Inexpensive Sports RTLS System

Authors: Udit Ranasaria, Shiva Nathan, Rhea Kudtarkar: Electrical and Computer Engineering,
Carnegie Mellon University

***Abstract*—A real-time location system, capable of tracking basketball players on the court during games in a non-invasive and inexpensive manner. Our project will focus on providing an easy-to-deploy system (both for players and court infrastructure), at a total hardware cost of <$1000, with better in data-accuracy than has been previously achieved at this price-point.**

***Index Terms*—Design, RTLS, Sports Analytics, Sports Tracking, TDoA, UWB,**

I. INTRODUCTION

In 2013, the NFL started searching for a system to collect detailed on-field player performance data for advanced trends and analytics. They partnered with Zebra Technologies to build an ultra-wideband (UWB) based Real-Time Location System (RTLS) called Zebra MotionWorks Sport (ZMWS). ZMWS' hardware at each football game comprises of around 250 tags on the players, match footballs, end-zone pylons, match-referees, and even the first-down chains. Each NFL stadium was also retrofitted with powered stationary anchors that communicate with the tags. In 2017, the NFL combined raw ZMWS data with an Amazon Web Services-based cloud analytics software stack to build an end-to-end system analytics and insight system known as Next Gen Stats (NGS) [4]. They publish the data and insights to each team, and a subset is also available to the general public.

The advent of these nickel-sized [6] tags has transformed the century-old league. Coaches, players, and scouts have used NGS to improve every part of their job and game. However, NGS is prohibitively expensive, and requires 20-30 UWB anchors [7] installed per stadium as well as dedicated on-site experts to run the system. This technology is totally inaccessible but just as important to lower level sports, ranging from regular high school games to D2/D3 collegiate sports. The technology can help players get scouted at the next level and help coaches teach the sport through statistics. It can also help league administrators record safety measurements to design better equipment [5].

Our capstone project aims to revolutionize this space by building a similar end-to-end system, with an affordable price and easy field deployment. The RTLS field is quite crowded (Eliko, Sewio, TCI, Pozyx, etc.) but these companies cater solely to enterprises at high price points. Decawave is the only company currently promoting (near) open-source RTLS development, with well documented and accessible UWB-chip kits. However, even their development kit costs $25,000. We will leverage Decawave hardware and resources to build an RTLS system that works for a basketball game in a Wiegand sized gymnasium.

Our minimum goals for the tracking specifications are derived from research and data drawn from NBA Advanced Stats. We will need to support a minimum of 10 concurrent tags for the 10 players on the court at any given time. Our system's maximum range should be 35 meters. We also want to locate each player accurately to 0.35 m (1 ft). To do this, we must collect a player's location at least 7 times per second. The tags should be small so that they do not affect player mobility, and the anchors should be entirely wireless and relatively portable to ensure easier deployment. We aim to limit the hardware costs of our system to around $1000.

II. DESIGN REQUIREMENTS

Based on the requirements derived from Basketball movement, we derived the following technical requirements for our system:

It is common knowledge that basketball defenders stay "at arm's length" (~2 ft) from the attacker. Therefore, we must locate each player accurately to 0.35 m (1 ft) to distinguish between an attacker and defender in a 2 ft radius. Since the average speed of a basketball player is ~2 m (6.5 ft), the tags must have a transmission frequency of at least 7Hz. The radios need a range of 35 meters, as the longest possible distance on an NCAA-standard court is the diagonal at ~32.5 m. Power consumption must be extremely low, so that on-board batteries can remain tiny while staying powered for an entire season. The tags must be small as possible, but this is an optimized constraint rather than a hard requirement (hopefully <5cm^2).

The stationary anchors share the frequency (7 Hz) and range (35 m) requirements. The anchors do not need to be small, since they are on the ground, but should be wireless. They can be wall-powered but ideally will have a battery pack that can last for 3 hours, since NCAA games run for 2 hours and 10 minutes.

