane of the array. For some scanning angles, little side lobe activity occurs in the measured scan plane at $\theta = 0$ degrees; however, some scanning angles draw the primary side lobe levels directly into the scan plane.

With successful results on the first test an additional geometry was created and the same tests and simulations were run.

Table 7. Aperture dimensions of Test 2

X Distance (λ)	**Y Distance (λ)**	**Z Distance (λ)**	**Radius (λ)**
7.42	3.86	3.31	4.42

The framework scanned all the elements according to the procedure listed in the previous chapter and in the same way that the previous aperture was scanned.

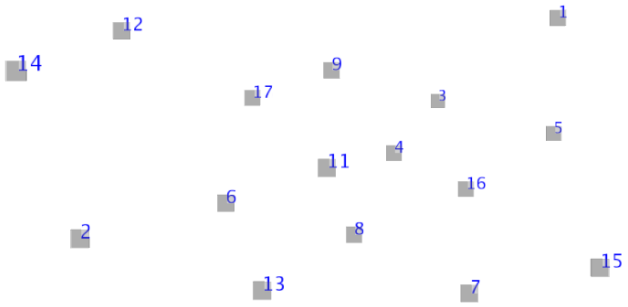


Figure 59. Physical geometry of Test 2

The simulation software read in the files to show the displacement of each antenna node in the new array configuration. A broadside pattern was programmed into the phased array controller to ensure that the framework had provided accurate measurements to calculate phasing.

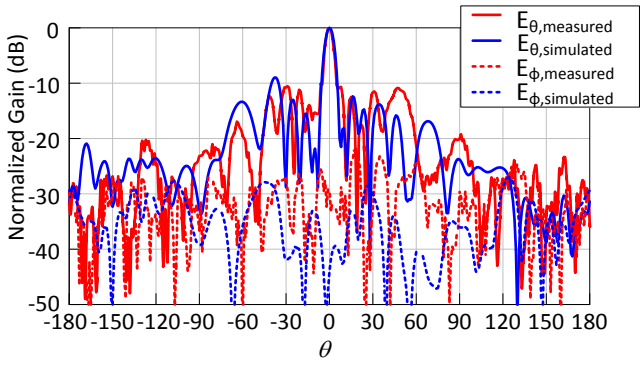


Figure 60. Normalized broadside pattern for simulated and measured results of Test 2

The broadside pattern shows alignment between the simulated and measured results. The average side lobe level showed a difference of 0.7 dB and the main beams are aligned. Again, scanning patterns were taken from this array configuration to determine the accuracy of the framework and the node estimation results that were produced.

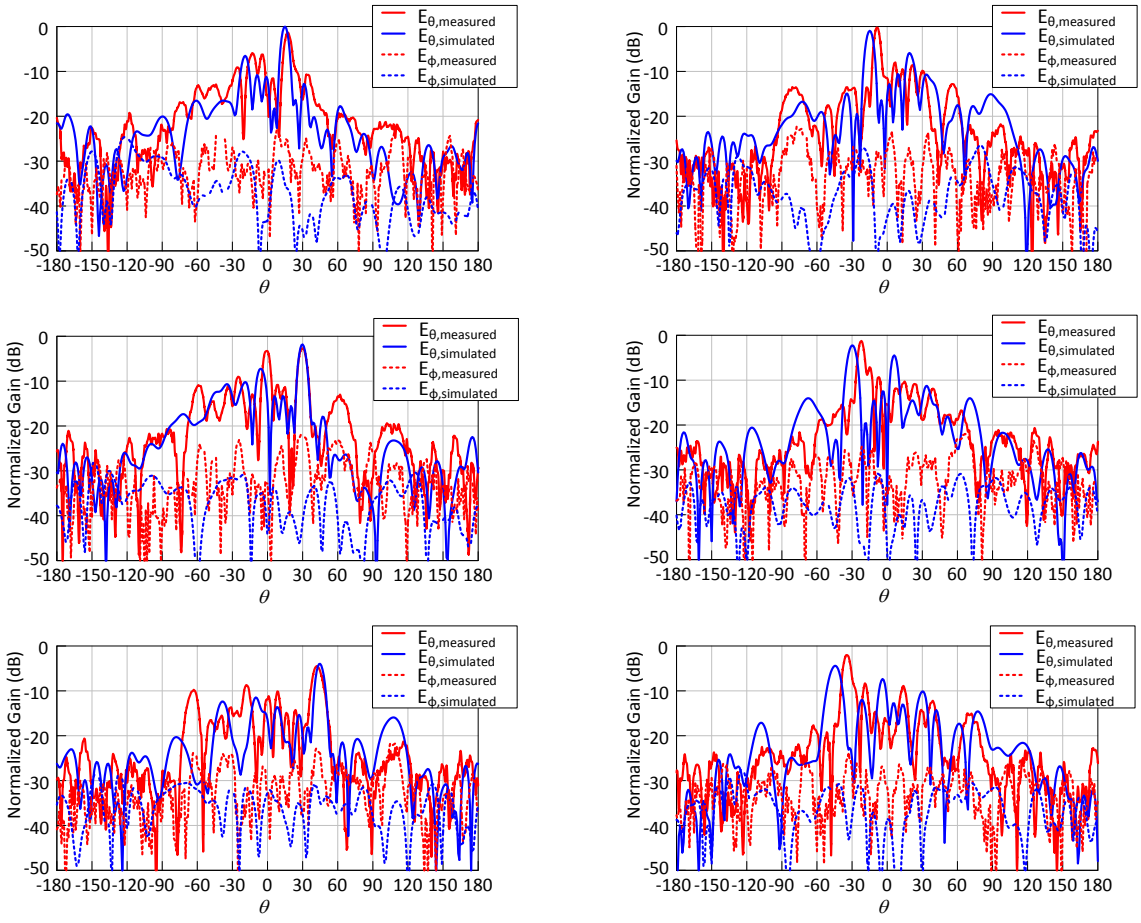


Figure 61. Normalized radiation scanning patterns for simulated and measured results from -45 to +45 degrees

The scanning patterns of Test 2 shown in the Figure above show a similar behavior to the results achieved in Test 1. In the positive scanning direction it is observed that the simulated and measured patterns have nearly aligned main beams with respect to angle and gain; however, scanning in the negative direction produces similar error results in that the difference of the main beams on average is 12 degrees. It is not clear whether this steering error arises from a configuration of the geometry or if it is a resultant error from the phased array controller. It is noted that the phasing of the phase center of the array could be off for the internal measurements taken within the chamber. An additional thought is that the array is being slightly under phased in both tests in this case leading to a deficiency in array scan to a particular direction. Because the phases are not measured with an analyzer in a feedback loop, it is not known whether this is the specific case for this particular behavioral occurrence in the measurements. This behavior is equated to a replicated system behavior rather than a deficiency in the operation of the array itself.

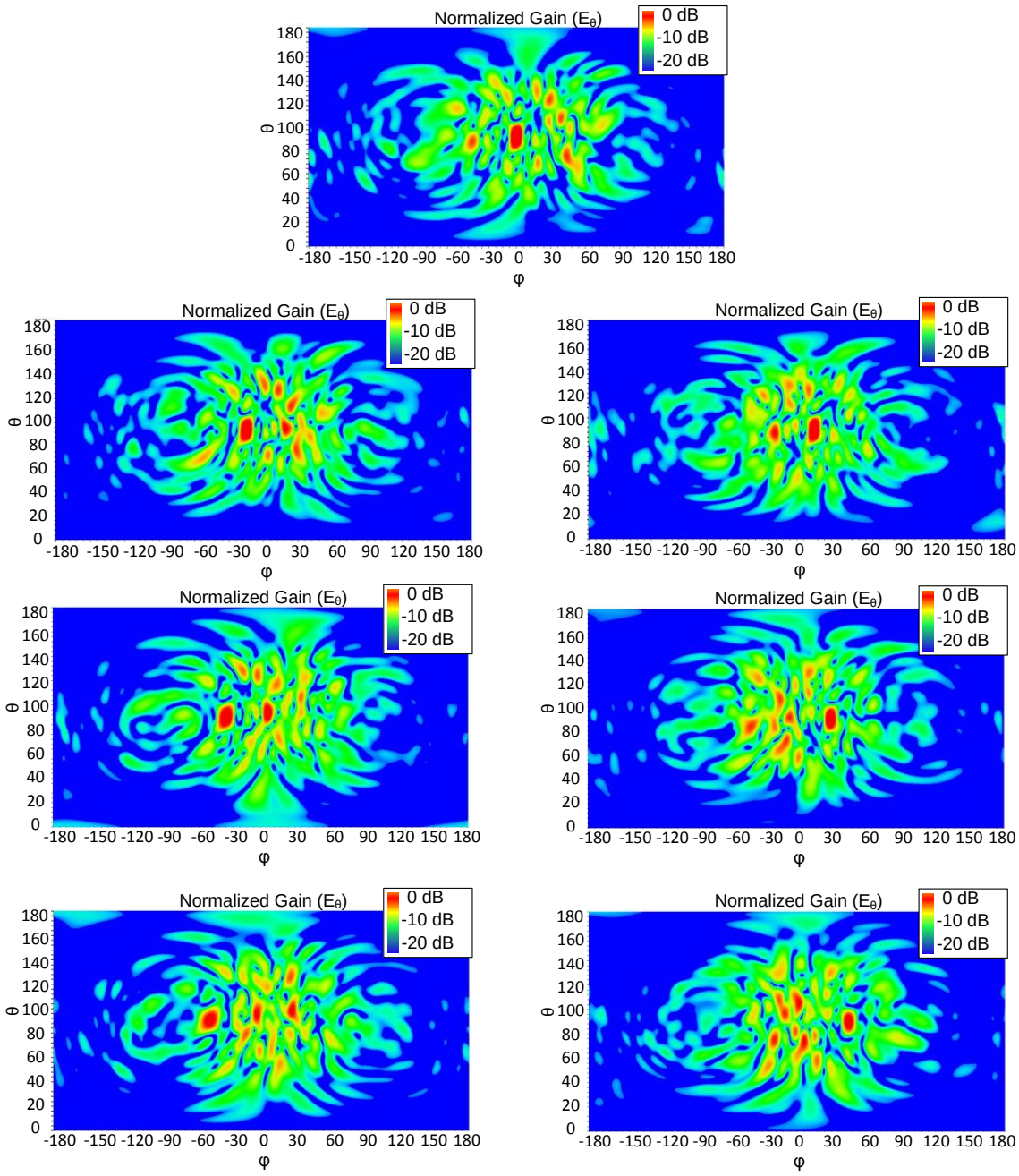


Figure 62. Scanning simulated heatmaps for Test 2 from -45 to 45 degrees

The heatmaps in this case show that the peaking side lobe occurs more in the measurement plane than Test 1. Note that the side lobe levels in this case continue to be asymmetric due to the random stochastic distribution of the nodes within the array. Overall, the results of Test 2 reaffirm the findings in Test 1 and reinforce the proposed design of the framework.

A third test of the same kind was conducted with a different geometry to observe any differences due to a changing structure. From the previous two tests, it was observed that scanning angles could be achieved, as well as, a broadside pattern with accurate measured results when compared to the simulation.

Table 8. Aperture dimensions of Test 3

X Distance (λ)	**Y Distance (λ)**	**Z Distance (λ)**	**Radius (λ)**
7.00	4.17	2.78	3.94

In comparison with the first two tests, Test 3 reduces the *x* and *z* distances making the array more compact while increasing the y distance. Increasing the y distance in this test case brings the elements further away from the center scan plane of the array and shortening the z distance forces the difference in phasing for a broadside pattern to be greater in similarity. The overall bounding sphere has also changed by more than a wavelength.

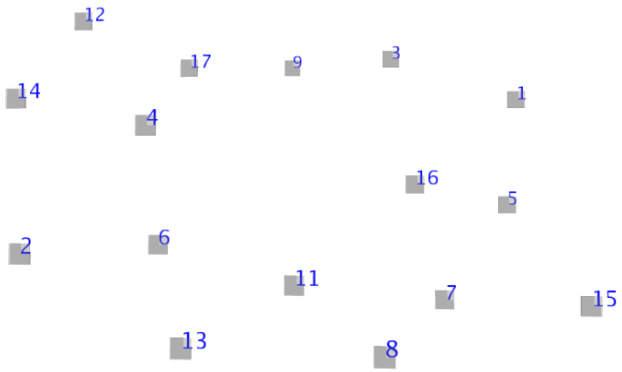


Figure 63. Physical geometry of Test 3

The increased *y* axis distance between the elements is observed in Figure 63 and there additionally is a small gap in the center of the distribution. In consistency with the previous tests, a broadside pattern was taken to observe the radiation pattern behavior of the distribution.

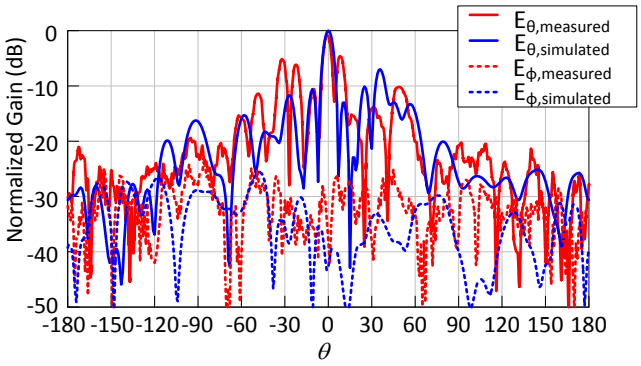


Figure 64. Normalized broadside pattern of simulated and measured results for Test 3

The broadside radiation pattern shown in Figure 64 is distinctly different from the previous broadside radiation patterns in the other tests. The pattern has a primary peaking side lobe 5 degrees off from the main beam in the measured result. This behavior was not observed in the simulated result. Scanning radiation patterns were still taken despite the measured broadside radiation pattern result.

