Smartphone Tracking App for Microsoft HoloLens

Project Plan

Team Number
27
Client
Optical Operations
Adviser
Dr. Qiao

Team Members/Roles
Travis Harbaugh/HoloLens Development
Ben Holmes/Android Development
Anthony House/Website Development/Security
Cory Johannes/Report Management
Jose Lopez/Website Development
Ryan Quigley/Android Development

Team Email
sdmay19-27@iastate.edu
Team Website
http://sdmay19-27.sd.eve.iastate.edu/team.html
Revised
October 30th, Version 1

List of Figures

List of Tables

List of Symbols

Not Applicable

List of Definitions

Not Applicable

1 Introductory Material

1.1 Acknowledgement

Sdmay19-27 would like to thank Andrew Guillemette and his engineers from Optical Operations for their contribution to our project. Andrew contributed significant assistance by providing our team with technical advice and resources.

1.2 Problem Statement

In the construction industry there is no form of tracking personnel and machinery to help monitor what goes on in a specific zone. GPS accuracy is typically within 3 meters, but it doesn't always work indoors due to interference with steel, concrete, and large objects (skyscrapers). Many large tech corporations have tried to improve indoor navigation but have been unable to find a scalable solution.
The purpose of our project is to make an application that can track an individual indoors within 1 meter. If we are successful, our client can utilize our research to create a prototype for a real-time construction site tracking solution.

Our goal is to create a mobile application which tracks construction workers and estimates their current location based on Android phone sensor data. Our solution must be accurate to within 1 meter. To achieve our goal, we will assume that the GPS signal on the cell phone may not be available. Therefore, we must rely on secondary phone sensors to determine the users' position. The data obtained will be stored for monitoring via the Microsoft HoloLens.

1.3 Operating Environment

Our operating environment is going to be for a construction site. Our end product will be using a mobile application for gathering a user's location without the use of GPS. We will be using a combination of the phone's sensors and Bluetooth beacons in order to accomplish this. Due to extreme weather and dangerous conditions our device must stay secured in a specific location and orientation on the user's body. A suitable position will not impede the workers ability to perform on the job site.

The user must wear this device at all times while in the designated work zone. While operating, our application will collect data from the phone's sensors, and use heading/distance prediction estimates to map the user's trajectory into a 3D representation of the work zone. In addition, this application will be able to store current location estimates on the Android phone's file system when an established internet connection is not available. As a final requirement, the application will limit battery usage when possible, to achieve battery life durations of 8 hours or more.

1.4 Intended Users and Intended Uses

Our intended user base is construction workers. Our smartphone application will track its users using a combination of the Android phone sensors, bluetooth, and sound. Doing so will provide a new level of supervision over construction projects. It is our hope that this additional level of supervision can be useful in solving logistics issues. It will also provide a record of daily work, which will detail the productivity and current progress of the construction project.

1.5 Assumptions and Limitations

We assume that all equipped phones have the following hardware capabilities:

1. Magnetometer
2. Gyroscope
3. Accelerometer
4. Bluetooth 4.0
5. Wifi chip
6. Speaker & Microphone

We also assume that the audience will be either walking or running when wearing the device. Our proposed solution is not meant to track people riding in vehicles.

To begin with, in order to ensure the accuracy of our tracking algorithm, we will require that users hold their phone vertically in front of their bodies so that the top of the phone points in the direction of motion. Eventually, as our methods improve, we will require that users wear the phone on an armband strapped to their right arm, again oriented so that the top of the phone points in the direction of motion. We are also limiting the environment to Durham. Aside from this, the application is expected to work regardless of a user's trajectory, or type of movement.