Location accuracy directly corresponds to timing inaccuracies and drifts. We calculate our max timing inaccuracy based on our location accuracy requirement:

$$\Delta d / c_{air} = \Delta t \tag{1a}$$

Given a desired max error in distance (Δd) of 0.35m, and the speed of light in air (c_{air}) as 299,702,547m/s, we get the max error in time to be:

$$\Delta t = \frac{0.35m}{299702547\frac{m}{s}} = 1.167ns \tag{1b}$$

The last major requirement is that of backhaul (i.e. transport of data to central location). The total number of messages we must backhaul per second (M) is:

$$M = 10 \text{ tags} * \frac{7\, messages}{second} * 4\, anchors = 280 \frac{messages}{second} \quad (2a)$$

Based on this the minimum bandwidth needed (B) is,

$$B = \frac{100 \text{ bits}}{\text{message}} * 280 \frac{messages}{second} = 0.028 Mbps \quad (2b)$$

This minimum bandwidth is relatively low, but we want to keep in mind scalability and on a larger field requiring 10 anchors, 10 messages/second, 100 tags the calculation in Eq. 2 rises to 1Mbps which we kept in mind as our max needed ceiling.

III. ARCHITECTURAL DESIGN TRADE-OFFS

Our first main architectural design decision was the choice of radio standard/technology for our system. Thankfully, this decision was made relatively simple. RTLSs have been a topic of industry and academic research for some time. An evaluation of Indoor Localization Technologies in 2016 [2] concluded that "after more than a decade of intensive work in this area, the indoor location problem remains unsolved" (best was 0.72m in best-case scenario settings). The tested technologies, WiFi, Bluetooth Low Energy (BLE), IR, Ultrasonic, and RFID, were retrofitted with novel software and signal analysis for accurate localization. Most of these technologies suffer from Line-of-Sight (LoS) obstruction sensitivities, while RFID suffers from a severe range issue [1]. In contrast, Ultra-wideband (UWB) is a recent radio technology that was standardized in 2011 but popularized in systems around 2017. It was designed for precision Time-of-Flight (ToF) calculations and is low-power, low-cost, high-range and high-reliability for localization.

We decided to use UWB and therefore localization via some form of ToF. There are three main types of ToF algorithms that we could implement: Two Way Ranging (TWR), Time Difference of Arrival (TDoA), and Phase Difference of Arrival (PDoA).

TWR is the simplest ToF calculation. Tags send out timestamped pulses and anchors respond to get a round trip time (RTT). Distance (d) from each anchor is calculated similarly to Eq. 1:

$$d = \frac{RTT * c_{air}}{2} \quad (3)$$

These distances must be piped to the central server to calculate exact location using distance from each anchor.

TDoA relies on time deltas to detect relative movement from an anchor rather than absolute RTT. Tags send out 1-way pulses, which are timestamped at the anchor before being passed to the server. The server takes the deltas and absolute time traveled between every anchor to get the final location, in a process called multilateration.

PDoA leverages TWR to get the distance between anchor and tag but also uses the difference in phase of the signal to calculate a bearing. This provides both the tag's magnitude and direction from the anchor, which can be translated to a location with just one anchor.

In our analysis between these three technologies, we quickly moved away from PDoA as the error for even the state-of-the-art implementations was too high at the distances we needed. Picking between TWR and TDoA was more challenging. TWR offered development simplicity at the expense of extra communication. TDoA required more complicated location algorithms and complex anchor design, but reducing the number of messages sent. The tags in a TDoA system would consume much less power and could be made much smaller. TDoA tags available on the market fit our size requirements with multi-year battery life. Because TDoA fit our requirements much better, we chose to implement it over TWR.

IV. ARCHITECTURE AND/OR PRINCIPLE OF OPERATION

We are designing a TDoA system for a basketball court. This means we need mobile tags and stationary anchors. The operation of the system (Figure 1) is as follows.