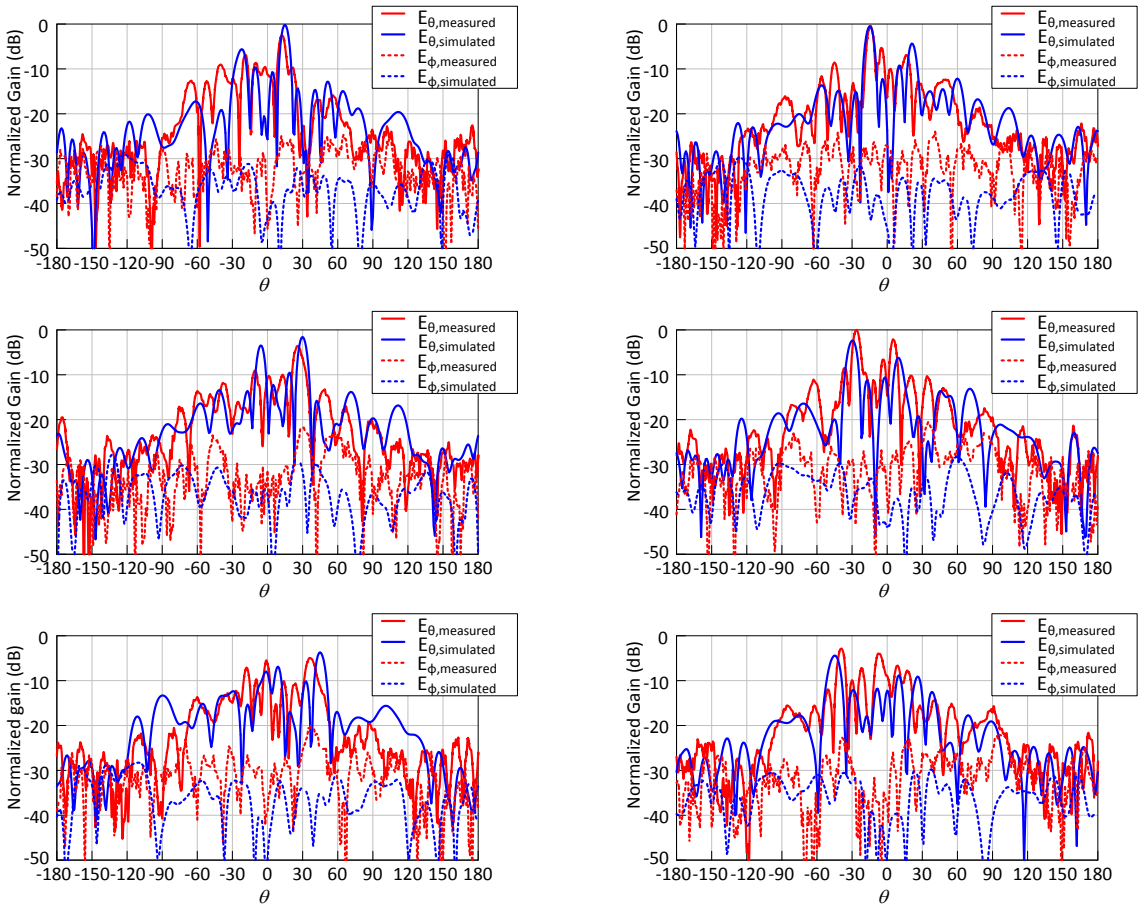


Figure 65. Normalized scanning patterns for simulated and measured results from -45 to +45 degrees for Test 3

The graphs in Figure 65 indicate that beam steering and scanning was still possible despite a broadside radiation pattern that did not align with the simulated broadside radiation pattern. Moreover, the scanning patterns in the positive theta direction indicate a consistent -3 dB gain with respect to the simulated value; however, the negative theta scans show that the measured gain for the -15 and -30 degree scans stayed nearly equal with the broadside pattern. Additionally, the scanning patterns were more consistent in this test than in the previous tests as the average angle error measurement from simulation to measured results was 8.5 degrees. The 15 degree scans only differed by 2 degrees in the positive and negative scan cases. This case of the closer scan angles was expected as compared to the previous tests due to the element pattern of the patch antennas being used in the array. It is expected that the overall gain and the pattern accuracy should be reflected in the accuracy of the phase measurement and will match the binding pattern of the element pattern.

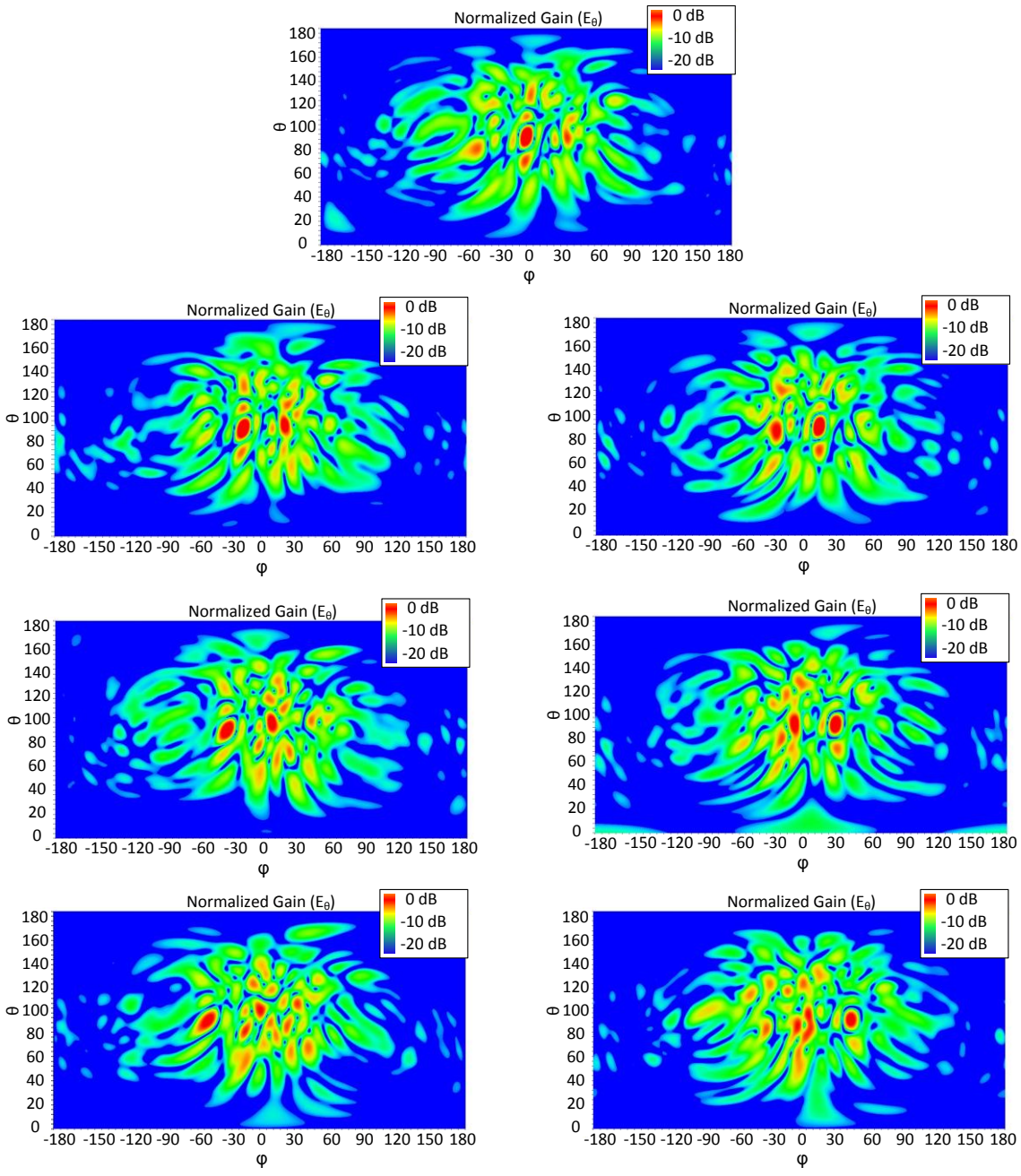


Figure 66. Scanning simulated heatmaps for Test 3 from -45 to 45 degrees

The heatmaps shown in Figure 66 indicate the presence of at least one peaking side lobe in the $\theta = 0$ scan plane. The random distribution of the peaking side lobes is also observed due to the random configuration of the nodes in the structure.

The conclusion of these three tests provided the results needed to move forward with more advanced testing focusing on swarming arrays and testing some additional properties of the framework and simulations. These tests provided a ground metric that demonstrated the framework's ability to measure nodes in space and subsequently relay this information to the phased array control to produce broadside and scanning radiation patterns.

Rotational Impacts on Simulations

After conducting the first three experiments, it was clear that the quasi-static plastic arms that held the nodes slightly moved during testing due to the angular rotation of the test bed in the anechoic chamber. Additionally, mounts were 3D printed to hold antenna nodes in place during tests to minimize the amount of rotation for a specific element. It was observed during these tests that antennas were slightly rotated in all directions and due to human arrangement of the nodes, it was nearly impossible to align all antennas to a single plane.

Data from the Kinect provides all four pixel locations on an antenna node. Using these pixel locations it is possible to determine the angular rotation of all the elements in space with respect to yaw, pitch, and roll. Therefore, the goal of this small experiment was to determine the yaw, pitch, and roll for each element and insert these values into a simulation to observe any effects that it has on the radiation behavior of the array. Yaw,

pitch, and roll values were calculated using normal vectors from planes created from the pixel locations. The largest angular rotation found during this calculation was 16 degrees for a yaw measurement.

A simulation was created that reflected the individual yaw, pitch, and roll rotation values for each element with respect to Test 1 in the previous section.

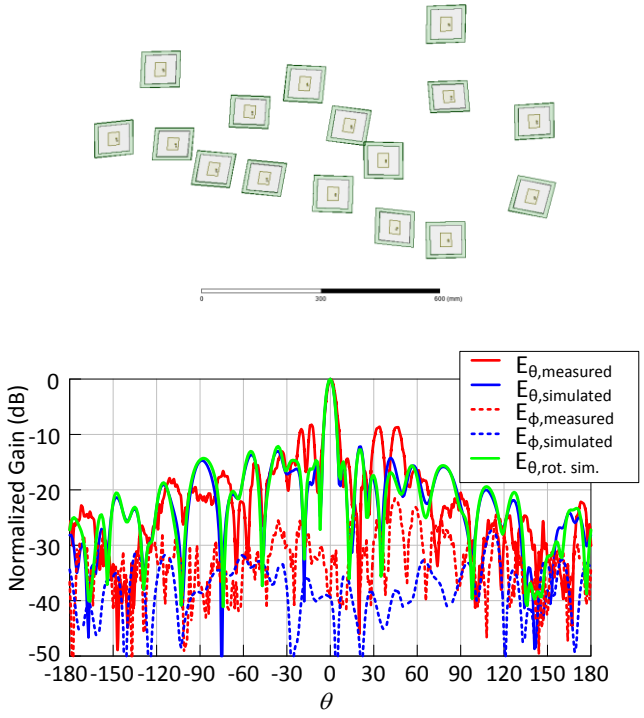


Figure 67. (Top) Test 1 elements rotated in simulation (Bottom) simulation results of normalized broadside pattern

The results from the simulation shown in Figure 67 show that the rotation had some impact on the side lobe levels of the radiation pattern but did not change the location or

characteristics of the main beam. The side lobe average was raised by about 0.3 dB in the rotated case compared with the ideal simulation. From this simulation experiment, it is indicated that slight rotations in any direction have minimal impact on the overall main beam and should not be considered as a strong impact on the overall radiation pattern. Inputting the rotations however did not force the simulation pattern to align any closer to the shape of the measured results.

Kinect Depth Test

A depth test was created to determine the Kinect's accuracy on the overall radiation pattern of the aperiodic array to reflect the results previously covered in the first section of this chapter. The Kinect was placed at two varying distances within the anechoic chamber. As aforementioned, due to space constraints in the anechoic chamber, only two significantly varying distances could be achieved.

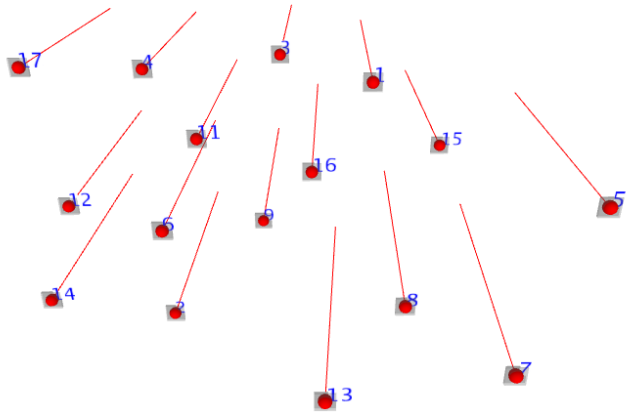


Figure 68. Physical geometry for the Kinect depth test

Using the developed simulation program to observe measurements from the Kinect, the individual paths of elements can be traced as shown in Figure 68. The red dots represent the second position of all elements and the red lines leading to the red dots represent a linear trajectory from the original point in space. The aperture dimensions of the aperiodic array used for this test are shown in Table 9.

