Time Difference of Arrival Localization Testbed: Development, Calibration, and Automation

Kristen McClelland KMCCLELLAND3@GATECH.EDU
Hayden Flinner HAYDEN@GATECH.EDU
Randal Abler RANDAL.ABLER@GATECH.EDU
Paul Garver GARVERP@GATECH.EDU
Jaison George JGEORGE33@GATECH.EDU

Georgia Institute of Technology, North Ave NW, Atlanta, GA 30332 USA

Abstract

As congested RF environments become increasingly prevalent, understanding and learning how to manage them becomes more important. Particular applications including localization must be tolerant to high levels of interference that are common in such settings. In order to study such RF environments with Extreme Emitter Densities (EED), RF Sensor Nodes (RFSNs) have been deployed in Bobby Dodd football stadium at the Georgia Institute of Technology, as well as a corresponding laboratory testbed. A control center was established for the automated coordination and management of the nodes, enabling the recording of RF spectrum and associated metadata. GNU Radio is used to calibrate the nodes, analyze the data, and deploy Time Difference of Arrival localization algorithms as well as improve spectrum utilization. This paper discusses the hardware setup deployed in these testbeds, the networked management of the nodes, and an initial mechanism for time synchronization.

1. Introduction

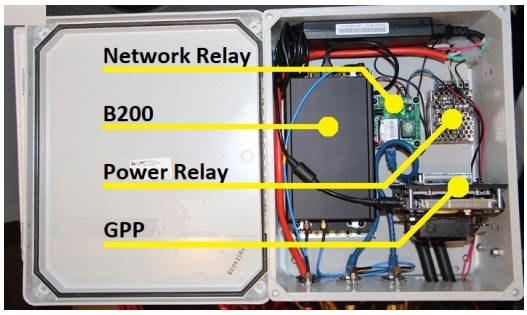
Extreme emitter density (EED) RF environments, defined as 10k-100k emitters within a footprint of less than 1 km^2, are becoming increasingly common as the number of wireless Internet-connected devices continues to increase exponentially (Garver et al., 2014). In order for all of these devices to communicate effectively, significant improvements in spectrum management must be achieved.

Before spectrum management can be achieved, the RF spectrum must first be characterized. As part of ongoing efforts to characterize the RF spectrum in an EED RF environment, the Georgia Institute of Technology Intelligent Digital Communications (IDC) Vertically Integrated Projects (VIP) team has created and deployed a software-defined radio sensor network testbed, LOC-EED, in Bobby Dodd Stadium in Atlanta, GA (Garver et al., 2015; 2017). In parallel, a testbed has been deployed in a laboratory setup for more controlled experimentation (Garver et al., 2014). The VIP program (Coyle & Krueger, 2006) is a research program consisting of multidisciplinary teams of undergraduate, graduate students and faculty advisors who work together on long term research projects. The stadium setup is ideal for real-world testing due to the multitude of devices, cell phones, radios, headsets, that are present and trying to communicate with each other at any given moment in time. By observing the RF spectrum, spectrum management applications and algorithm prototyping can be investigated in EED environments using a software-defined radio such as GNU Radio. Successfully analyzing the RF spectrum enables management of the spectrum to minimize interference by identifying and locating transmitters. Using software defined radio nodes instead of specialized hardware enables easy reconfiguration for specific applications.

Figure 1. RFSN Antenna in Stadium

The motivation behind this work is to provide an example solution for observing RF environments. By documenting our setup, management, and demonstration of successful

Proceedings of the 7^{th} GNU Radio Conference, Copyright 2017 by the author(s).



(a) Inside RFSN Enclosure

(b) RFSN Enclosure in Stadium

Figure 2. Pictures of RFSNs in the field

applications, others interested in a similar setup can use ours for reference. In addition, we are also documenting an initial mechanism for time synchronization.

These testbeds consist of multiple RF sensor nodes (RFSNs) that use wide-band RF digitizers and processors to detect the RF environment and record it (Garver et al., 2015). The RFSNs in the stadium each have a direct-conversion RF digitizer, general purpose x86-based processor (GPP), Ethernet power relay, GPS Disciplined Oscillator (GPSDO), and a 2.4/5 GHz panel antenna (Garver et al., 2015). The laboratory RFSNs have similar hardware to the stadium RFSNs except they lack the panel antenna and GPSDO. The GPSDO is replaced by a Jackson Labs LC-XO providing a 10 MHz clock and 1 PPS reference for synchronization (Garver et al., 2015). Instead of the panel antenna and GPSDO, each RFSN is cabled to splitters which enable them to receive input signals from transmitters and the 10 MHz clock and 1 PPS reference signal from a distribution board. All of the RFSNs have Ubuntu 14.04 LTS and GNU Radio installed to enable signal processing. GNU Radio allows for the data from these RFSNs to be time-correlated which enables more complex analysis of the data such as emitter localization through time difference of arrival.