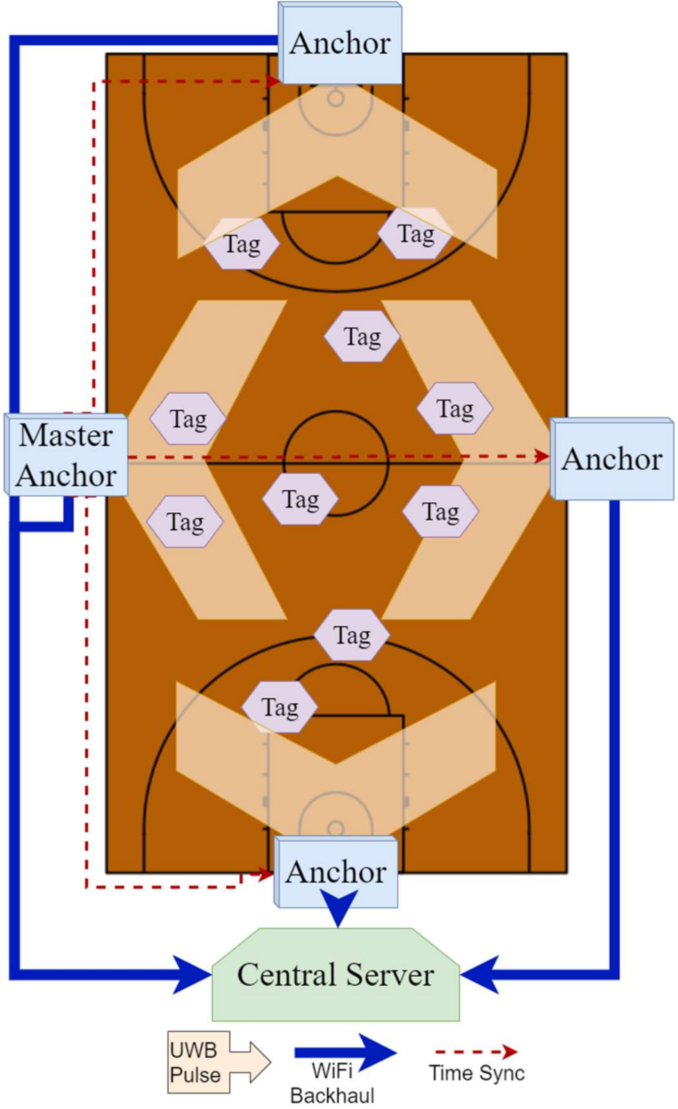


Fig. 1. Architectural diagram of system without technical detail.

At a determined rate, every tag (located on the players body) will emit a UWB pulse with its respective ID. The anchors, which are stationary around the court, will receive these IDs from each tag. Each anchor will add a timestamp and its own ID to each message it receives, then backhaul it to the central server. The central server will use these messages from all anchors to calculate a location.

This algorithmic approach means that we need each anchor to be synced to a common clock to the degree of accuracy determined in Eq. 1 (~1.2ns). We will need to determine a master anchor (whose clock will be the reference) and have each slave anchor sync to the master anchor often to achieve our low time sync error requirement.

V. SUBSYSTEM DESIGN DECISIONS

A. *Tags*

There are very few open-source and self-programmable UWB chips on the market. Decawave is one of the only companies offering enthusiasts and individuals who want to experiment with their technology. Their DWM1001C chips are sold for TWR use and their DWM1004C tag-only chips for TDoA use. The DWM1001s are sold in a dev-kit attached to breakout boards.

The DWM1001 has all the hardware of the 1004C with extra functionality (e.g. Bluetooth capability and more powerful MCU), since it was designed to serve as a tag, anchor, and gateway in both TWR and TDoA applications. Since we pre-owned the kit with the DWM1001s, we are using that as our development tags. We may upgrade to the 1004Cs later in the project's lifetime.

Both the DWM1001 and 1004C are equipped with 3-axis accelerometers that can dynamically adjust the pulse rate based on detected acceleration of the player wearing the tag. However, we decided that implementing these was beyond the scope of the project.

B. *Anchor Design*

Most of our design research went into the anchor design. Anchors are the pinnacle of the system and have the most requirements. They need to receive UWB, backhaul via some other radio, and keep accurate time-sync; all while remaining mobile and cheap. The UWB problem was the easiest to solve. We would use the DWM1001s we already had since they had the appropriate radios in a friendly, communicative form-factor.

We researched many solutions for the backhaul requirement. The first was using UWB itself for backhaul. We dismissed this idea due to high risk of crowding the spectrum and causing interference. We considered BluetoothThe MCU in DWM1001 was BLE equipped and we considered using this as well. But after researching into BLE range and bandwidth they were weaker compared to Wi-Fi. Since our application was indoor basketball, we realized that using Wi-Fi over a connected RPi would offer an easy and quick solution for backhaul. TCP ensures backup and connection via IP is quick and easy.

Time-synchronization remains the hardest part of our design. After extensive research, we found that nano-second level sync is clearly no joke. Our state-of-the-art research found that almost everyone seems to use a 30ish KHz TCXO as a reference clock on each anchor. We believe that this design might end up becoming necessary but want to approach with a simpler approach at first. Just use the C reference library *time.h* monotonic clock values. The clock in a RPi4 is 1.5Ghz so should technically be ticking fast enough.