Table 9. Aperture dimensions of Kinect Depth Test

X Distance (λ)	**Y Distance (λ)**	**Z Distance (λ)**	**Radius (λ)**
6.9	3.23	3.55	4.01

The measured distances of the elements between locations 1 and 2 varied slightly for x and *y* but had a noticeable difference of 18 mm in the *z* direction on average. This measurement difference can have a substantial impact on the broadside radiation pattern as the phasing for broadside radiation is only dependent on the *z* position in space.

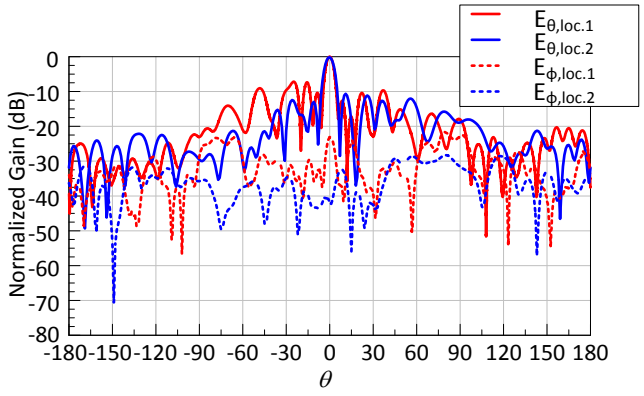


Figure 69. Measured normalized radiation patterns for location1 and location 2

The radiation pattern shows that when the Kinect was further away from the array a better radiation pattern was produced with respect to the average side lobe level. This seems inconsistent with the original measurements that indicated less accuracy of the Kinect further away from the object it is measuring. Upon inspection of several factors surrounding this result, it was concluded that the measurement could be more accurate due to the impact that the light from the LEDs of the antenna node on the camera. At close distances, the light emitted from the LEDs is very bright and causes the image processing algorithm to have varying estimations of accuracy on nodes that are very close to the camera. If all the elements are farther away from the camera, their light appears closer in strength to the camera and thereby produces a more accurate measurement.

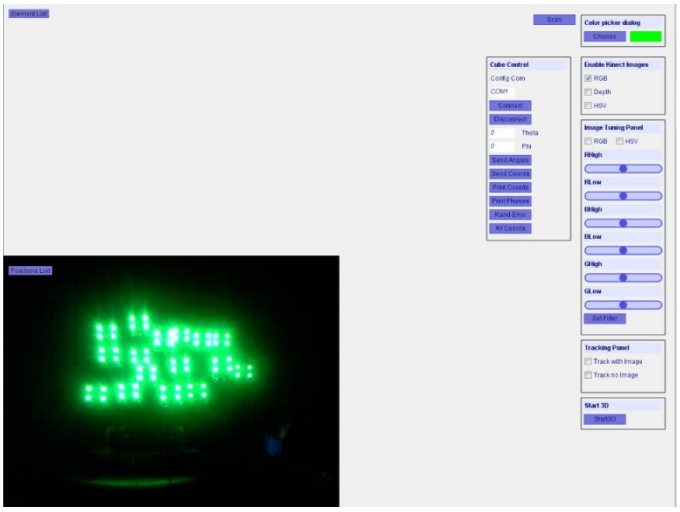


Figure 70. Screenshot of apparatus with the Kinect at location 2

Note that the apparatus fills nearly half of the image field of view of the Kinect as shown in Figure 70; whereas, the apparatus nearly fills the entire image at the first distance (no image available). From these results, the Kinect was left at location 2 for the remainder of the swarming experiments. The *z* distance at location two is approximately 1.7 meters away from the closest element in the array.

Slow Morphing Cluster

The morphing cluster experiments are one of the primary goals in creating the framework listed in this work. Measuring the movements of individual elements through time and observing the radiation patterns are important characteristics in defining electromagnetic properties about morphing swarms. In this first experiment, the elements move to two additional locations beyond their starting configuration. Radiation

patterns are compared with simulation patterns as in previous experiments to show the accuracy of the framework in measuring positions of nodes.

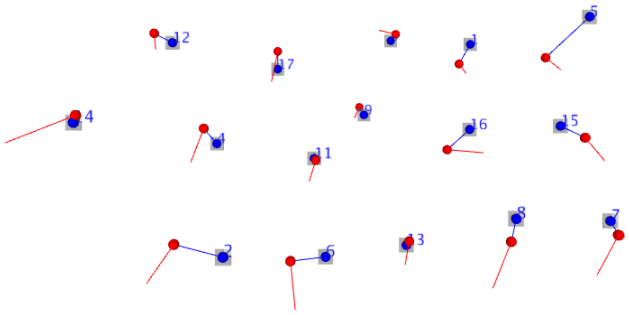


Figure 71. Physical geometry of all 3 locations for slow morphing cluster

In Figure 71, the red and blue dots represent locations 1 and 2 respectively after the starting location, which is at the tip of the first red line. It is clear that some elements have coordinated movements with one another, such as elements 2 and 6, while other elements move randomly in the confined geometry. Overall, their path trajectories do not overlap with one another that define this structure as slow morphing.

Table 10. Aperture dimensions of locations for Slow Morphing Test

Location	X Distance (λ)	Y Distance (λ)	Z Distance (λ)	Radius (λ)
1	8.15	4.18	3.26	4.75
2	8.18	3.9	3.05	4.44
3	8.44	3.98	2.75	4.69

The varying distances shown in the table above indicate that the structure is slowly becoming smaller by nearly a wavelength with respect to the radius of the bounding sphere. The primary change of element distribution happened in the *z* direction in order to observe any effects on the broadside radiation pattern of this structure.

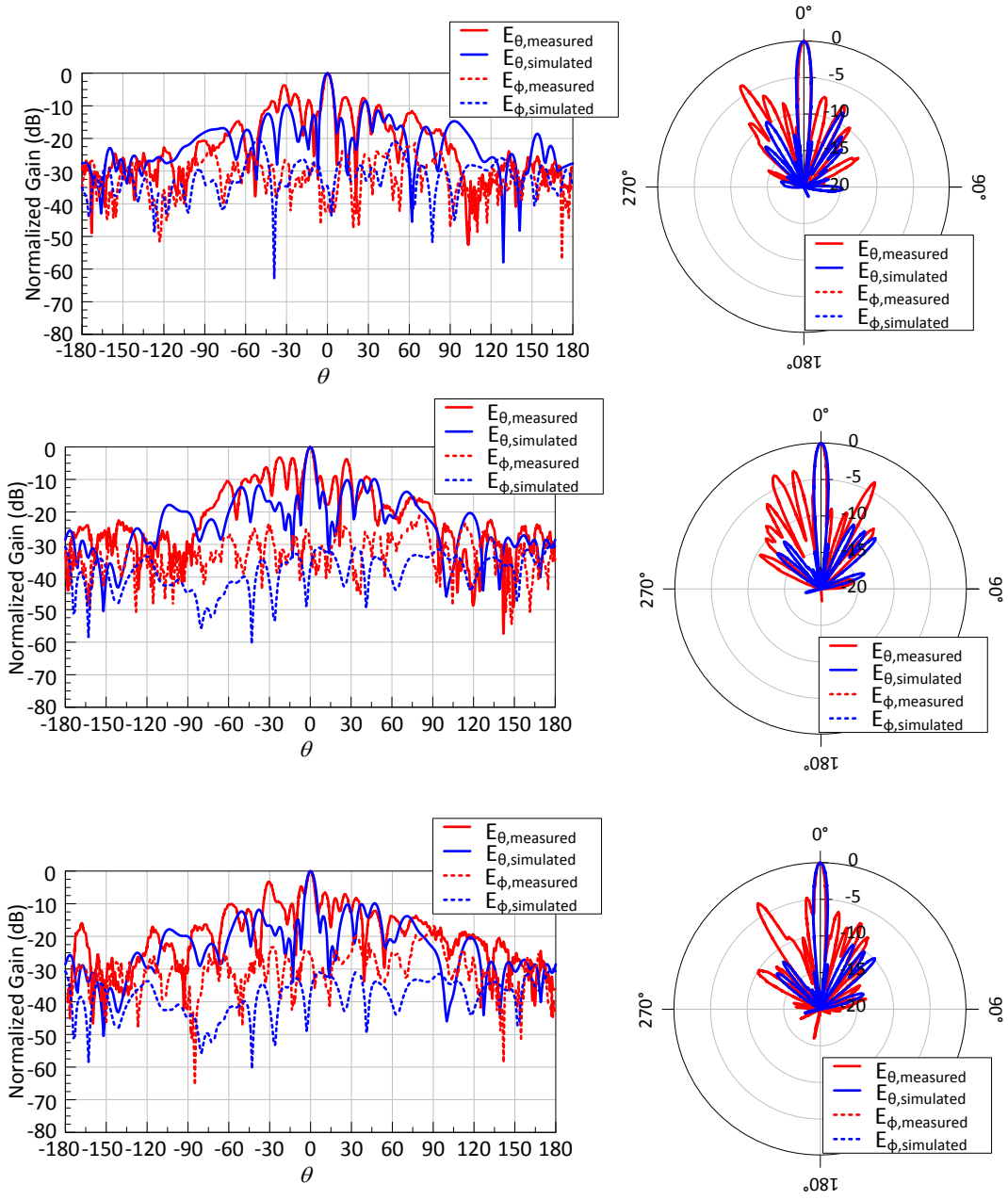


Figure 72. Measured and simulated normalized broadside radiation patterns for (Top) location 1, (middle) location 2, and (bottom) location 3

The radiation patterns shown in the figure above indicate that the swarm is able to produce a main beam as the elements shift from their starting positions to the following locations. There is an increase in the average side lobe level upon reaching position 3. It is not clear what the source of this increase is as the simulation demonstrates that the main beam should maintain lower side lobe levels than what the measurement is showing. All the normalizations in the plots are normalized to the maximum received power at one of the locations. This is to demonstrate how the main beam can lose gain at a particular observation point under different configurations. In this experiment however the main beam kept nearly identical respective gains through the different locations and the gain change is not observable from the plots.

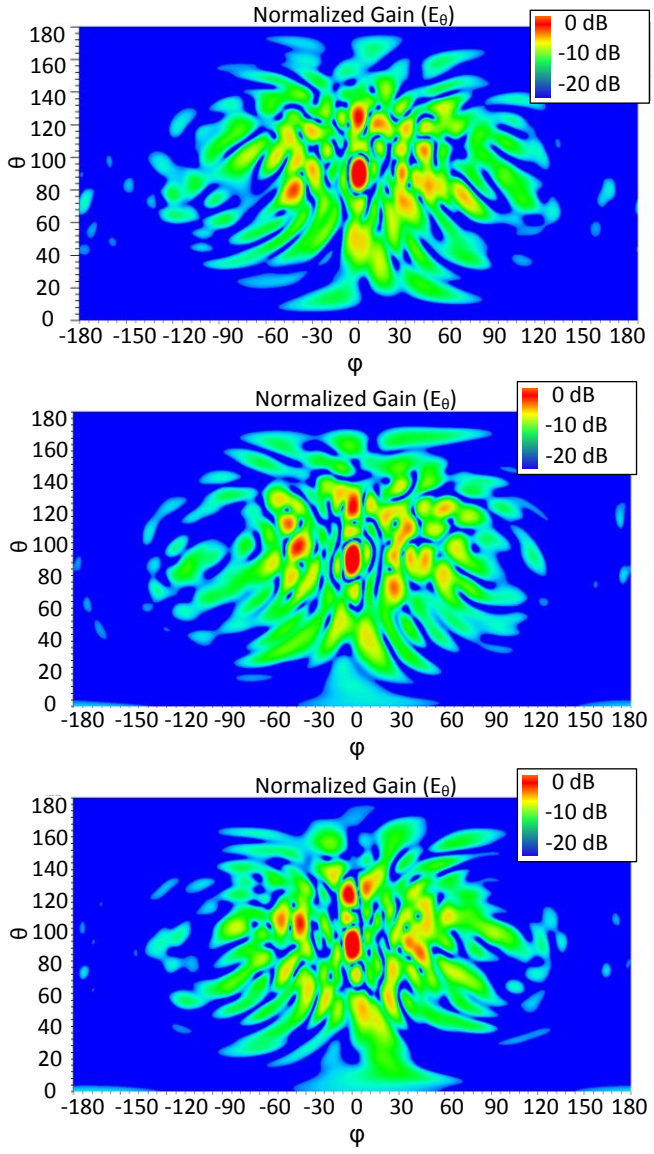


Figure 73. Simulation heatmaps for (Top) location 1, (middle) location 2, and (bottom) location 3

The simulation heatmaps show that the side lobes change angular position as the cluster changes shape but that the main beam stays in place and maintains the gain. The shape of the main beam changes slightly (see locations 1 versus location 3); however, the total

beam width is preserved. The heatmaps also indicate the presence of a strong peaking side lobe about 30 degrees above the main beam through all locations. These first results of the slow morphing cluster indicate that the framework is capable of measuring nodes in space and observing the radiation pattern of a cluster as it changes shape through space.