The testbed setups described above are influenced by many limitations and challenges. The RFSNs are located in an outdoor football stadium which limits physical access to the devices and requires that they must be able to tolerate a wide range in weather conditions and not require a significant amount of power (Garver et al., 2014). As a result of these physical demands, the RFSNs are controlled via a remote network, requiring a robust control system, which we call RFSN Control Center (RFSNCC). As the bandwidth to this network is limited and unreliable, particularly when capturing, network-based solutions for time synchronization would be impractical and counting on 24/7 access to the RFSNs could result in losing valuable data. Previous approaches into real-time TDoA localization in small testbeds have focused on a mobile network of receivers with a common reference receiver, a fusion center, that performs all of the calculations in real-time (Schmitz & Herndez, 2016). Because the fusion center needs to receive data from the receivers it requires a real-time bandwidth-intensive back-haul which would not be possible in our EED RF environment. Due to the distance between the nodes in the stadium, it is impractical to distribute a timing reference via cable so instead a GPSDO is used.

Table 1. RFSN Components

Manufacturer	Model	Description
National Instruments	782980-01	RF Digitizer
National Instruments	783454-01	GPS Oscillator
Intel	BOXD54250WYK	Haswell i5 NUC PC
Samsung	MZ-MTEIT0BW	1TB Solid State Disk
Crucial	BLS2K8G3N169ES4	16GB DDR3 RAM
National Control Devices	R110PL_ETHERNET	Ethernet Relay
L-COM	HG2458-20P	2.4/5GHz Antenna

2. RFSN Control Center

During a football game we schedule around 100 time-synchronized recordings that are a minute in length. The recordings rotate through the four most common 20 MHz WLAN channels, and have a sample rate of 25 megasamples/sec. Repeating this schedule over a few football seasons has given us about 40TB of data.

Along with any large data set comes the challenges of managing it, such as maintaining metadata, keeping data backed up, and giving users easy access to the data. Before

the creation of the RFSN Control Center (RFSNCC), manually pulling the data from the stadium and onto our backup servers was tedious and time consuming. To schedule on all three testbed nodes at one time, an intimate knowledge of the testbed's IP addresses, logins, and commands used to launch *specrec*, our high sample rate recording tool (Lincoln & Garver, 2015), was required. Obviously, manual scheduling would not work with much more than the three nodes we had at the time, and it required an experienced operator to actively schedule and monitor each recording.

Our solution to these problems is RFSNCC. As well as providing solutions for all of the problems listed above, RFSNCC has other advantages, too. It allows for fusion of multiple data sources that were previously impractical to do when manually recording data, temperature during a recording, number of people in the stadium, and etc., could all be easily recorded alongside RF data with an automated system. The automation also greatly reduces the chances of human error causing problems during a recording session.

RFSNCCs software architecture is straightforward. A web-based interface allows recording schedules, frequencies, etc. to be entered into a central controller. The controller issues commands to multiple listeners, which are run on each RFSN in the stadium and testbed. To keep the listener software running during recording sessions, it is kept as lightweight as possible, leveraging Python's *Falcon* module to serve our API. Upon being told to schedule a recording at a certain time, a listener schedules a recording in Linux's *atq* and goes back to listening for commands. The controller, which was implemented in Python's *Django* framework, reports the status of all RFSNs, stores all metadata related to recordings in a central database, and can take in a schedule of recordings and re-issue it to chosen listeners. *AngularJS* was also used in the website to speed up data entry for schedules and allow rapid development.

Thanks to the modular design of RFSNCC, it remains relatively independent from its actual scheduling mechanism. It could be used to schedule GNU Radio flowgraphs, or could even be used as an easy interface for recording and maintaining data on a single machine. If you would like to use RFSNCC in your own projects or learn more about it, please visit our GitHub at http://bit.ly/2vlgQBQ.