Since we plan to timestamp on the RPi, we should just be able to use Wi-Fi to send time-sync messages using the UDP protocol (fast delivery is more important than guaranteed delivery) between the RPis at each anchor. We still need to look into detailed time-sync protocols but we plan to use the PTP method.

If this simplified timekeeping and sync approach doesn't work, we have a various backups. Part of the problem could be the fact that the UWB pulse gets received in the DW1000 radio and passes through 2 SPI radios before is time-stamped which could add a lot of just internal variance between anchors. In this case we would have to push the timestamping closer (get it on the Nordic MCU). If we believe the timestamps drift too much, we will have to leverage TCXO references at each anchor via the RPi (or maybe a Particle Board). Processing these TCXO signals will take a Kalman filter amongst other things. If this also fails, we might have to hardwire a reference clock with the same length of wire to each anchor as done in this Spring 2018 capstone [8].

C. *Central Server Design*

The only requirements for the central server are that it be able to receive backhauled messages and be powerful enough to run our multilateration algorithms. For this, any Wi-Fi connected laptop should do.

We considered a cloud approach here, but we deemed it as a stretch goal. If we run into computation limits with the volume of data we are receiving, we can stream the data to the cloud, and have it process, and report back the data.

In a complete system, we would most definitely leverage the cloud to run a ML-stack to process all the location data and provide insights as well as a way to share the data effectively between coaches and teams.

VI. SYSTEM DESCRIPTION

All references to Fig. 2.

A. *Tag Design*

Our tags will be entirely off-the-shelf. There is little reason to pursue time in chip design when there is the commercially available Decawave DWM1004C. It is smaller and lighter than our requirements with a multi-year battery life. It uses the same DW1000 UWB Radio as underlying technology and is even equipped with an optional accelerometer unit that we can use to optimize our blink rate as a stretch goal.

B. *Anchor Design*

The UWB component of the anchor is handled by a Decawave DWM1001-Dev chip. It has a DWM1000 UWB Radio on board which will receive the UWB pulses and transmit it to an onboard Nordic nRF52832 MCU via an internal SPI connection and interface.

The DWM1001-Dev board will be connected to a RaspberryPi4 (RPi) via a ribbon cable. The Nordic MCU will communicate via SPI to the RPi. The RaspberryPi4's primary function is to backhaul timestamped and ID'd pulses over Wi-Fi to the central server using a traditional sockets API (TCP protocol).