Fast Morphing Cluster

Like the slow morphing cluster, a fast morphing cluster was created that forced elements to move larger distances in location changes throughout the measurements. The fast morphing cluster provides configurations that do not maintain similar morphologies as compared to the slow moving cluster. This methodology resembles a swarm of coordinated random movements of UAVs or other objects that quickly move from one position to the next and attempt to maintain a primary main beam at a specific point of reference.

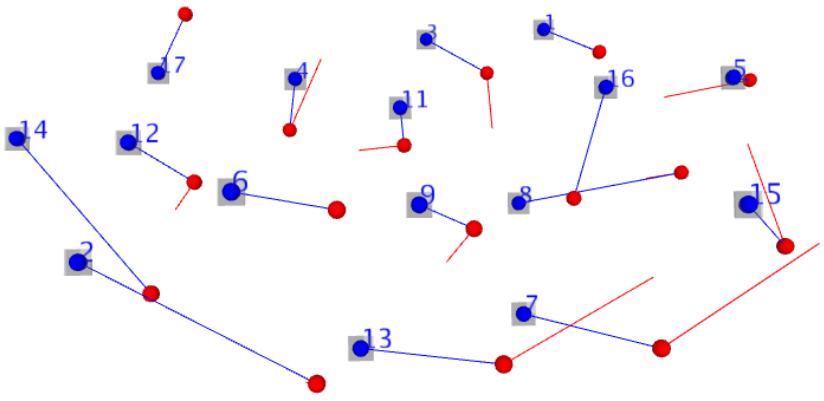


Figure 74. Physical geometry of all three locations for fast morphing cluster

As aforementioned and shown in previous figures, the red dots in Figure 74 represent the second position of the morphing cluster and the blue dots represent the third and final position of the dots. The termination of the red lines represents the original starting positions of the elements. In this experiment, the elements moved a much greater distance and in a choreographed manner. Nodes 13 and 7 moved together as a pair as did nodes 14 and 2. Other nodes seemingly took random paths non choreographed paths with other surrounding nodes.

Table 11. Aperture dimensions of locations for Fast Morphing Test

Location	X Distance (λ)	Y Distance (λ)	Z Distance (λ)	Radius (λ)
1	6.73	4.00	4.52	3.71
2	6.52	3.98	4.54	4.04
3	7.37	3.72	4.54	3.77

These dimensions reflect a consistent spacing in the x and z directions with a decreasing aperture distance in the y direction; however, the overall bounding sphere for the structure stayed approximately the same distance as the nodes moved throughout the cluster.

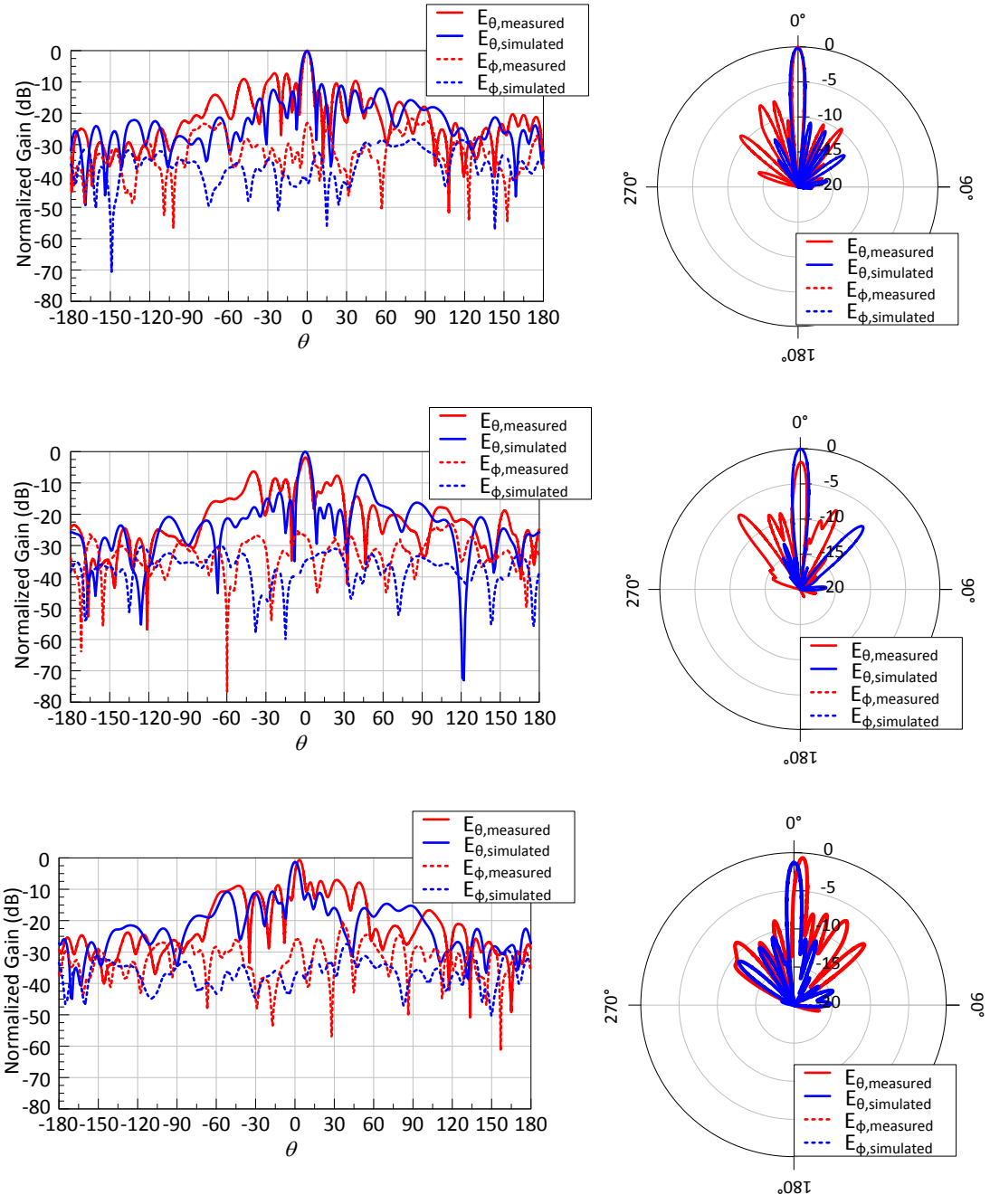


Figure 75. Measured and simulated normalized broadside radiation patterns for (Top) location 1, (middle) location2, and (bottom) location 3

The radiation patterns shown in Figure 75 demonstrate the ability of the morphing cluster to position a main beam at a point of reference as the cluster quickly morphs and changes overall shape. Consequently, the radiation patterns also have been normalized to the pattern that exhibited the highest gain so that the radiation pattern could properly be observed at a single point, or in this case, the receiving antenna in the anechoic chamber. In location 2 and location 3, the measured gain drops from the original gain by a factor of 1.7 dB and 0.6 dB respectively. Only the third location for the simulated case shows a decrease in the overall gain. The measured main beam in location three also exhibits a slight angular shift of 2 degrees.

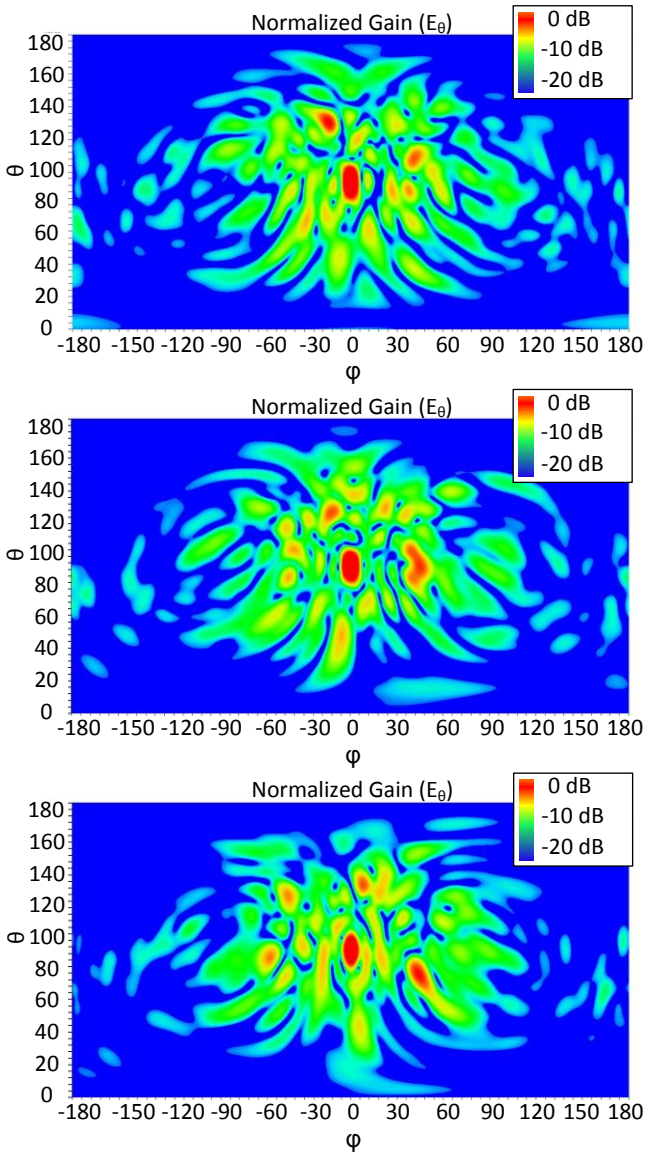


Figure 76. Simulation heatmaps for (Top) location 1, (middle) location 2, and (bottom) location 3

The simulation heatmaps demonstrate the ability for the cluster to maintain the main beam at the desired radiation point while the side lobes shift around in varying angles

and gain intensities through each transition. In particular, location 1 exhibited a comparatively high peaking side lobe than the other two locations exhibited.

Beam steering experiments were also taken to help verify the validity of these results by demonstrating that the framework and phased array controller still functioned correctly through various reprogramming and scanning instances.

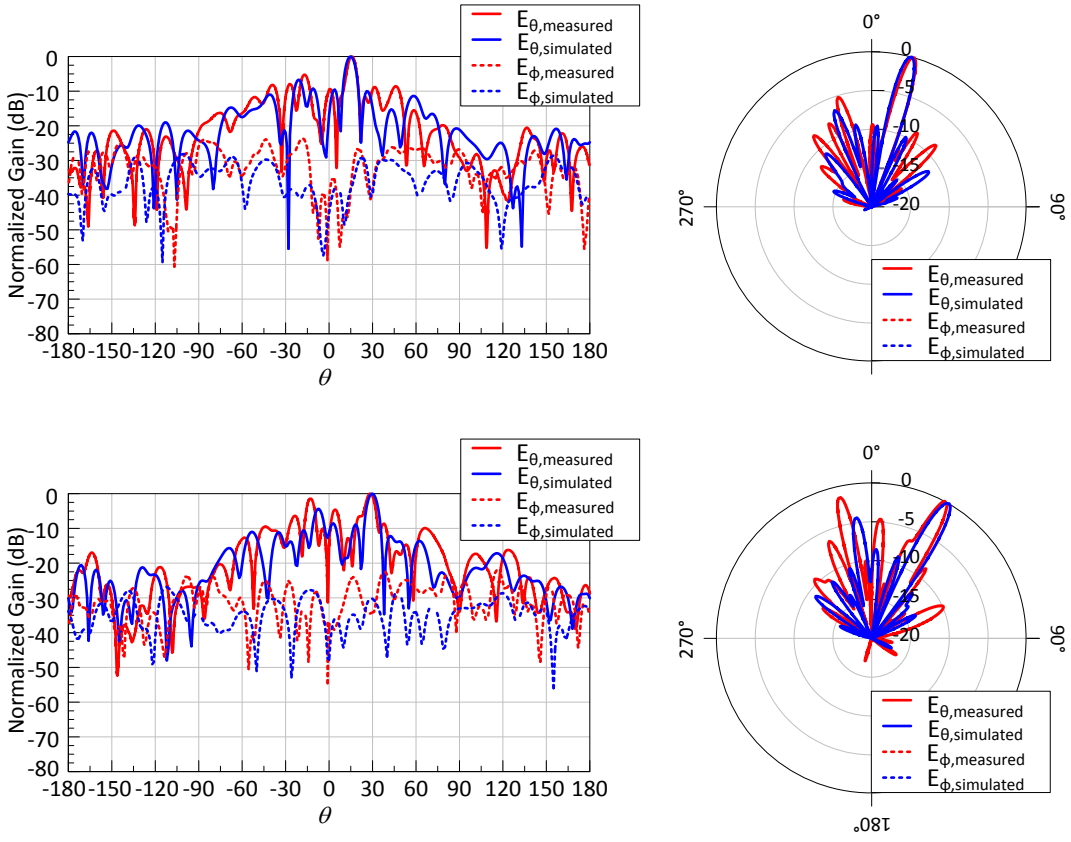


Figure 77. Simulated and measured radiation patterns for steering angles (Top) +15 degrees and (Bottom) +30 degrees for location 3

The measured steering patterns for location 3 demonstrate accurate beam steering to within 1 degree of the simulated patterns. These results emphasize that the controller was programmed correctly throughout the morphing cluster movement and produced accurate beam steering results upon reaching the final location. It is observable that the average side lobe level of the measured results is higher than those of the simulated results but this was also the case with many of the other test patterns. Overall, the fast morphing cluster test demonstrated the ability of the framework to accurately track nodes as they moved through a cluster quickly. Measured broadside radiation patterns showed that the main beam of the aperiodic array could be maintained at an observation points with some effects to gain and angular steering. The framework and system was still able to perform exhibiting accurate beam steering results in the final location.