1.6 Expected End Product and Other Deliverables

The end product will be an Android application with the following seven functional requirements:

1. Will track a user within an accuracy of 1 meter
2. Will run on all Android phones which have API 21 (Lollipop) or higher
3. Will use the phone's Magnetometer and Gyroscope to determine the direction of motion
4. Will use phone's accelerometer to measure movement
5. Will use the Bluetooth sensor to detect re-calibration points and reset the user's location
6. Will use RSSI values obtained from WiFi access points to infer location
7. Will include a web interface which maps the phone's movement into a 2D and 3D representation of Durham

2 Proposed Approach and Statement of Work

2.1 Objective of the Task

The objective is to create an Android application capable of monitoring user movement on a construction site. This app will use the various sensors available through the phone in order to predict the user's trajectory. It will then relay this information to a server, where it can be translated and used for a 3D visual representation through the HoloLens. The application should be able to store the tracking data locally on the phone's storage in the event that the device is disconnected from the network, or is unable to upload for a period of time.

2.2 Functional Requirements

Requirement	Title
Indoor/Outdoor User Tracking	The software shall track 1 individual walking an unguided trajectory both indoor and outdoors (proposed test site is in and around Durham).
Movement Sensitivity	The software shall detect users moving at basic walking and running speeds.
Store Trajectory Information in a Rethink Database	The software shall send tracking information from the Android device, to the Rethink database through a WiFi connection.
Distance Accuracy	The software shall track the locations of its users to an accuracy of ±1 meter.
Delay Accuracy	A delay time of no more than 5 seconds will exist between the time when data is collected, and the time when data is sent to the database.

HoloLens	Our solution will use the location data stored in our Rethink database for monitoring movement through the HoloLens.

Table 1. Functional Requirements

2.3 Constraints Considerations

2.3.1 NON-FUNCTIONAL REQUIREMENTS

Battery Life Cycle	The Android application shall run in a background service, and last for an entire work day.
GPS Sensor	The tracking algorithm shall work without the use of GPS.
Look and Feel	The tracking device shall be comfortable enough to be worn for the entire work day.
Environmental	HoloLens simulation shall be used in indoor conference rooms

Table 2. Non-Functional Requirements

2.3.2 CONSTRAINTS

The phone shall be strapped to the right arm in a horizontal position with the top of the phone facing in the direction of motion.
The application must be turned on before going inside a building.
There shall be no movement until the application has been initialized and calibrated to the user's current position.
The cellphone shall be an Android device with Accelerometer, Magnetometer, Gyroscope, and Bluetooth sensors.
The Android phone shall use API 21 or above.
A WiFi connection will not always be available.

Table 3. Constraints

2.4 Previous Work And Literature

The purpose of this section is to present a few of the basic concepts we have considered for this project.

Our project aims to track individuals more accurately than current tracking systems utilizing GPS alone. Our client's last team implemented a Raspberry Pie which used RSSI triangulation (WiFi), and turned that information into latitude and longitude coordinates. Their solution suffered from floating point inaccuracy when converting RSSI based distance estimates to latitude/longitude coordinates. We aim for a solution implementing a Dead Reckoning system with Bluetooth/LE calibration for more accurate results.

Truong Et Al.

The first piece of literature that was read refers to Truong et al, an indoor tracking solution that utilizes insole sensors. In order to track a location this literature proposes using insole sensors which can estimate walking distance by summing up the total number of steps detected, and multiplying by a designated stride length. They used accelerometer and pressure sensors to record each movement. They then transmitted the data to a cellphone which filters out the error and records the distance that the user has walked. The advantage of this approach is more accurate step detection because of the placement of the sensors (in the shoe). The cons of this approach are large error accumulations when distances exceed ~80 meters due to the inconsistencies in stride lengths.

Gallagher Et Al.

The second piece of literature refers to Gallagher et al, a human posture tracking solution utilizing the accelerometer, gyroscope and magnetometer sensors of 3 different phones oriented in a fixed position around a user's waist, equidistant from the bodies center. The basic premise is that since all devices experience acceleration relative to a center of motion, the readings of the 3 different phones can be combined in order to obtain more accurate results. They found the change in posture by integrating the angular velocities read from the gyroscopes, and adding that integration result to a previous posture estimate. This technique has a big advantage over magnetometer based orientation tracking solutions
because the gyroscope is not as affected by stagnant electromagnetic radiation in the environment, whereas the presence of a strong magnetic field will greatly disrupt magnetometer readings.