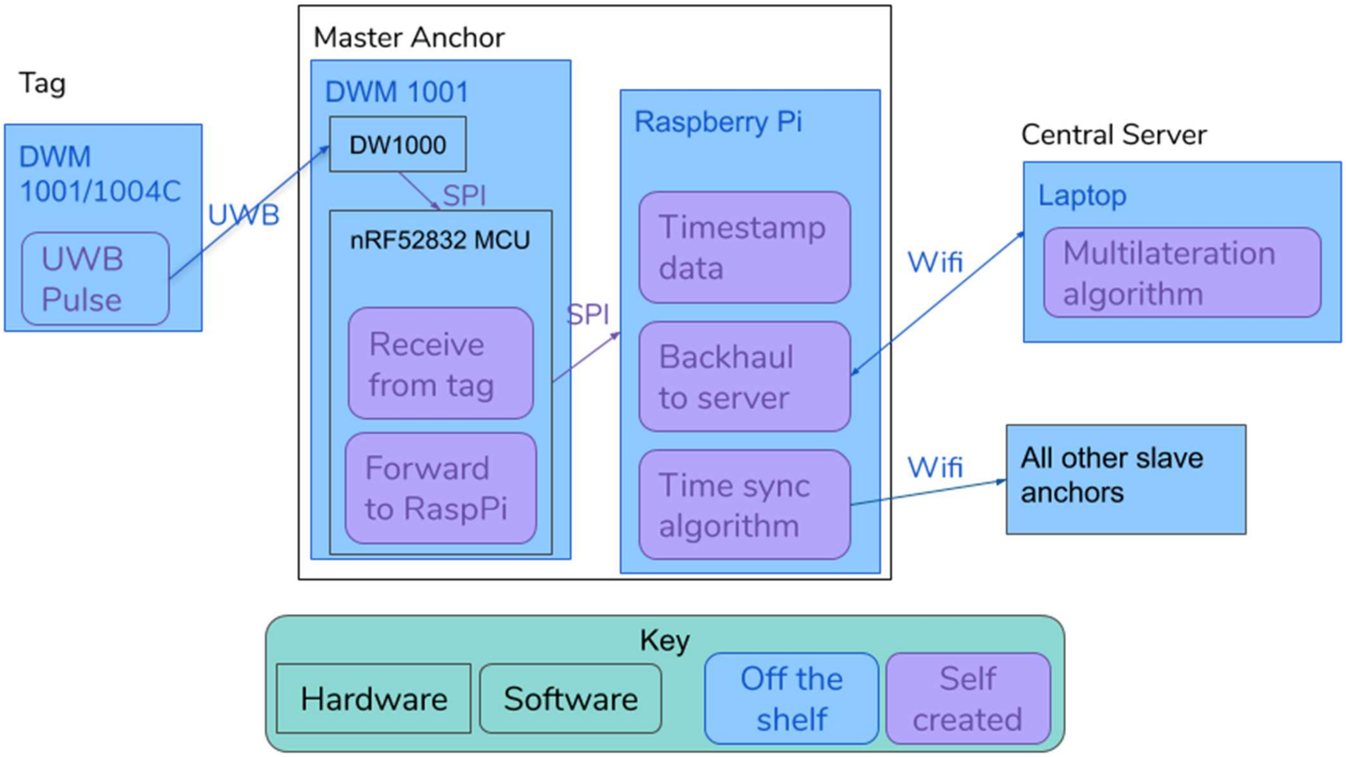


Fig. 2. System Block Diagram

C. Timestamping

By far the most unsteady part of the design. As of right now, we are pursuing the design that will be the fastest to test and so we can start lab testing and making improvements from there. We will likely try **all** of the time-syncing design approaches outlined in the trade-offs section until we can achieve something that will fit our requirements.

Our current design involves leveraging the monotonic "performance counter" clock available in the C time library. This has nanosecond resolution. We will pipe received IDs from the radio to the Nordic MCU to the RPi. As soon as the RPi receives the message, it will timestamp it. Then this will get sent for backhaul.

We will sync RPis over Wi-fi in between every UWB message. So, after the Master-RPi backhauls a timestamped message, it will immediately start a time-sync cycle and finish before the next UWB pulse comes in. We will use the Precision Time-Protocol (PTP) time sync protocol as it has been shown to work at a nanosecond level for LAN settings.

We do believe this simple approach has low chance of success. This is mainly since the physics and reliability question marks don't seem promising. We have many backup options such as adding a TCXO to our RPis to give us a more accurate reference. More options are discussed in part IV.

D. Central Server Design

The central server will be a simple laptop running a locally hosted server for incoming messages. These messages will be stored and stamped with the anchor it was received from. Using the differences in times you can try to find the point intersection of multiple spheres (multilateration) as described in this paper [3]. We will be writing a python program to compute these locations and output them rudimentarily onto a court.

VII. PROJECT MANAGEMENT

A. Schedule

Our original schedule was heavily parallelized with a lot of work in each of the hardest design areas. Problem was that we stagnated making progress on the basic chip programming and setting up the basic hardware system before we could even tackle the tough software components.

So, we changed the plan significantly with a focus on ensuring we can setup a basic system with hardware interfaces working and basic software uploading to Decawave chips before moving into all our custom work.

Our focus after the basic proof-of-concepts are done we are back to parallelization and making progress on the tough time-sync, localization algorithms, and improving signal reliability.