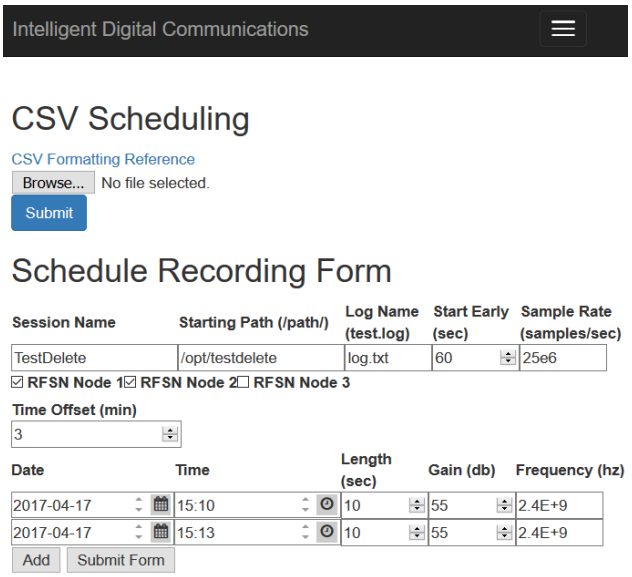


Figure 3. RFSNCC Schedule Recording Page

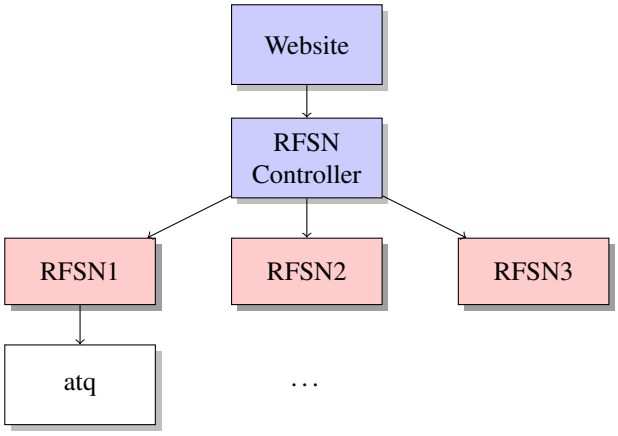


Figure 4. RFSNCC Software Architecture

3. Laboratory Setup and Calibration

Because the equipment is more accessible in the laboratory than in the stadium, it can be used for controlled experimentation and iterative algorithm development. Algorithms are simulated in software such as MATLAB or GNU Radio before they are implemented in the laboratory testbed with known inputs. If the algorithm is successful in the laboratory, it is deployed to the stadium (Garver et al., 2015). To perform localization or other applications, the laboratory can simulate different emitter geometries by changing the cable lengths to the RFSNs instead of physically moving an emitter to different locations in the stadium for testing. A diagram representing the connections between the devices in the laboratory testbed setup is given in Figure 5.

The laboratory setup is illustrated in Figure 6. As the laboratory testbed has a closed RF environment, performing tests on it eliminates the challenges of transmitting a signal through three-dimensional space such as multi-path propagation. Once all preliminary testing has been finished, the software algorithms can be easily deployed to the stadium for real-world testing.

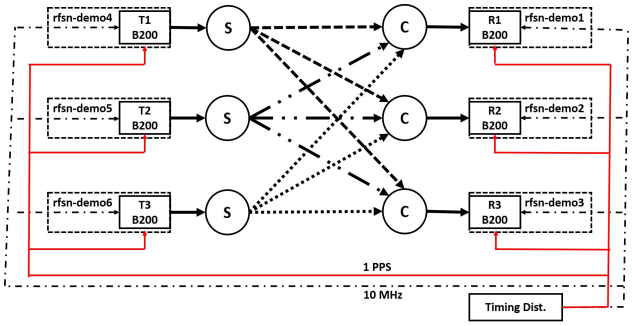
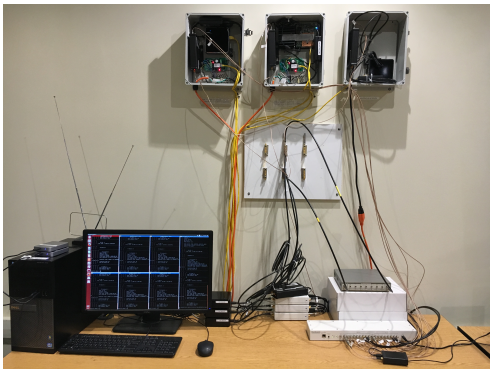


Figure 5. This diagram illustrates the setup of the testbed in the laboratory. T1, T2, and T3 represent transmitting RFSNs which are cabled to splitters, S. The cable lengths, L, from each splitter or combiner, C, to its corresponding RFSN is the same. The cable length between S and C can be varied depending on the desired spatial geometry being simulated. The combiner sums the signals from all transmitters and delivers that signal to receiving RFSNs, R1, R2, and R3. A 10 MHz clock and a 1 PPS reference signal is provided to all nodes by a timing distribution board for time synchronization.

(a) All Three Receiving RFSNs Mounted on the Wall



(b) Complete Laboratory Testbed Setup

Figure 6. These pictures illustrate the current hardware in the laboratory testbed.