Park, Kwanghyo Et Al.

The third piece of literature refers to Park, Kwanghyo et al, a pedestrian tracking solution that uses a combination of the accelerometer, magnetometer and step detection in order to determine if a user has made a 90 degree turn. The literature focuses on having a map and using the

dimensions of a building to determine where a user is based on where they have been. Their solution uses machine learning to determine if a user turns down some specific hallway, or corridor. The disadvantage of this approach is that implementing a machine learning algorithm would be time consuming, and CPU intensive on an Android phone.

A. R. Jiménez Et Al.

The fourth piece of literature I would like to discuss is A. R. Jiménez et al, and their implementation of LE/UWB radio waves for distance estimation and triangulation in a narrow hallway. The literature claims similarly high levels of accuracy in distance estimation for both LE and UWB through a simple Path Loss formula.

RSS is the received power in decibels, RSS0 is a mean RSS value obtained at the reference distance d0=1m, d is the distance between emitter and receiver, p is the path loss exponent, and v is a Gaussian random variable with zero mean and standard deviation σRSS that accounts for the random effect of shadowing (A.R. Jiménez et al).

Our Plan

We will be using a combination of the techniques reviewed above. We will be implementing a Pedestrian Dead Reckoning system combined with RSSI calibration. We will use the following ideas from the sources listed above:

Similar to Gallagher, we will use the gyroscope data for orientation estimation as opposed to the magnetometer. However, we will manage our rotations by using Quaternions instead of Euler rotations. Quaternions are more appropriate than Euler rotations, because a Quaternion system does not suffer from Gimbal lock, which is a state when two rotation axis' become aligned and cause the rotation matrices to produce unexpected results.

Similar to Truong et al, and Park et al, we will implement a form of step tracking and distance estimation. Distances will be estimated by summing up the amount of steps detected and multiplying by a step length. While we know this is prone to large amounts of error accumulation, as Truong et al discusses, we are hopeful that the use of Bluetooth and WiFi RSSI values can serve a certain degree of error correction.

This combination of heading and distance prediction provides our basic PDR system. It is essentially the same concept as an Inertial Navigation System, only our two looming issues include how to accurately classify steps as opposed to random movement, and how to accurately estimate distance based on fluctuating step lengths.

Finally, similar to Jiménez et al, we will utilize RSSI values obtained from Bluetooth 4.0 beacons, fed through a Signal Propagation formula to estimate relative distances. Our solution will be slightly different from what Jiménez et al describes, but will still use the Signal Propagation formula. We will instead create a Radio map, representing a collection of RSSI readings in

different areas of Durham. During the tracking phase, we will compare current RSSI readings to our map in order to determine which quadrant of the room a user is most likely to be in.

It is our hope that in the future we will be able to incorporate ultrasonic frequencies into our localization methods. One possibility is measuring sound waves in the area to determine the relative proximity of a user in relation to a sound emitting source.