Single Element Move

The radiation pattern response for morphing clusters has been explored through two different experiments. The output of both of these patterns demonstrate the morphing clusters ability to maintain a main beam at a location but these clusters also exhibited random side lobe behaviors as the cluster morphed into different shapes. In this experiment the goal is to determine the effects on the radiation pattern, if any, from moving a single element in the morphing array cluster. This experiment was configured in three different locations. The two configurations after the starting configuration feature one element that moves on its own trajectory while the rest of the elements stay equidistant with respect to one another in the cluster.

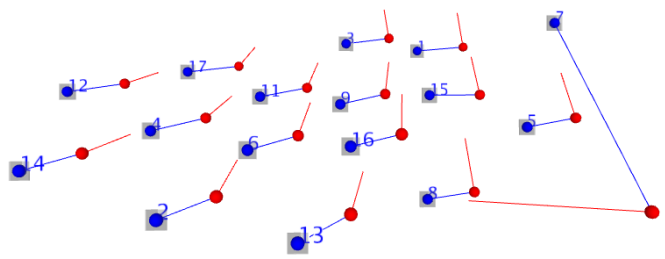


Figure 78. Physical geometry of all three locations for single element move test

From Figure 78, it is clear that element 7 moves on its own trajectory while the rest of the elements maintain an equal distance from one another. As per the other experiments, location 2 is represented by the red dot and location 3 is represented by the blue dot. Location 1 is defined as the tip of the red line opposite of the red dot. Element 7 moves at least twice the distance of the other element movements during movement to location 2 and moves at least 3 times greater from location 2 to location 3. The total distance moved by element 7 through this configuration is approximately four wavelengths.

Table 12. Aperture dimensions of locations for Single Element Move test

Location	X Distance (λ)	Y Distance (λ)	Z Distance (λ)	Radius (λ)
1	6.81	3.2	3.43	3.98
2	6.9	3.3	5.52	4.62
3	7.14	4.44	5.06	4.88

The bounding sphere for this configuration is increases to nearly a wavelength greater due to the movement from element 7. The most significant increase or decrease for the geometry happened in the z direction as the element moved from its original position and retreated to the back of the volume away from the Kinect camera. The x distance was held nearly constant while the y distance saw a similar increase as the z distance. This type of morphing cluster movement should see a change in the broadside pattern as the z distance was change significantly between the various cluster formations for measurement. This broadside distance change would force a recalculation of the phases that equate to the overall broadside radiation pattern. The y distance would also provide a variation in the phased elements for scanning off of broadside. The varying distances force the cluster and the respective phasing of each element to recalculate the phase weightings based on the positions. In this case, because the movements are significantly more than partial wavelengths, it is expected that the phase weightings and overall radiation pattern will change.

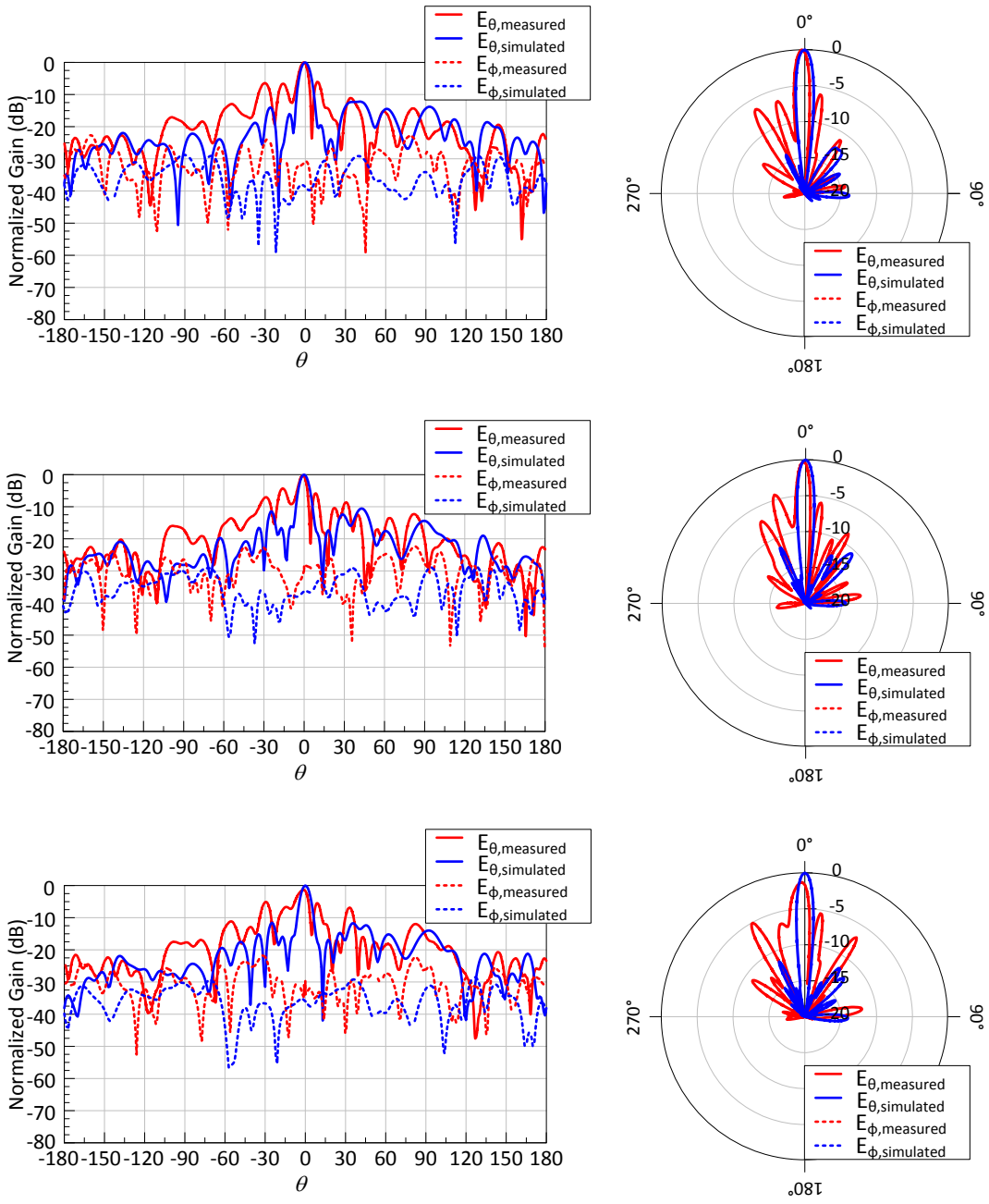


Figure 79. Measured and simulated normalized broadside radiation patterns for (Top) location 1, (middle) location 2, and (bottom) location 3

The measured radiation patterns for this experiment show relatively high side lobe levels compared to the other experiments as the side lobe level average recedes only around 4 dB from the main beam gain level. All three experiments have been normalized to the maximum gain radiation pattern in both the measured and simulated cases to demonstrate the ability to keep the main beam focused on an observation point. It is clear that location 3 experienced some difficulty producing a main beam and a qualitatively, 'good' radiation pattern as the gain was less than the first two locations and the main beam pattern is clipped with the neighboring side lobes.

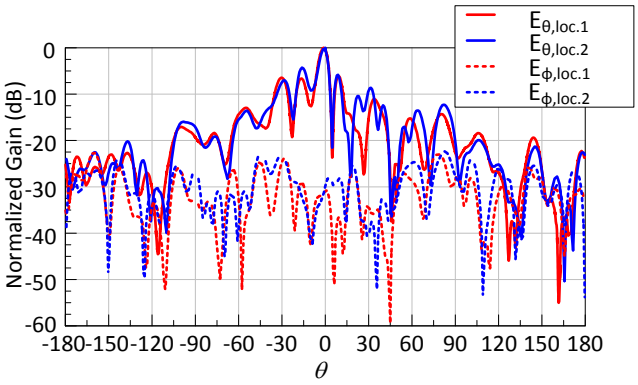


Figure 80. Combined measured radiation patterns from location 1 and location 2

Although the relative radiation patterns were not as clear as the previous experiments, it is interesting to note the minimal pattern shape change in this particular cut plane between location 1 and location 2. In the negative theta scanning plane the pattern footprint is similar in location 1 and 2, differing in the peak side lobe level by 1.2 dB;

however, on the positive theta scanning angle, one of the primary side lobe areas experiences a change in both gain and the overall shape of the side lobe pattern.

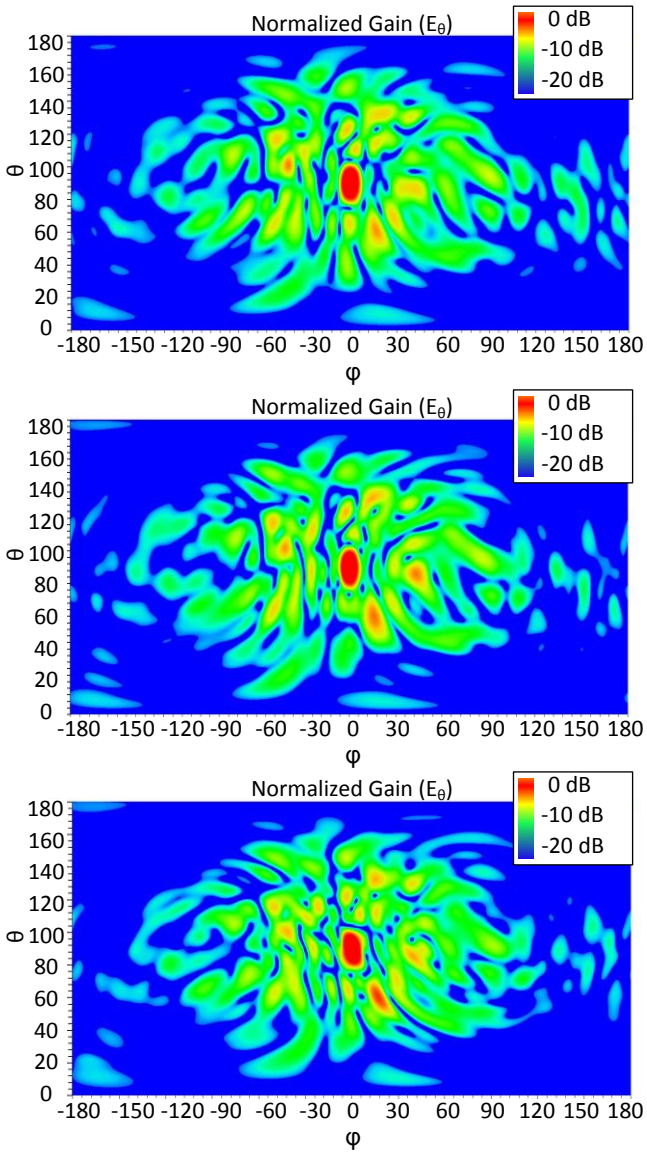


Figure 81. Simulation heatmaps for (Top) location 1, (middle) location 2, and (bottom) location 3

The heatmaps also show similar behavior among the three locations that was observed in the singular cut planes of the measured results. The pattern comparison of location 1 and location 2 demonstrate two radiation patterns that are similar in overall shape. The pattern of location 2 also demonstrates side lobe peak levels 2 dB higher in comparison with location 1, which was a similar effect that was observed in the measured results. The simulation heatmaps also show radiation patterns that demonstrate the ability of the cluster to maintain the main beam at an observation point with minimal side lobe change despite one element moving significantly in the structure. Overall, this experiment helped demonstrate the ability of a morphing cluster to permit one element to move while maintaining a main beam at an observation point and showed the possibility that a radiation pattern, including side lobe levels, may exhibit similar behaviors as one element moves in the array.

Line, Plane, and Volume

This experiment focused on specific types of spatial geometries and how these geometries affect the overall radiation pattern. This test specifically focused on bringing elements from a single dimension, a line, to expanding dimensions such as planes and volumes. The goal of this experiment is to observe how the main beam radiation pattern changes depending on the specific type of geometry used. The amount of inference from the various patterns will be limited; however, the larger changes such as beam shape can be observed.

Figure 82. Starting physical geometry of Line, Plane, and Volume test

The starting physical geometry for this experiment is shown above. The elements were placed in a line like structure. The actual grey squares representing the antenna elements do not reflect actual size but rather the size of the patch itself. Not all of the antennas could be placed in a single line due to physical constraints of the apparatus as some of the plastic interfered with elements joining the line. A total of 10 elements comprised the beginning line structure.