The laboratory testbed can simulate a number of arbitrary geometries and interference situations in order to test localization. To perform this testing procedure, the inherent difference in delays in the system between RFSNs must be characterized via a calibration process. The inherent delay in the testbed setup is very small, on the range of tens of nanoseconds. It might be easy to dismiss this inherent delay given it seems really small. However, when performing localization applications on signals that are traveling at nearly the speed of light, the localized position of an emitter is off by approximately a foot for each nanosecond of error. This presents a challenge when applying the localization process in a stadium full of people as an error of even a few feet would make it impossible to locate the emitter.

The calibration process used in the laboratory setup sends a known Psuedo-Noise (PN) signal sequence modulated with Binary Phase Shift Key (BPSK). The pulse shape is selected as a Root-Raised Cosine (RRC) to band-limit the signal. This signal is transmitted from a host computer to receiver RFSNs which sample the signal. These receiver RFSNs were connected to a single transmitter using equal length LMR-240 cables. These samples can be analyzed to determine the delay between when the RFSN received the signal and when the signal was transmitted, allowing for this inherent delay in the system to be eliminated from any other calculations that are performed. To perform this analysis, the received signal is cross-correlated with the original PN BPSK signal. The sample index corresponding to the peak of the cross-correlation function indicates the delay of the signal. However, simply finding the discrete sample of data corresponding to the peak of the cross-correlation is not precise enough for our calibration calculation as samples are 40 nanoseconds apart. By applying an interpolation scheme to the peak sample and the data points near it, a more accurate estimate of the peak of the cross-correlation of the signal in continuous time can be found.

In order to verify the performance of our calibration process we have performed experiments to characterize the inherent system delay. The numerical results of these experiments are in Table 2 and 3. Table 2 contains the Time of Arrival (ToA) results for each RFSN in the laboratory stadium. Table 3 contains the results for the Time Difference of Arrival (TDoA) between each pair of RFSNs. For both of these sets of data a total of four recordings were taken with the results averaged in Table 2 and 3 and graphed in 7. Both sets of data have very low sample standard deviation and variance which is promising. The difference in ToA between RFSNs is the TDoA and it can be used for localization. Calibration implies that the delay of a signal through the testbed is known for each of the nodes, removing that source of error for calculating unknown delays. The PN BPSK signal used is particularly useful for this calibration process because the sidelobes of the autocorrelation are minimized, reducing the probability that noise will cause a false peak in the cross correlation.

Table 2. Calculations on the delays for each RFSN

	RFSN 1	RFSN 2	RFSN 3
Mean (ns)	1.0015	1.0015	1.0015
Variance (ns^2)	2.567E-5	2.069E-5	1.666E-5
Std Dev (ns)	5.067E-3	4.548E-3	4.082E-3
SNR (dB)	79.904	80.279	80.279

Table 3. Calculations on the differences in delays between RFSNs

	Btw 1&2	Btw 1&3	Btw 2&3
Mean (ns)	18.228	26.440	8.213
Variance (ns^2)	4.004E-5	3.478E-5	1.644E-5
Std Dev (ns)	6.327E-3	5.898E-3	4.054E-3
MSE (ns^2)	7.241E-5	3.506E-5	5.508E-5

Previously, we used a Matlab script to process the data and calibrate the RFSNs but we have recently shifted to using GNU Radio. Using GNU Radio to process the samples instead of Matlab confers several advantages. GNU Radio is faster and more efficient because it has a number of built-in functions that are required to process the data such as cross-correlation and locating the maximum value of a vector of data. GNU Radio can use timestamps from the associated metadata of the received signal to perform various time delay estimations including ToA and TDoA. The largest advantage is that GNU Radio can be used on samples in real-time, enabling real-time calibration and localization in a dynamic environment. Real-time localization process is a key goal of our group.

In order to verify the performance of our calibration process we have performed experiments to characterize the inherent system delay. The numerical results of these experiments are in Table 2 and 3. Table 2 contains the ToA results for each RFSN in the laboratory testbed. Table 3 contains the results for the TDoA between each pair of RFSNs. For both of these sets of data a total of four recordings were taken with the results averaged in Table 2 and 3 and graphed in 7. For both sets of data the sample standard deviation and sample variance are very low, which is promising. To confirm the theoretical accuracy of this data, we compared the variance of the TDoA between the RFSNs to the theoretical Cramér-Rao Lower Bound (CRLB).