2.5 Proposed Design

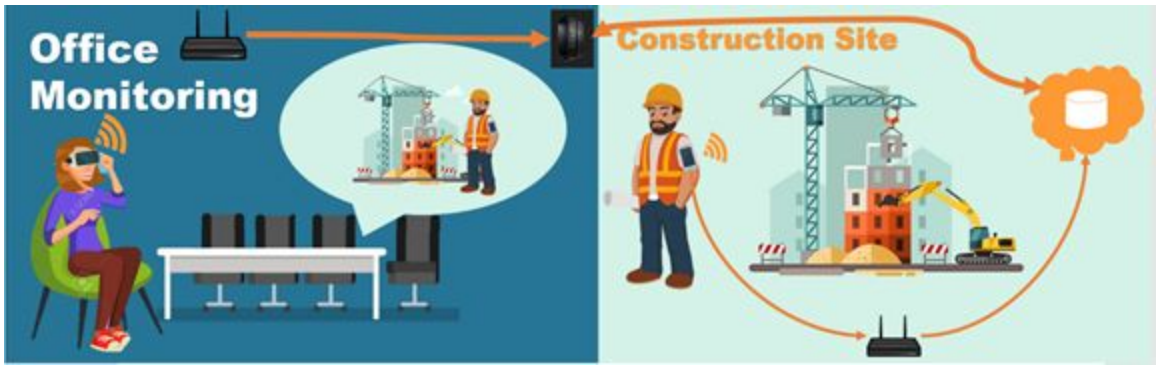


Figure 1. Concept Diagram

Office Monitoring

From the client's office a supervisor will be sitting in the conference room with the Microsoft HoloLens monitoring the jobsite. The HoloLens renders the construction site, buildings, vehicles, and employee avatars in augmented reality through a connection with the Rethink database. Once the Rethink database receives a location update, Rethink will automatically notify all change listeners attached to the given data, and transmit the information through the corresponding sockets (to the HoloLens). The HoloLens will use an access point from the office to receive this transmission.

Construction Site

Before the construction workers enter the work site, the Android application will calibrate to the user's specific orientation and step size. The phone will be attached to the users right arm using an armband and must be worn at all times during construction hours. As the construction workers walk around the site, the mobile application will record data from the phone's sensors. The data collected from these sensors will then be used to estimate the user's new position in real-time. Finally, these position estimations will be sent to the server in order to be saved in the database.

2.5.1 Architecture Diagram

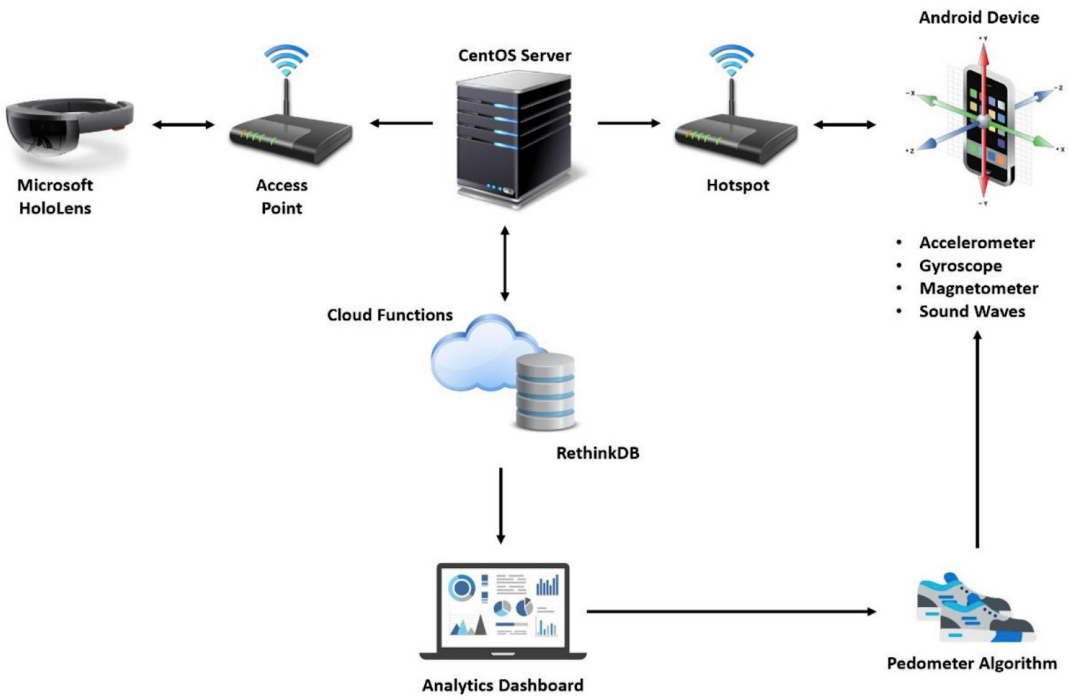


Figure 2. Architecture Diagram

Website

The website receives latitude/longitude coordinate data from the Android phone, saves the data in the database, and updates corresponding user markers representing the devices being tracked. The website will maintain a display which moves these markers through a 2D representation of the worksite (Durham) and trace the paths as they develop.