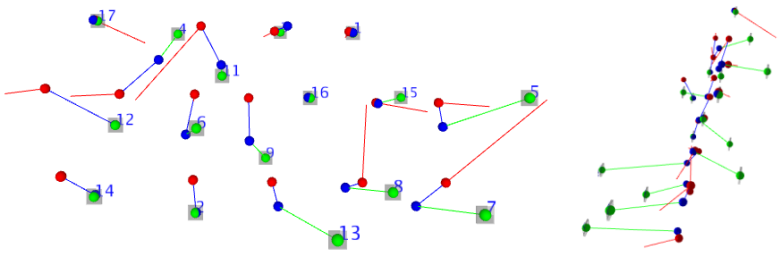


Figure 83. Physical geometry of all four locations for Line, Plane, and Volume test (Left) front view (Right) side view

As with the previous experiments, the red dots represent location 2, blue dots represent location 3, and the green dots represent location 4. Additionally from the figure above, it is clear that after the starting line geometry, a few of the elements on the left side of the structure move from the line configuration to a plane configuration. A second plane configuration is then achieved by moving all of the elements into a 2-dimensional plane structure. Finally, the volume is achieved by moving elements closer and further away from the Kinect as shown in the side view in the figure above. Some elements were not moved during the location reconfiguration as the physical apparatus prevented movement of the elements due to movements of other elements.

Table 13. Aperture dimensions of locations for line, plane, and volume test

Location	X Distance (λ)	Y Distance (λ)	Z Distance (λ)	Radius (λ)
1	8.79	2.85	2.63	4.81
2	6.61	3.25	1.42	3.5
3	5.92	3.24	1.38	3.36
4	6.9	3.23	3.25	4.01

The line configuration starts with a slight distance due to the apparatus' ability to support 10 elements in a single line. As the apparatus moves to the first plane configuration, the z distance can be decreased as elements have moved out of the line allowing other elements to be configured nearly co-planar with one another. The third location realizes a more compact plane as seen by the reduced *x* value and *z* value. Finally, the volumetric geometry increases in both *x* and *z* distances. The transformation of this type of geometry demonstrates the ability of a cluster to take on drastically different radiation patterns based on the geometry. The initial line structure should create a radiation pattern that is similar to a linear array, although the slight difference in distances will affect the radiation pattern. The second movement to a plane requires the cluster to stay in a plane but is now affected by two phase components. The third and fourth volumes are additionally affected due to the increasing distances and respective z distance in the fourth movement.

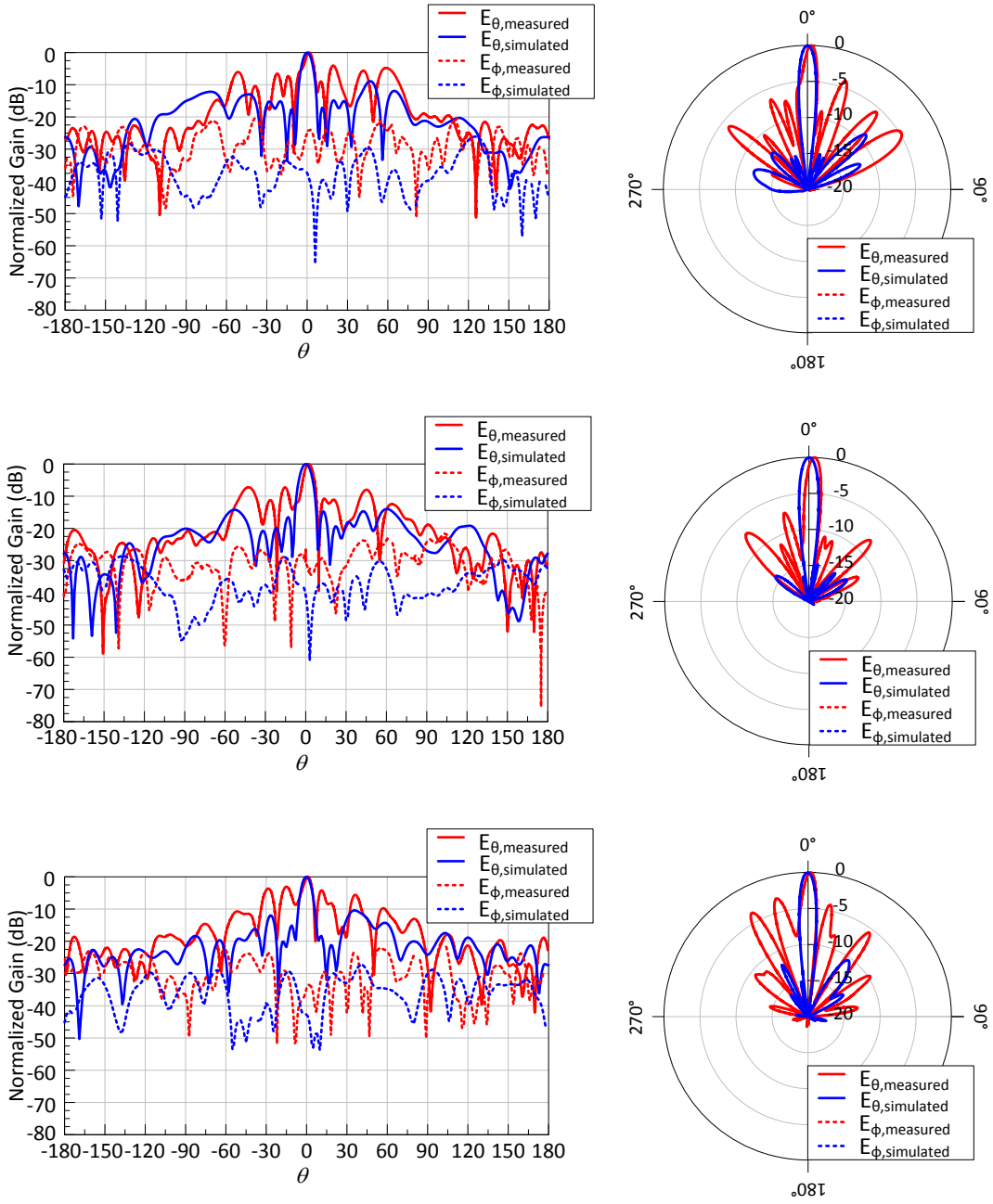


Figure 84. Measured and simulated normalized broadside radiation patterns for (Top) location 2, (middle) location 3, and (bottom) location 4

In the figure above, location 2 is the first plane, location 3 is the second plane, and location 4 is the volume. Data for the line is not available in this particular experiment as the data became corrupt after processing the analysis from the anechoic chamber. The radiation patterns have been normalized to the highest gain scenario for the two plane measurements and the volume measurement. It is clear from the radiation patterns that the main beam can be kept at a point of interest while the side lobe levels fluctuate through the different configurations. The main difference in beam shape and side lobe behavior cannot be observed in this particular cut plane but can be observed in other cut planes as will be shown in the following simulation heatmaps.

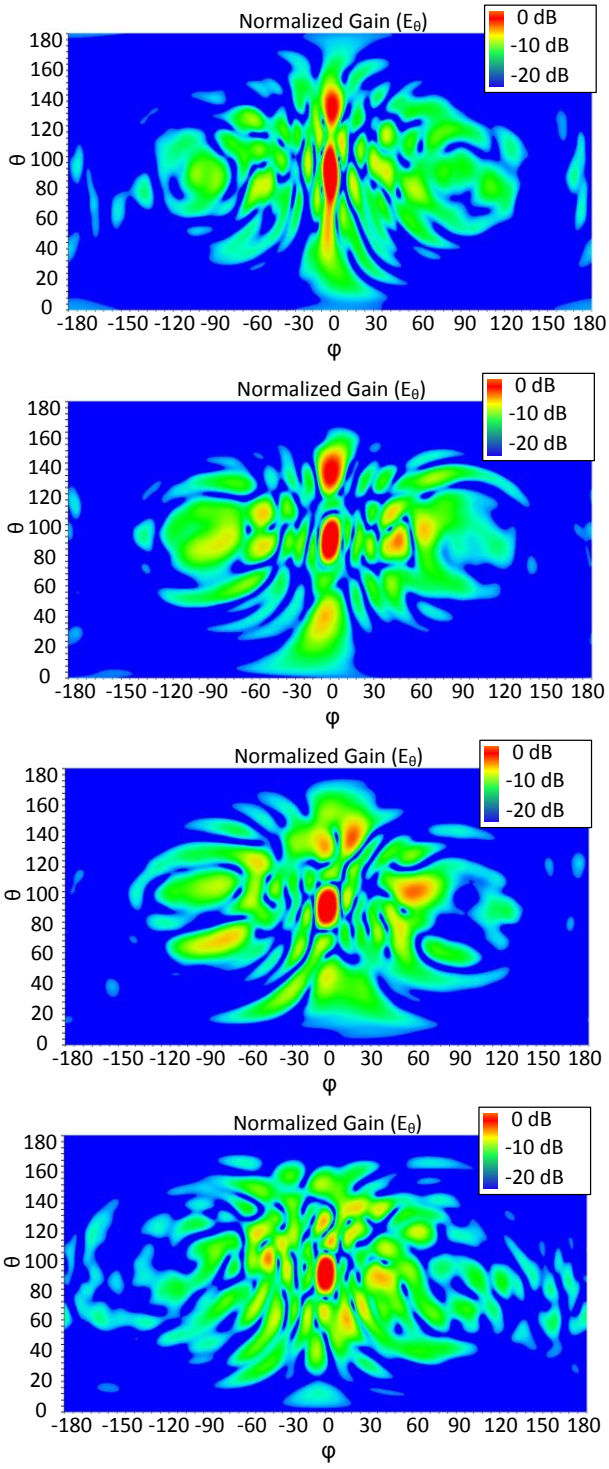


Figure 85. Simulation heatmaps for (Top) line, (2nd) plane 1, (3rd) plane 2, and (4th) volume

From Figure 85, it is clear to see from the simulation heatmaps how the beam shape and gain of the side lobe levels change through the various transformations of the structure. Notably, the peaking side lobe is consistent in the line and first plane configuration. This side lobe disappears with the configuration of the second plane and the final volumetric configuration. It is also clear that the measurement scan plane, according to the simulation heatmaps, would not yield large differences in the beam width if the framework was configured correctly. Note that the beam width in the phi direction (in the simulation heatmaps) is nearly constant. This is also shown in the measurements provided in Figure 84. Overall, this experiment demonstrated that the framework can track multiple dimensions and planes; however, further investigation is needed to gain more constructive insight into specific morphing topologies from single to multiple dimensional geometries.

Minimum Element

Testing multiple geometries and morphing clusters have provided great insight into the framework and aperiodic arrays. One of the final tests performed with the framework focused on discovering the performance of an aperiodic array with respect to the amount of active elements and contributing nodes in the array. In an application space, this emulates a swarm of UAVs in a cluster where not all elements can actively participate in producing the radiation pattern for some given reason; however, the presence of these non contributing nodes in the array can still have an effect on the overall radiation pattern. This framework already consisted of a relatively small array of only 16 elements. Therefore, based on the array theory presented in Chapter III, the

effects of disabling contributing elements should be dramatic and noticeable from the radiation patterns. The goal of this experiment was not to specifically find a 'breaking point' of the array or a scenario where the radiation pattern was not produced. It was to simply observe the effect of non contributing nodes in an aperiodic array.

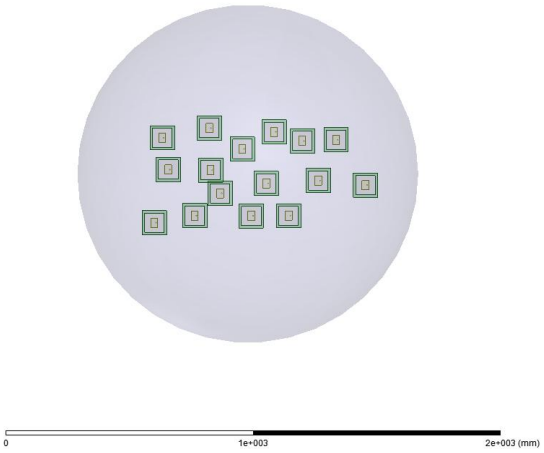


Figure 86. Physical geometry for the Minimum Element test

The physical geometry used for the minimum element test was configured to be similar to the volumes tested with the single element tests as they produced radiation patterns where the main beam was defined and side lobe levels were 7 to 8 dB down from the main beam.

Table 14. Aperture dimensions of locations for Minimum Element test

Location	X Distance (λ)	Y Distance (λ)	Z Distance (λ)	Radius (λ)
1	6.91	3.12	3.27	4.03

The aperture dimensions for the minimum element test exhibit properties similar to the single tests that were performed earlier. Because the element positioning never changes in this test, only one location is provided in the table above. Elements were turned off at random in five different tests and the signal cables were terminated in a matched load to minimize reflections from an open circuit port.