For any unbiased estimator $\hat{\theta}$, the CRLB is an asymptotic lower bound on the variance of the estimator, $\sigma^2_{\hat{\theta}}$. For our data to be valid, the variance of it should not be less than the CRLB. It can be significantly higher than the CRLB based on hardware constraints but the goal is to get the experimental data to be as close to the CRLB as possible without going below it (Richards, 2005). If it is less than the CRLB that means the data may be biased or not asymptotic. The CRLB calculation for time of arrival applications is given by Equation 1, given $\hat{t}_0$ is the unknown time of arrival at

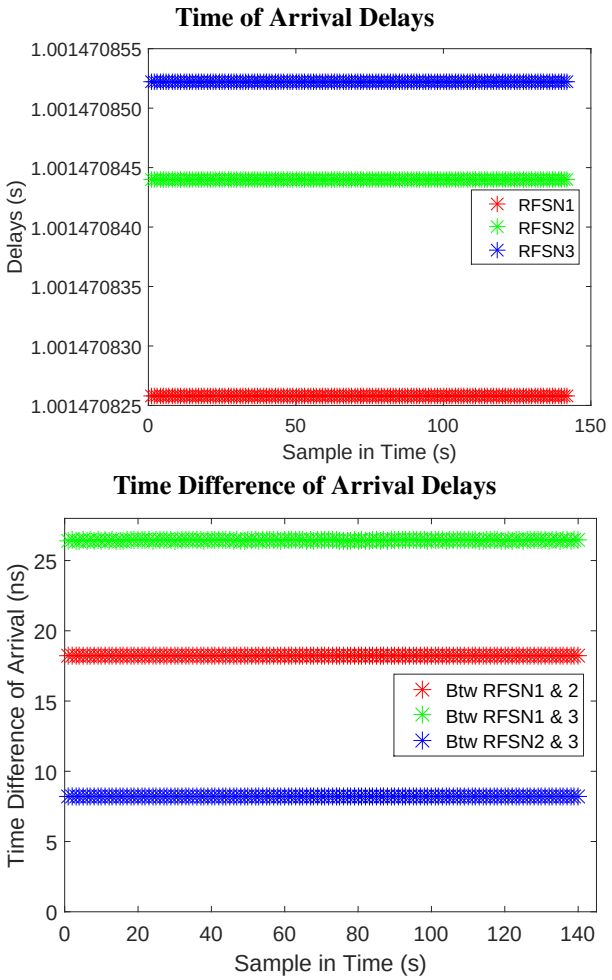


Figure 7. These graphs display the results of the ToA and TDoA tests in the laboratory setup averaged over 4 recordings.

Sensor 0 and $\hat{t}_1$ is the unknown time of arrival at Sensor 1 and so on (Richards, 2005). This CRLB given is for a known complex signal in a complex additive white Gaussian noise.

$$\sigma^2_{\hat{t}_0} \geq \frac{1}{8\pi^2(SNR_{linear})\beta^2_{rms}}, (s^2) \tag{1}$$

The ˆ denotes that the term is an estimate. Given variance is additive when two events are independent, the CRLB of time difference of arrival is given by Equation 2.

$$\sigma^2_{(\hat{t}_0-\hat{t}_1)} \geq \frac{1}{4\pi^2(SNR_{linear})\beta^2_{rms}}, (s^2) \tag{2}$$

Assuming a rectangular spectrum, β_{rms} is given by Equation 3 and SNR_{linear} is given by Equation 4 and 5.

$$\beta_{rms} = \frac{\beta}{\sqrt{12}}\text{ , where } \beta \text{ is the bandwidth} \tag{3}$$

$$SNR_{linear} = 10^{\frac{SNR_{dB}}{10}} \tag{4}$$

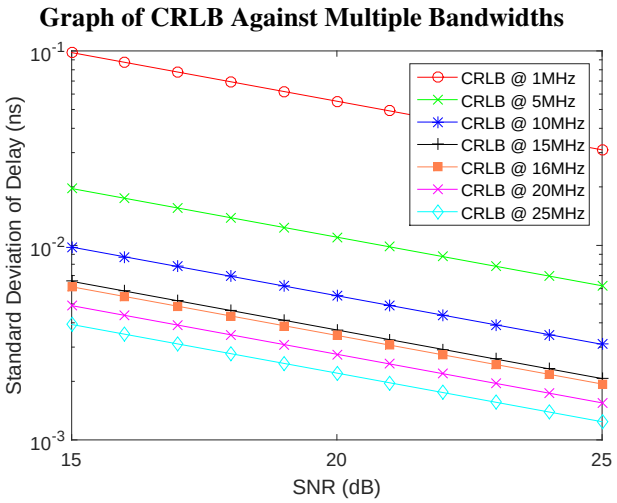


Figure 8. This graph demonstrates the relationship between the CRLB and bandwidth. The CRLB decreases as the bandwidth increases.