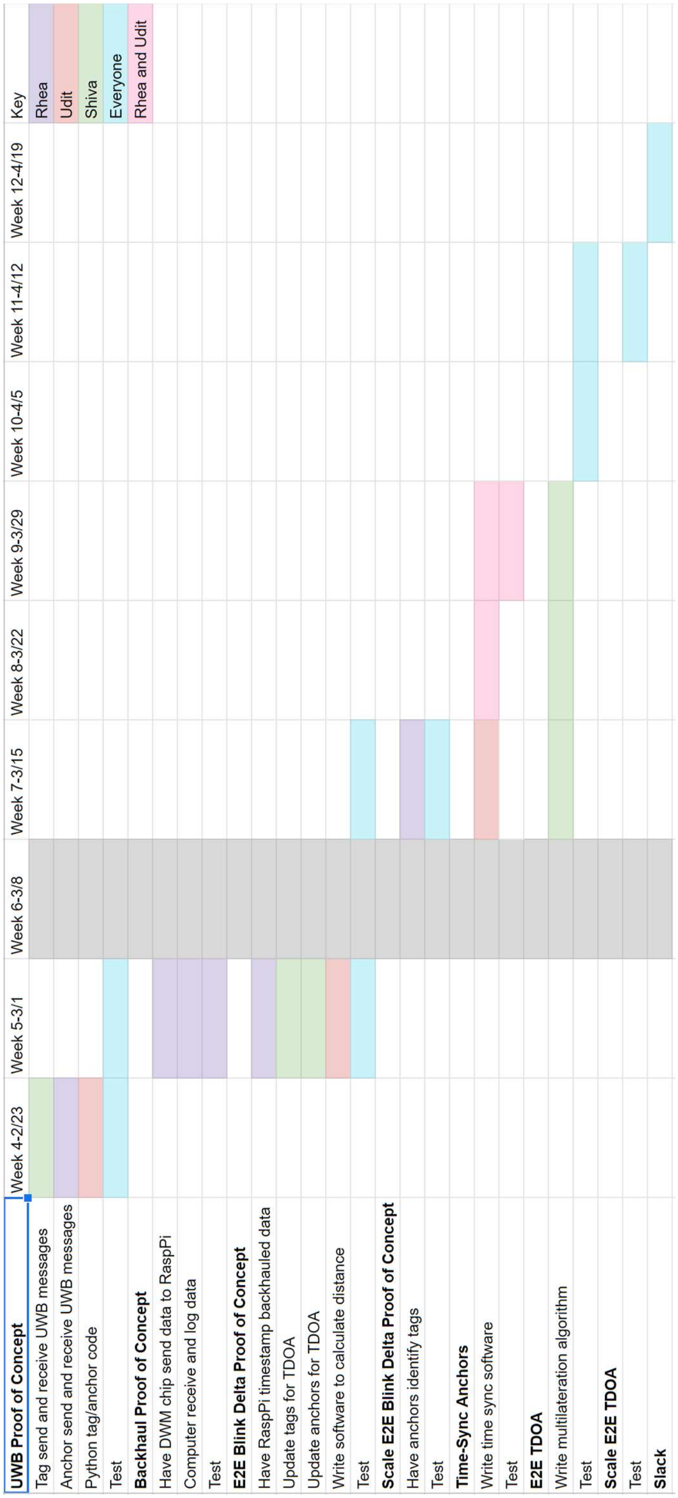
Our Gantt Chart is in Figure 3.

B. Team Member Responsibilities

Rhea is strong in signals and systems as well as embedded programming. So her focus will be writing C programs for passing data from the DWM1001 in the anchor via SPI to the RPi. If we need to do signal processing of a TCXO, she is also our signal processing expert.

Shiva is strong in many areas and contributed in helping out with the C programming taking lead on programming the TDoA anchor reception. His research skills are very strong since he compared all our options for backhaul incredibly well. He will be working on our multilateration algorithm.

Udit has been the design lead planning out the whole system and developing a lot of the requirements based on the target application. He is also familiar with distributed programming and will be handling the time-synchronization protocols and implementation. He will also aid as secondary for research and understanding the math behind multilateration algos.