HoloLens

The HoloLens connects to the office access point in order to receive updates from the server. The server sends a notification when new location data is received from the mobile application. The HoloLens then takes this location data from the database and renders 3D avatars representing the current positions of all devices being tracked.

Android Application

The Android application detects user movement via the phone's sensors. The application determines if the user has moved from his/her current position, and if so, makes a new position estimation based on both the previous known location, and the collected data. The application then connects to an access point within the construction site (Durham) and sends this new position estimate to the server in order to be saved in the database.

Database

The database we are using is RethinkDB. Our schema will store 3D accelerometer data and position coordinate data. When the database receives a request from the server to store a new user position or piece of data, the database updates its storage and broadcasts the new data to the Microsoft HoloLens.

2.5.2 Process Flow

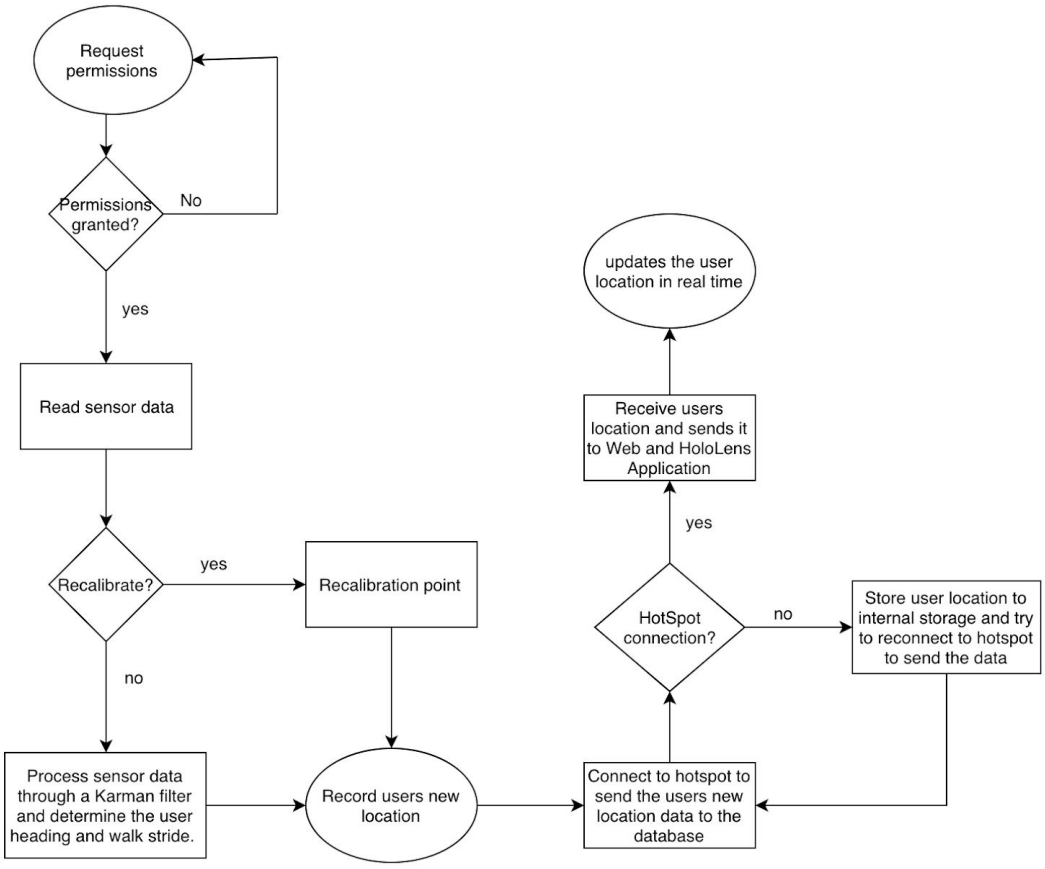


Figure 3. Process Flow Diagram

When the android device application is executed the user will be asked to accept permissions that will allow the software to perform the track the user location. If the user doesn't grant the software permission, then the application will request the user to grant the application permission until either they accept or kill the application. If the user grants the application permission it will then start to read the hardware sensor on the device. It will then scan the area for a recalibration point. If it finds a recalibration point with in 1 meter of the device, then it will receive the recalibration location from the beacon and then record the user new location. But if there is no beacon nearby then it will process the data gathered by the hardware and then calculate the user's new location and then record it. After the application has gathered a new user location it will try

to connect to a WiFi hotspot. If the application successfully connects it will send the location data to the database. If the application doesn't successfully connect to the hotspot it will then store the new user location to the hardware internal storage and try to connect to the hotspot again. After the database receives a new user location it will send it to the web and HoloLens application. After the web and HoloLens receives an update from the database it will update the user's location in their mapping environment.