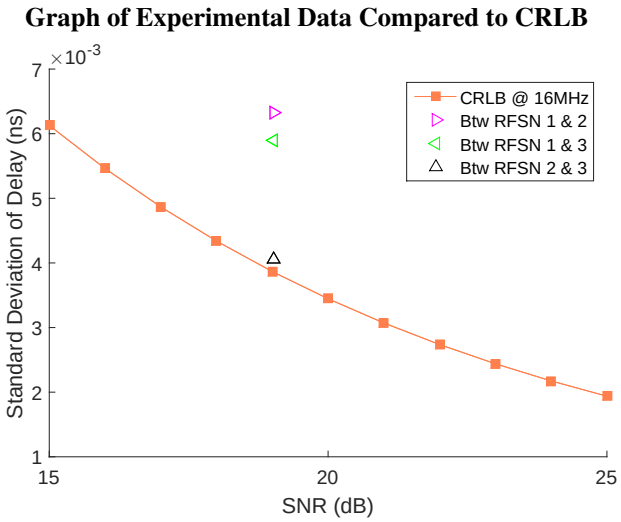


Figure 9. This graph compares the experimental data standard deviation of the TDoA (from Table 3) between the RFSNs to the theoretical CRLB at the specified sampling bandwidth, 16 Megahertz.

$$SNR_{dB} = 10\log_{10}(\frac{\sum_i |X_i|^2}{\sigma^2_{noise}}) \tag{5}$$

Note that this is an energy SNR, which implies the SNR increases for known signals with additional energy. Excluding trivial zero padding, longer known signals will increase the SNR and thus lower the CRLB. Based on the calculated SNR_{dB} given in Table 2, the average SNR in dB across all three RFSNs is approximately 80 dB which corresponds to a linear SNR of 1×10^8 using Equation 4 which was used to plot the TDoA values against the CRLB to ensure that the data is valid (Richards, 2005). We do not expect our results to meet the CRLB given the hardware and the clock resolution. We use the CRLB to ensure that we select a signal bandwidth rate and SNR such that our results are limited by the hardware and not by theory.

The CRLB was plotted at multiple bandwidths in Figure 8 to illustrate how bandwidth affects the CRLB. The time resolution of our plots is limited by the clock rate. Time resolution is based on the sampling frequency and the amount of time data is captured for. Figure 9 is a plot of the standard deviation of the experimental difference in delays between the RFSNs against the CRLB at the bandwidth that the data was captured at, 16 Megahertz with a 32 MHz master clock. Normally, the CRLB is plotted against variance but we chose to plot it against standard deviation here because standard deviation is significant to us as a standard deviation of one nanosecond means the localization could be off by one foot.

By establishing a process that produces results that are close to the CRLB, we know that the precision of our testing is only limited by our hardware, not bandwidth or SNR. These hardware limitations are influenced by our limited budget.

While this experiment confirmed that the results found are theoretically feasible, the next step in verifying the calibration process is to determine how accurate the process is when a known delay is introduced to it. A laboratory setup enabling such testing is depicted in a diagram in Figure 10. This diagram shows a signal being transmitted from idc-dev1 to a splitter which relays the signal to rfsn-demo1 and demo6 before the signal traverses through an LMR-240 cable of various lengths, N, with N representing the length of the cable in feet. This cable serves to delay the signal before it reaches rfsn-demo2, rfsn-demo3, and rfsn-demo4. By calculating the difference in delays from rfsn-demo1 to rfsn-demo2, rfsn-demo3, rfsn-demo4, and rfsn-demo6 we are determining the TDoA of the system and can compare it to the theoretical TDoA to determine the calibration process accuracy. rfsn-demo1 and rfsn-demo6 should have a very small TDoA as they both receive the signal before it traverses the LMR-240 cable.

These tests were performed at a 25 megasamples/sec sampling rate with a 50 MHz master clock. The LMR-240 cable lengths used in our tests were 12 ft, 50 ft, 100 ft, and 200 ft. Four independent recordings were captured for each cable length and were averaged together to produce the results displayed graphically in Figure 11 and numerically in Table 4, where MSE stands for Mean Squared Error. The start-up delay associated with each recording was removed from the TDoA by subtracting the mean of the ToA of the first 100 values from each ToA data set, in