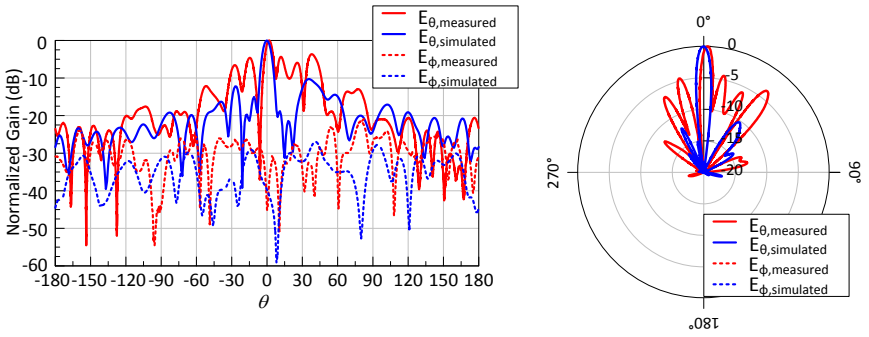


Figure 87. Measured and simulated normalized broadside radiation patterns for (Top) all elements, (2nd) missing elements 1-2, (3rd) missing elements 1-4, (4th) missing elements 1-7, and (bottom) missing elements 1-11

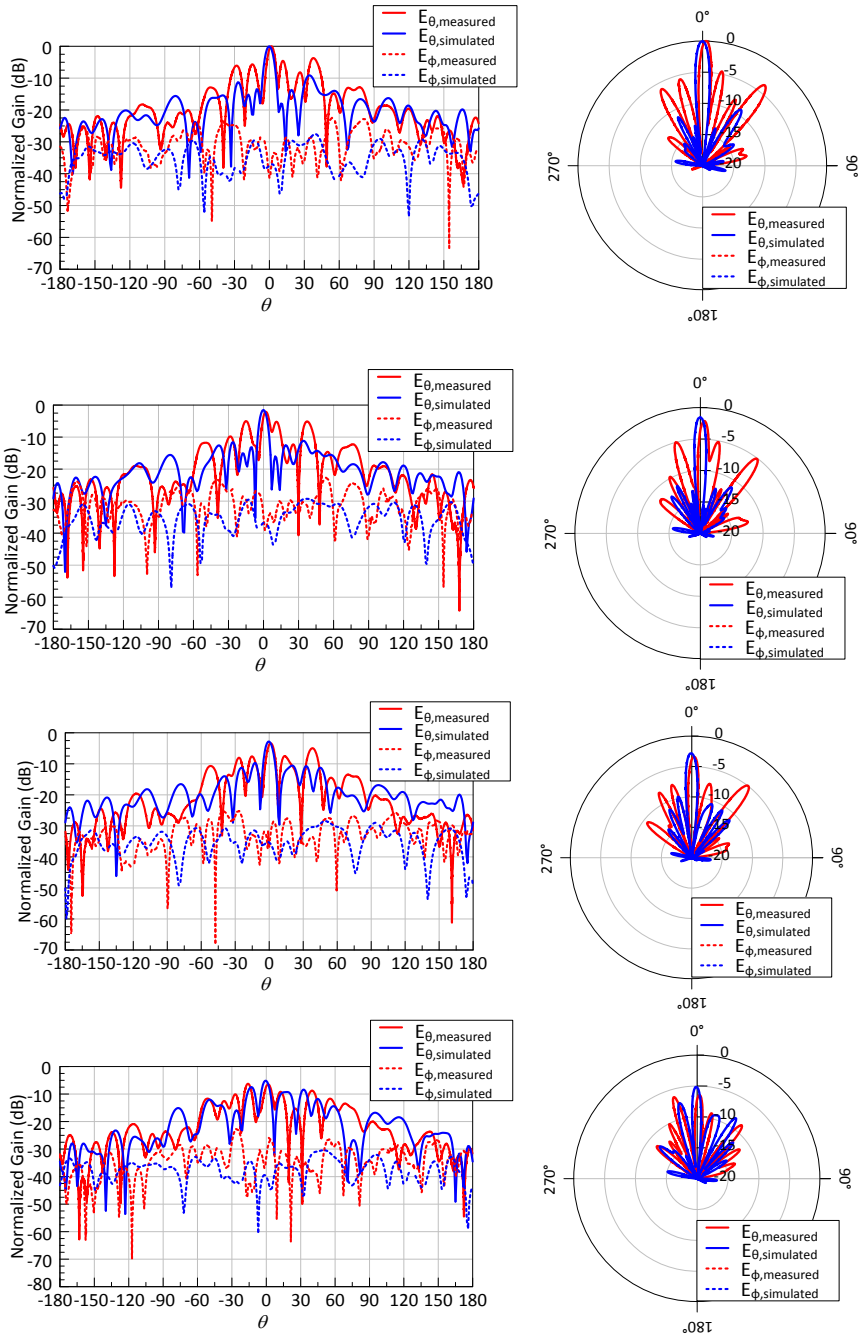


Figure 87. Continued

The radiation pattern shown at the top of Figure 87 initially has a peaking side lobe in the scan plane that is only 5 dB down from the main lobe which is slightly higher in comparison with the earlier tests. As in the previous tests, each of the radiation patterns has been normalized to the pattern with the greatest gain. Each of the subsequent radiation patterns shows a reduction in overall gain due to the loss of contributing nodes to the array. Additionally, the side lobes exhibit increasing gain relative to the gain of the main beam as additional elements are removed through each phase. The measured results follow the general pattern of the simulated results with respect to this increasing side lobe level through the different configurations. The overall radiation pattern however shows higher side lobes in the measured radiation patterns compared to the simulated radiation patterns.

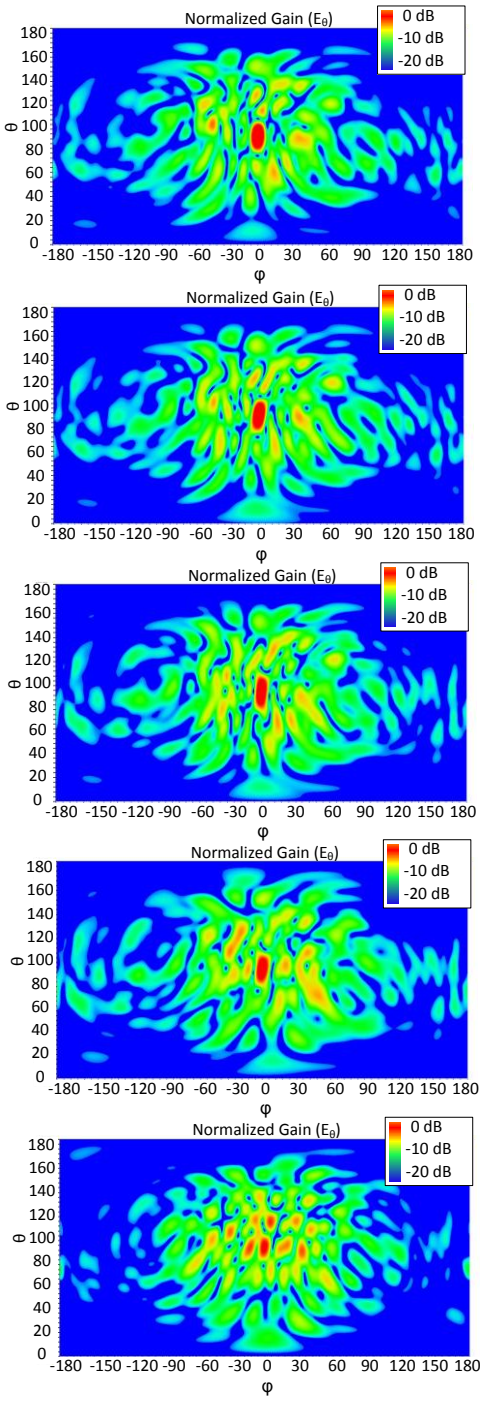


Figure 88. Simulation heatmaps for (Top) all patches, (2nd) missing elements 1-2, (3rd) missing elements 1-4, (4th) missing elements 1-7, and (bottom) missing elements 1-11

The simulation heatmaps also show the side lobe level gain through each iteration and the displacement of gain from the main beam into the side lobes. The final iteration shows a main beam that has nearly half of the starting beam width with several side lobe levels within 1.5 dB of the main beam. Overall, this measurement has correlated with the theory to show that more elements reduce the side lobe levels of aperiodic arrays. Additionally, a main beam can still be observed with 4 elements; however, the performance of the radiation pattern is significantly less than the 16 element pattern.

Summary of System Tests

Overall, the system tests of the framework demonstrated the ability to track various geometries and measure the respective radiation patterns. Simulated and measured results were presented that showed alignment of theory and practice; however, the measured results consistently displayed an average side lobe level of 4 dB higher than the simulated results in some cases. The simulated and measured patterns provide useful insight into the operation of aperiodic arrays and have shown that the formation of main beams and the control of these main beams are possible. Additional morphing tests have also shown that swarming clusters can maintain a main beam at an observation point despite elements moving in the cluster. The reduction of elements in the swarm also demonstrates the ability to maintain a main beam at a desired observation point at the cost of increased side lobe levels.

CHAPTER VII

CONCLUSION AND FUTURE WORK

A computer vision assisted framework has been engineered to study and interact with phased arrays formed by randomly distributed clusters of antennas. It provides a viable tool to study and control the radiation behavior and other characteristics of volumetric, planar, and other distributions as elements move and the topology of the cluster morphs. Complex control, element identification, and beamsteering have been experimentally demonstrated for a number of test cases. Simulation and measured results for basic beam steering scenarios illustrated the overall effectiveness and utility of the system. The system was also used to track morphing clusters and swarms of elements while maintaining a desired beamforming radiation behavior through each bounding geometry. This presents an opportunity for further investigation into radiation characteristics of morphing swarms and how complex geometries in dynamic choreographed movements can be controlled to achieve desired radiation behavior.

Additionally, tests were conducted on the framework itself to determine the accuracies of the Kinect and phased array control system as viable options to study aperiodic arrays. Based on the system tests and overall performance of the results achieved, the Kinect and coupled phased array controller provide adequate ability to study aperiodic arrays in this type of framework. This type of computer vision processing has also been shown to exist in several other robotics and autonomous

systems applications making it a viable option for future study of swarming random aperiodic arrays.

The primary contribution of this work to be further studied is the overall assessment and capability of a low cost framework to study aperiodic arrays. Before this study, only a few volumetric arrays and their radiation characteristics had been studied. The construction and design of this framework have shown that a rapid measurement and development tool for aperiodic arrays can be created and used for the future study of swarming clusters and their electromagnetic properties. This work has demonstrated the capability of a low cost computer vision framework to serve as a viable platform to study the future of morphing swarms and their electromagnetic capabilities. This platform, or iterations of this platform, can provide valuable insight into the overall electromagnetic synchronization problem that is currently ongoing in the field. These results demonstrate the aperiodic arrays can collectively form useful radiation patterns and can allow beam steering and other advanced control methods to reconfigure the radiation pattern to desired states.

Future work of this platform will include the advancement of the computer vision system and the types of elements or nodes that it will track. This framework presented nodes that were powered over a wired connection. This was done to simplify the communications and power distribution within the system; however, the next logical step for this framework is to analyze wireless systems that utilize a brute force method of electromagnetic synchronization. Many open source quad or nano copters exist that can easily be programmed and outfitted with an embedded LED control system.

Additionally, more advanced cameras and imaging technologies can be leveraged to further enhance the computer vision and image processing capabilities of this framework.

In conclusion, this framework has demonstrated previous developed theory on aperiodic arrays through a series of successful radiation pattern experiments. Accordingly, a valuable platform utilizing computer vision has been developed as the backbone of this framework and has demonstrated the capability of tracking and measuring morphing swarms of elements.

REFERENCES

[1] M. F. A. Ahmed and S. A. Vorobyov, "Collaborative beamforming for wireless sensor networks with Gaussian distributed sensor nodes," *Wireless Communications, IEEE Transactions on,* vol. 8, pp. 638-643, 2009.

[2] Y. Chen, J. Dickens, S. Holt, D. Reale, J. Mankowski, and M. Kristiansen, "Synchronization of phased array pulsed ring-down sources using a GPS based timing system," in *Power Modulator and High Voltage Conference (IPMHVC), 2010 IEEE International*, 2010, pp. 624-627.

[3] Y. L. David Jenn, Tong Chin Hong Matthew, Yeo Eng Choon, Ong Chin Siang, Yeo Siew Yam, "Distributed Phased Arrays and Wirless Beamforming Networks," *International Journal of Distributed Sensor Networks,* vol. 5, pp. 283-302, 2009.

[4] H. Ochiai, P. Mitran, H. V. Poor, and V. Tarokh, "Collaborative beamforming for distributed wireless ad hoc sensor networks," *Signal Processing, IEEE Transactions on,* vol. 53, pp. 4110-4124, 2005.

[5] S. PalChaudhuri, A. K. Saha, and D. B. Johnsin, "Adaptive clock synchronization in sensor networks," in *Information Processing in Sensor Networks, 2004. IPSN 2004. Third International Symposium on*, 2004, pp. 340-348.

[6] R. Baldoni, A. Corsaro, L. Querzoni, S. Scipioni, and S. T. Piergiovanni, "Coupling-Based Internal Clock Synchronization for Large-Scale Dynamic Distributed Systems," *Parallel and Distributed Systems, IEEE Transactions on,* vol. 21, pp. 607-619, 2010.

[7] K. Buchanan and G. H. Huff, "A Stochastic Mathematical Framework for the Analysis of Spherically-Bound Random Arrays," *Antennas and Propagation, IEEE Transactions on,* vol. 62, pp. 3002-3011, 2014.

[8] D. King, R. Packard, and R. Thomas, "Unequally-spaced, broad-band antenna arrays," *Antennas and Propagation, IRE Transactions on,* vol. 8, pp. 380-384, 1960.

[9] C. Ling, S. Wee, Y. Zhu Liang, S. Rahardja, and C. Wei, "Linear Sparse Array Synthesis With Minimum Number of Sensors," *Antennas and Propagation, IEEE Transactions on,* vol. 58, pp. 720-726, 2010.

[10] Y. Lo, "A probabilistic approach to the design of large antenna arrays," *Antennas and Propagation, IEEE Transactions on,* vol. 11, pp. 95-96, 1963.