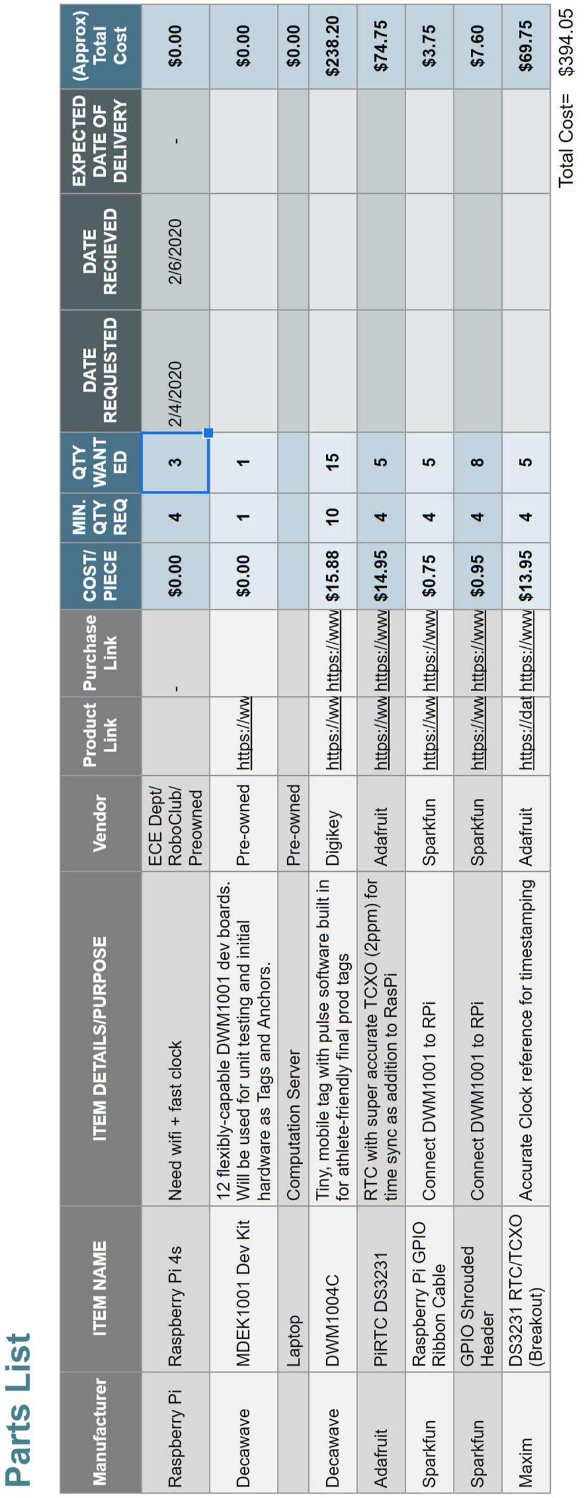


UWB Proof of Concept	Week 4-2/23	Week 5-3/1	Week 6-3/8	Week 7-3/15	Week 8-3/22	Week 9-3/29	Week 10-4/5	Week 11-4/12	Week 12-4/19	Key
Tag send and receive UWB messages	🟩		▨							🟪 Rhea
Anchor send and receive UWB messages	🟪		▨							🟥 Udit
Python tag/anchor code	🟥		▨							🟩 Shiva
Test	🟦	🟦	▨							🟦 Everyone
Backhaul Proof of Concept			▨							🟧 Rhea and Udit
Have DWM chip send data to RaspPi		🟪	▨							
Computer receive and log data		🟪	▨							
Test		🟪	▨							
E2E Blink Delta Proof of Concept			▨							
Have RaspPi timestamp backhauled data		🟪	▨							
Update tags for TDOA		🟩	▨							
Update anchors for TDOA		🟩	▨							
Write software to calculate distance		🟥	▨							
Test		🟦	▨	🟦						
Scale E2E Blink Delta Proof of Concept			▨							
Have anchors identify tags			▨	🟪						
Test			▨	🟦						
Time-Sync Anchors			▨							
Write time sync software			▨	🟥	🟧	🟧				
Test			▨			🟧				
E2E TDOA			▨							
Write multilateration algorithm			▨	🟩	🟩	🟩				
Test			▨				🟦	🟦		
Scale E2E TDOA			▨							
Test			▨					🟦		
Slack			▨						🟦	

Fig. 3. Gantt Chart

C. Budget

Our budget has been surprisingly low because RPis and DWM1001s were already owned. Because of this the main cost to our project would be buying 10 of the DWM1004C tags which are not even needed till after we have a strong demo working with the DWM1001's programmed to be tags. They only offer us a smaller package realization of fitting our original requirements. Our full parts list is in Fig. 4 but keep in mind the TCXO/RTC module might not be needed.



Parts List

Manufacturer	ITEM NAME	ITEM DETAILS/PURPOSE	Vendor	Product Link	Purchase Link	COST/ PIECE	MIN. QTY REQ	QTY WANTED	DATE REQUESTED	DATE RECIEVED	EXPECTED DATE OF DELIVERY	(Approx) Total Cost
Raspberry Pi	Raspberry Pi 4s	Need wifi + fast clock	ECE Dept/ RoboClub/ Preowned		-	$0.00	4	3	2/4/2020	2/6/2020	-	$0.00
Decawave	MDEK1001 Dev Kit	12 flexibly-capable DWM1001 dev boards. Will be used for unit testing and initial hardware as Tags and Anchors.	Pre-owned	https://ww		$0.00	1	1				$0.00
	Laptop	Computation Server	Pre-owned									$0.00
Decawave	DWM1004C	Tiny, mobile tag with pulse software built in for athlete-friendly final prod tags	Digikey	https://ww	https://www	$15.88	10	15				$238.20
Adafruit	PiRTC DS3231	RTC with super accurate TCXO (2ppm) for time sync as addition to RasPi	Adafruit	https://ww	https://www	$14.95	4	5				$74.75
Sparkfun	Raspberry Pi GPIO Ribbon Cable	Connect DWM1001 to RPi	Sparkfun	https://ww	https://www	$0.75	4	5				$3.75
Sparkfun	GPIO Shrouded Header	Connect DWM1001 to RPi	Sparkfun	https://ww	https://www	$0.95	4	8				$7.60
Maxim	DS3231 RTC/TCXO (Breakout)	Accurate Clock reference for timestamping	Adafruit	https://dat	https://www	$13.95	4	5				$69.75
											Total Cost=	$394.05