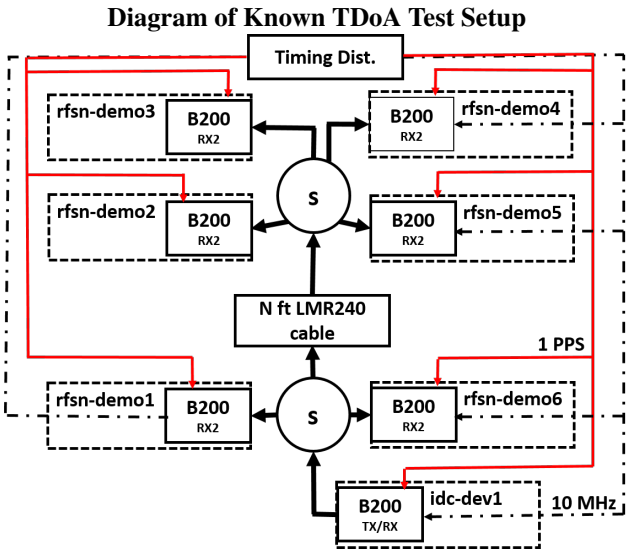


Figure 10. The setup depicted above was used to verify the accuracy of the calibration process given a known TDoA is provided in the system. Circles with the letter "s" are RF splitters.

each recording session. This start-up delay occurs because each USRP takes a certain, random amount of time to tune to the recording frequency for each recording session on each node. By averaging and removing the mean of the first 100, non-delayed ToA values we also minimized any error from any difference in the distribution of the master clock. The known length LMR-240 cable was manually added, between the splitters, about 110 seconds into each recording session. By averaging and removing these first 100 samples we also minimized any error from any difference in the distribution of the master clock. TDoAs that fell outside two standard deviation were removed from the data. During this test, all RFSNs were scheduled to record after a fixed number of seconds after the test was started.

To determine the accuracy of these results, we have to calculate the theoretical TDoA through the cable. Given the LMR-240 cable has a velocity of propagation of 84%, the speed of the signal through the medium should be 0.8268 ft/ns. Therefore, we calculate that the theoretical TDoA for the 12 ft cable should be 14.514 ns. We can also perform a similar calculation for the 50 ft, 100 ft, and 200 ft cables to determine their theoretical TDoAs.

The percent errors from comparing the experimental TDoAs from our tests to the theoretical TDoAs can be found in Table 5. To put these percent errors in perspective, a 9.915% error from one RFSN on an expected TDoA of 14.514 ns means an error of 1.439 ns. In a real-world scenario in the stadium, an error of 1.439 ns means that the distance estimation from each node to the emitter will be off by approximately 1.1 ft in free-space propagation.

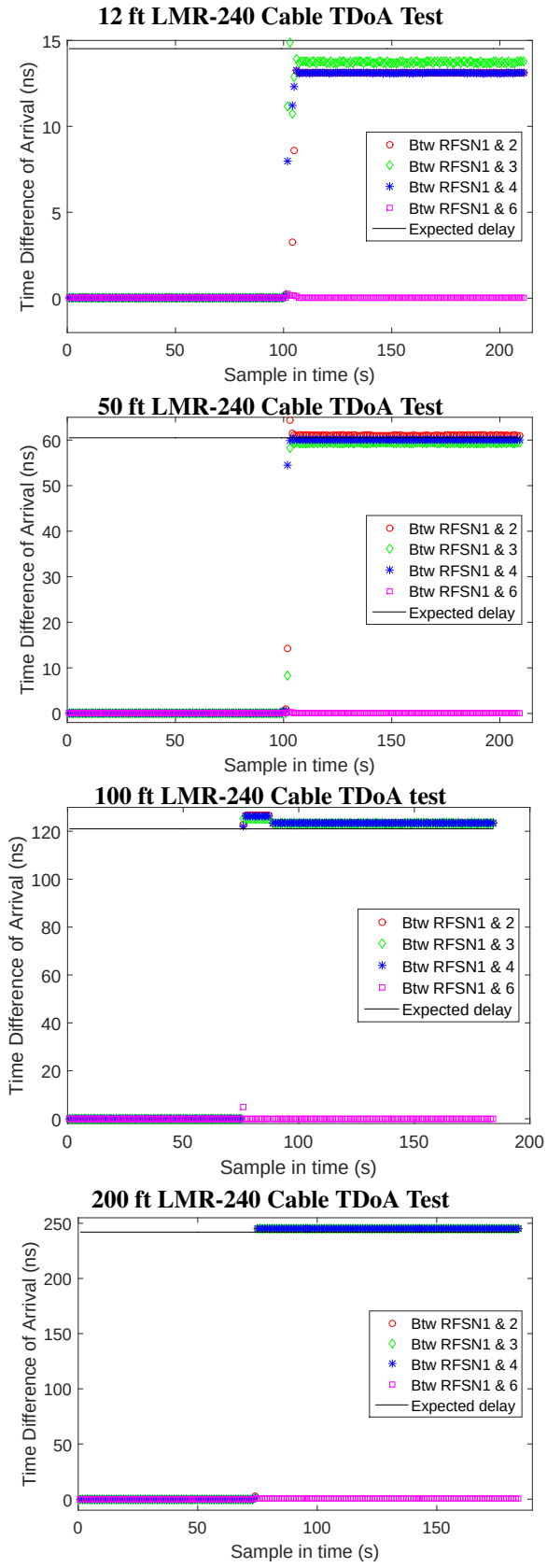


Figure 11. These graphs display the data captured during known TDoA tests. The LMR-240 cable was manually added to the system around 110 seconds for all tests demonstrating how the delay impacted the TDoA in the system.