[11] Y. T. Lo, "A mathematical theory of antenna arrays with randomly spaced elements," *Antennas and Propagation, IEEE Transactions on,* vol. 12, pp. 257-268, 1964.

[12] Y. Weijun and W. Yuanxun Ethan, "Beamforming for Phased Arrays on Vibrating Apertures," *Antennas and Propagation, IEEE Transactions on,* vol. 54, pp. 2820-2826, 2006.

[13] V. Agrawal and L. Yuen, "Mutual coupling in phased arrays of randomly spaced antennas," *Antennas and Propagation, IEEE Transactions on,* vol. 20, pp. 288-295, 1972.

[14] L. Medina, E. Diaz, E. Rubio, and K. Rodriguez, "Design of a planar array of sensors for 3D location using genetic algorithms," in *Ultrasonics, 2003 IEEE Symposium on*, 2003, pp. 1384-1387 Vol.2.

[15] H. Unz, "Linear Arrays with arbitrarily distributed elements," *Antennas and Propagation, IRE Transactions on,* vol. 8, pp. 222-223, 1960.

[16] M. C. Vigano, G. Toso, P. Angeletti, I. E. Lager, A. Yarovoy, and D. Caratelli, "Sparse antenna array for Earth-coverage satellite applications," in *Antennas and Propagation (EuCAP), 2010 Proceedings of the Fourth European Conference on*, 2010, pp. 1-4.

[17] G. H. Huff, J. F. Chamberland, and J. S. Jensen, "Cognitive Motion-Dynamic Tethering of a Phased Array to an Android Smartphone," *Antennas and Propagation, IEEE Transactions on,* vol. 62, pp. 1093-1101, 2014.

[18] S. Kun, N. Peng, and C. Wang, "Secure and resilient clock synchronization in wireless sensor networks," *Selected Areas in Communications, IEEE Journal on*, vol. 24, pp. 395-408, 2006.

[19] A. R. Swain and R. C. Hansdah, "A Weighted Average Based External Clock Synchronization Protocol for Wireless Sensor Networks," in *Distributed Computing Systems Workshops (ICDCSW), 2011 31st International Conference on*, 2011, pp. 218-229.

[20] Y. Zhe, H. Liang, C. Lin, and P. Jianping, "Temperature-Assisted Clock Synchronization and Self-Calibration for Sensor Networks," *Wireless Communications, IEEE Transactions on*, vol. 13, pp. 3419-3429, 2014.

[21] J. Bian, J. Yin, J. Wei, and L. Zhang, "Hand detection based on depth information and color information of the Kinect," in *Control and Decision Conference (CCDC), 2015 27th Chinese*, 2015, pp. 4205-4210.

[22] Z. Changchen, C. Weihai, Z. Zhiwen, and L. Jingmeng, "An RGBD data based vehicle detection algorithm for vehicle following systems," in *Industrial Electronics and Applications (ICIEA), 2013 8th IEEE Conference on*, 2013, pp. 1506-1511.

[23] L. Cruz, D. Lucio, and L. Velho, "Kinect and RGBD Images: Challenges and Applications," in *Graphics, Patterns and Images Tutorials (SIBGRAPI-T), 2012 25th SIBGRAPI Conference on*, 2012, pp. 36-49.

[24] L. Cuiling, X. Jizheng, and W. Feng, "Object-based coding for Kinect depth and color videos," in *Visual Communications and Image Processing (VCIP), 2012 IEEE*, 2012, pp. 1-6.

[25] V. R. Duseev and A. N. Malchukov, "Kinect sensor depth data filtering," in *Mechanical Engineering, Automation and Control Systems (MEACS), 2014 International Conference on*, 2014, pp. 1-4.

[26] P. Fankhauser, M. Bloesch, D. Rodriguez, R. Kaestner, M. Hutter, and R. Siegwart, "Kinect v2 for mobile robot navigation: Evaluation and modeling," in *Advanced Robotics (ICAR), 2015 International Conference on*, 2015, pp. 388-394.

[27] A. V. Gulalkari, H. Giang, P. S. Pratama, K. Hak Kyeong, K. Sang Bong, and J. Bong Huan, "Object following control of six-legged robot using Kinect camera," in *Advances in Computing, Communications and Informatics (ICACCI, 2014 International Conference on*, 2014, pp. 758-764.

[28] F. Jurado, G. Palacios, and F. Flores, "Vision-Based Trajectory Tracking on the 3D Virtual Space for a Quadrotor," in *Electronics, Robotics and Automotive Mechanics Conference (CERMA), 2012 IEEE Ninth*, 2012, pp. 31-36.

[29] A. Kanezaki, T. Harada, and Y. Kuniyoshi, "Scale and rotation invariant color features for weakly-supervised object Learning in 3D space," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, 2011, pp. 617-624.

[30] H. Lili and M. Barth, "Tightly-coupled LIDAR and computer vision integration for vehicle detection," in *Intelligent Vehicles Symposium, 2009 IEEE*, 2009, pp. 604-609.

[31] T. Mallick, P. P. Das, and A. K. Majumdar, "Characterizations of Noise in Kinect Depth Images: A Review," *Sensors Journal, IEEE*, vol. 14, pp. 1731-1740, 2014.

[32] S. Miyata and Y. Yashiki, "Evaluation of Kinect vision sensor for bin-picking applications: Improved component separation accuracy with combined use of depth map and color image," in *Control Automation Robotics & Vision (ICARCV), 2014 13th International Conference on*, 2014, pp. 94-99.

[33] T. Nakamura, "Real-time 3-D object tracking using Kinect sensor," in *Robotics and Biomimetics (ROBIO), 2011 IEEE International Conference on*, 2011, pp. 784-788.

[34] P. J. Noonan, T. F. Cootes, W. A. Hallett, and R. Hinz, "The design and initial calibration of an optical tracking system using the Microsoft Kinect," in *Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC), 2011 IEEE*, 2011, pp. 3614-3617.

[35] H. Ocak, M. Ambarkutuk, G. Kucukyildiz, and S. Karakaya, "Image processing based package volume detection with Kinect," in *Signal Processing and Communications Applications Conference (SIU), 2015 23th*, 2015, pp. 515-518.

[36] H. C. N. Premachandra, T. Yendo, M. P. Tehrani, T. Yamazato, H. Okada, T. Fujii, *et al.*, "High-speed-camera image processing based LED traffic light detection for road-to-vehicle visible light communication," in *Intelligent Vehicles Symposium (IV), 2010 IEEE*, 2010, pp. 793-798.

[37] H. C. N. Premachandra, T. Yendo, T. Yamasato, T. Fujii, M. Tanimoto, and Y. Kimura, "Detection of LED traffic light by image processing for visible light communication system," in *Intelligent Vehicles Symposium, 2009 IEEE*, 2009, pp. 179-184.

[38] Z. Qiang, K. Lingcheng, and Z. Jianghai, "Real-time general object recognition for indoor robot based on PCL," in *Robotics and Biomimetics (ROBIO), 2013 IEEE International Conference on*, 2013, pp. 651-655.

[39] E. Whitmire, T. Latif, and A. Bozkurt, "Kinect-based system for automated control of terrestrial insect biobots," in *Engineering in Medicine and Biology Society (EMBC), 2013 35th Annual International Conference of the IEEE*, 2013, pp. 1470-1473.

[40] W. Yi-Hua, S. Ming-Hwa, and S. Chi-Chia, "Efficient object motion detection based on RGB-D image," in *Consumer Electronics - Taiwan (ICCE-TW), 2015 IEEE International Conference on*, 2015, pp. 438-439.

[41] Z. Yimin, J. Guolai, X. Guoqing, W. Xinyu, and L. Krundel, "Kinect depth image based door detection for autonomous indoor navigation," in *Robot and Human Interactive Communication, 2014 RO-MAN: The 23rd IEEE International Symposium on*, 2014, pp. 147-152.

[42] T. Yonezawa and K. Ueda, "Real-time 3D data reduction and reproduction of spatial model using line detection in RGB image," in *Soft Computing and Intelligent Systems (SCIS), 2014 Joint 7th International Conference on and Advanced Intelligent Systems (ISIS), 15th International Symposium on*, 2014, pp. 727-730.

[43] X.-l. Zhou and K. Peng, "Image Processing in Vision 3D Coordinate Measurement System," in *Image and Signal Processing, 2009. CISP '09. 2nd International Congress on*, 2009, pp. 1-5.

[44] C. Warty, R. W. Yu, K. ElMahgoub, and S. Spinsante, "MUSIC algorithm DoA estimation for cooperative node location in mobile ad hoc networks," in *Aerospace Conference, 2013 IEEE*, 2013, pp. 1-8.

[45] S. S. Stankovic, "Swarming and flocking: Cooperative collective behavior," in *Neural Network Applications in Electrical Engineering, 2008. NEUREL 2008. 9th Symposium on*, 2008, pp. 1-1.

[46] T. J. M. Sanguino and F. P. Gomez, "Improving 3D object detection and classification based on kinect sensor and hough transform," in *Innovations in Intelligent SysTems and Applications (INISTA), 2015 International Symposium on*, 2015, pp. 1-8.

[47] L. Weihua, F. Yangyu, Z. Zhang, and L. Tao, "A new method for calibrating depth and color camera pair based on Kinect," in *Audio, Language and Image Processing (ICALIP), 2012 International Conference on*, 2012, pp. 212-217.

[48] K. Khoshelham, "Accuracy Analysis of Kinect Depth Data," *International Archives of the Photogrammetry, Remote Sensing and Spatial Informaiton Sciences,* vol. XXXVIII-5/W12, pp. 133-138, August 2011 2011.

[49] O. D. Team, "Finding contours in your image," 2011.

APPENDIX A

```
/*Database Schema for MYSQL */;
CREATE DATABASE `medusa` /*!40100 DEFAULT CHARACTER SET utf8 */;
CREATE TABLE `elementpositions` (
  `ANTENNA_NUM` int(11) DEFAULT NULL,
  `ANTENNA_X` double DEFAULT NULL,
  `ANTENNA_Y` double DEFAULT NULL,
  `ANTENNA_Z` double DEFAULT NULL,
  `PIXEL1_X` double DEFAULT NULL,
  `PIXEL1_Y` double DEFAULT NULL,
  `PIXEL1_Z` double DEFAULT NULL,
  `PIXEL2_X` double DEFAULT NULL,
  `PIXEL2_Y` double DEFAULT NULL,
  `PIXEL2_Z` double DEFAULT NULL,
  `PIXEL3_X` double DEFAULT NULL,
  `PIXEL3_Y` double DEFAULT NULL,
  `PIXEL3_Z` double DEFAULT NULL,
  `PIXEL4_X` double DEFAULT NULL,
  `PIXEL4_Y` double DEFAULT NULL,
  `PIXEL4_Z` double DEFAULT NULL,
  `CONFIDENCE` decimal(10,0) DEFAULT NULL,
```

```
  `ELEMENT_ID` varchar(20) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
CREATE TABLE `elementrotations` (
  `ANTENNA_NUM` int(11) DEFAULT NULL,
  `MAC_ADDRESS` varchar(20) DEFAULT NULL,
  `YAW` int(11) DEFAULT NULL,
  `PITCH` int(11) DEFAULT NULL,
  `ROLL` int(11) DEFAULT NULL,
  `ELEMENT_ID` varchar(20) DEFAULT NULL,
  UNIQUE KEY `ELEMENT_ID_UNIQUE` (`ELEMENT_ID`),
  UNIQUE KEY `MAC_ADDRESS_UNIQUE` (`MAC_ADDRESS`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
CREATE TABLE `elements` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `MAC_ADDRESS` varchar(20) DEFAULT NULL,
  `IP_ADDRESS` varchar(20) DEFAULT NULL,
  `NETMASK` varchar(20) DEFAULT NULL,
  `CURRENT_STATE` bigint(32) DEFAULT NULL,
  `ELEMENT_ID` varchar(20) DEFAULT NULL,
  `PIXEL1_COLOR` bigint(32) DEFAULT NULL,
  `PIXEL2_COLOR` bigint(32) DEFAULT NULL,
  `PIXEL3_COLOR` bigint(32) DEFAULT NULL,
```

```
  `PIXEL4_COLOR` bigint(32) DEFAULT NULL,
  `ANTENNA_NUM` bigint(32) DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `MAC_ADDRESS_UNIQUE` (`MAC_ADDRESS`)
) ENGINE=InnoDB AUTO_INCREMENT=381 DEFAULT CHARSET=utf8;
CREATE TABLE `elements` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `MAC_ADDRESS` varchar(20) DEFAULT NULL,
  `IP_ADDRESS` varchar(20) DEFAULT NULL,
  `NETMASK` varchar(20) DEFAULT NULL,
  `CURRENT_STATE` bigint(32) DEFAULT NULL,
  `ELEMENT_ID` varchar(20) DEFAULT NULL,
  `PIXEL1_COLOR` bigint(32) DEFAULT NULL,
  `PIXEL2_COLOR` bigint(32) DEFAULT NULL,
  `PIXEL3_COLOR` bigint(32) DEFAULT NULL,
  `PIXEL4_COLOR` bigint(32) DEFAULT NULL,
  `ANTENNA_NUM` bigint(32) DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `MAC_ADDRESS_UNIQUE` (`MAC_ADDRESS`)
) ENGINE=InnoDB AUTO_INCREMENT=381 DEFAULT CHARSET=utf8;
```