Fig. 4. Parts List and BoM

D. *Risk Management*

Project Risk was something that we tried to handle consistently throughout the design process. We looked at many different options and fallbacks for almost every part of the project.

The first major risk we had was inability to understand or program Decawave's embedded chip technologies enough to program it with our custom software. We had one person handling this portion of the project in the early Gantt chart but it was quickly clear that that portion of the project was stagnating significantly. We mitigated the risk that was building by adding another person to work on that part and unclogged it so then the parallelization and demos could continue.

Time-synchronization posed a major risk because it was impossible to truly know if a scheme would work without trying and testing it. To combat this, we redid our plan for time-synchronization by looking to build simple systems that we could test while keeping in mind that we will have to add more hardware and signal processing to get this to work well.

The last major risk that exists with our project is the fact that this business is heavily reliant on real-world performance with many, many factors. Even though we feel like we designed out the project well, with a strong grasp of the theoretical math and design that goes into each component, executing and expecting the components to work to a level that would let the theory arrive at a useful system is still a high risk. The margin's for error are slim for many of these parts and that risk still remains relatively high. The only main option we see to mitigate this risk is to revert back to a TWR system but add extra scope to the project with Analytic processing of the location information once we get it.

VIII. RELATED WORK

There are many products out there that achieve what we are trying to do to a higher degree of accuracy. There are countless companies who advertise TDoA systems with sub-decimeter accuracy that can support hundreds-thousands of tags and can scale to infinite number of anchors. The commonalities amongst these products remain their obstructively high price tags and their anchor systems are immobile. While there exist competitors for the sports marketplace, they all seem to use Ethernet backhaul requiring complex stadium installation which limits the applications for lower-league sports teams that share fields and have limited bandwidth for installation and on-site experts.

The obstructive price for a lot of these systems do seem to be mostly in the software complexity. Decawave sells their TWR kits for $300 with 12 chips while their TDoA kit costs $25,000 for 20 chips and 10 anchors. The anchor design should only be a 50% increase in hardware cost at worst, so we have to assume that the 100x blow-up in cost is significantly attributed to software licensing.

IX. SUMMARY

We do believe that our project still has an incredibly important application area as this is an active area of industry work and we are trying to make it more accessible to the masses. That said, we are trying to simplify an incredibly complex system that in some ways might beyond our understanding and expertise.

We have a solid start and are definitely going to run into many challenges ahead, but we hope that we will make it to some form of demo-able project. Our first demo goal will be hopefully outputting messages in our central server that for a single stationary tag show some level of consistency. From there we can work on our location algorithms and perfecting the time sync further at scale for a final demo.

REFERENCES

[1] Decawave Technology. https://www.decawave.com/technology1/

[2] D. Lymberopoulos, et al., "A Realistic Evaluation and Comparison of Indoor Location Technologies." Seattle, 2016 [Online]

[3] G. Shen, R. Zetik, and R, Thoma, "Performance comparison of TOA and TDOA based location estimation algorithms in LOS environment." *5th Workshop on Positioning, Navigation and Communication,* Hannover, Germany, 2008

[4] "NFL Next Gen Stats" [Online]

[5] M. Long, "How the NFL is Using IoT". *Electropages,* 2019, [Online]

[6] J. Pollard, "Five Years Ago, a Nickel-sized RFID Sensor Changed the NFL Forever". *Zebra*, 2019. [Online]

[7] D. Kravets, "How the NFL—not the NSA—is impacting data gathering well beyond the gridiron". *ArsTechnica*, 2019. [Online]

[8] E. Tseng, S. Verma, R. Willison, "Indoor Localization with UWB TDoA". *18-500 CMU ECE Capstones S18,* 2018.