Table 4. Known TDoA Experimental Results

	Btw 1&2	Btw 1&3	Btw 1&4	Btw 1&6
Mean (ns)	13.075	13.727	13.0137	0.0247
Variance (ns^2)	0.190	0.010	0.007	0.000
Std Dev (ns)	0.436	0.098	0.082	0.018
MSE (ns^2)	2.298	0.645	2.032	0.001

(a) TDoA Calculations for 12ft LMR-240 Cable

	Btw 1&2	Btw 1&3	Btw 1&4	Btw 1&6
Mean (ns)	61.046	59.425	60.108	0.004
Variance (ns^2)	1.946E-04	6.037E-04	3.728E-05	4.154E-06
Std Dev (ns)	0.014	0.025	0.006	0.002
MSE (ns^2)	0.301	1.168	0.155	0.000

(b) TDoA Calculations for 50ft LMR-240 Cable

	Btw 1&2	Btw 1&3	Btw 1&4	Btw 1&6
Mean (ns)	120.357	119.881	120.954	0.188
Variance (ns^2)	5.551E-05	3.546E-04	1.049E-04	3.072E-05
Std Dev (ns)	0.008	0.019	0.010	0.006
MSE (ns^2)	0.417	1.263	0.002	0.036

(c) TDoA Calculations for 100ft LMR-240 Cable

	Btw 1&2	Btw 1&3	Btw 1&4	Btw 1&6
Mean (ns)	244.981	245.010	245.156	0.859
Variance (ns^2)	1.767E-04	2.546E-04	1.296E-05	9.392E-06
Std Dev (ns)	0.013	0.016	0.004	0.003
MSE (ns^2)	8.971	9.148	10.050	0.744

(d) TDoA Calculations for 200ft LMR-240 Cable

Table 5. Known TDoA Accuracy Calculations

Cable Lengths	12 ft	50 ft	100 ft	200 ft
Experimental TDoA (ns)	14.514	60.474	120.948	241.897
% Error Btw 1&2	9.913	0.946	0.489	1.275
% Error Btw 1&3	5.421	1.735	0.882	1.288
% Error Btw 1&4	9.720	0.605	0.005	1.348

An error of more than few feet means it can be difficult to locate a transmitting emitter, especially in a crowded stadium. However, the degree of error of our results correspond to a sufficient accuracy for locating a specific stadium attendee. Our next steps will be to deploy the calibration process in the stadium and see if it has similar levels of accuracy in a real-world situation.

4. Conclusion

Going forward, we will design a robust, real-time process that can be deployed on both our laboratory and stadium testbeds. Our node management system should enable easier analysis of our data, faster development of our localization algorithms, and encourage other testbeds of a similar nature to be built. Future capabilities will enable real-time calibration and TDoA calculations which could then be used in a number of applications including OSI cross-layer localization, frequency coordination, and spectrum utilization analysis.

References

Coyle, J. Allebach and Krueger, J. The vertically-integrated projects (vip) program: Fully integrating undergraduate education and graduate research. *ASEE Annual Conference and Ex-position*, 2006.

Garver, P. W., Abler, R., Coyle, E. J., and Narayan, J. Comparisons of high performance software radios with size, weight, area and power constraints. *ACM WiNTECH Workshop*, 2014.

Garver, P. W., Abler, R., and Coyle, E. J. Theory and development of cross-layer techniques for localization in environments with extreme emitter densities. *Milcom Track 4 - System Perspectives*, 2015.

Garver, P. W., Coyle, E. J., and Abler, R. T. Mac layer assisted localization in wireless environments with multiple sensors and emitters. *WCNC*, 2017.

Lincoln, Orin and Garver, Paul. Tools for high sample rate recording and post processing: gr-analysis. GNU Radio Conference, 2015. URL http://bit.ly/2trTbOl.

Richards, M. (ed.). *Fundamentals of Radar Signal Processing*. McGraw-Hill, New York, NY, 2005.

Schmitz, J. and Hernndez, M. Synchronization in distributed sdr for localization applications. Technical report, FOSDEM, 2016.