2.5.2 Mobile Application

Java

Java is the primary object-oriented language used to develop Android applications. Our solution will utilize all 3 of the above concepts to create an Android application for tracking user movement. Our app will be written with Java and support devices operating on android API 21 or above. We will use Java to communicate with the Android API which in turn will communicate with the Hardware Abstraction Layer (HAL) and send us sensor data.

GPS

Primarily this sensor will not be used except for an initial position determination. Before entering a work site, GPS would be used to collect the latitude/longitude coordinates of the current user. This latitude longitude value will then be sent to the website, and mapped to a specific location in our 2D visual representation of the work site. Note, the website will display a 2D representation of the work site, while the HoloLens will provide a 3D representation.

Accelerometer

The Accelerometer sensor is used to measure acceleration. Android phones primarily use piezoelectric accelerometers. These types of accelerometers are built using crystal materials (usually quartz) which generate an electric charge when squeezed. This property makes it possible to infer the relative acceleration by measuring the corresponding changes in the electrostatic field. We will use the Accelerometer for step detection in our Pedestrian Dead Reckoning (PDR) system.

Gyroscope and Magnetometer

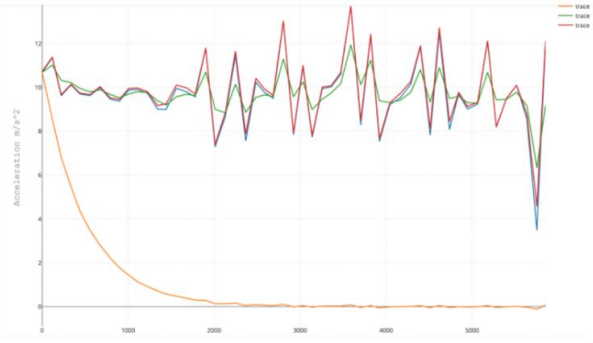
Typical phone gyroscope sensors are implemented with microelectronic mechanical systems (MEMS). The point being, they are hardware based, and not software based. The gyroscope is used to determine the cellphones orientation. Changes in a phone's orientation will be measured as angular velocities. We can then integrate these velocities to find displacement.

Bluetooth Beacon

The bluetooth beacon will emit an advertising packet to transmit data to the android device. The android device will scan for beacons within range of the device. It will then establish Radio Frequency Communications (RFC) and receive the advertising packet from the Bluetooth beacon

if it is within 1 meter. We can calculate the distance by using the TX power level that is transmitted from the beacon.

Step Tracking Algorithm



Magnitude values for 3D accelerometer data collected while walking 10 steps

Basic Algorithm

```
max = 11
min = 9
peakDetected = false
currentPeak = -1
stepCount = 0;

for item in data:
        if (item >= max)
                peakDetected = true
                currentPeak = item
        if(peakDetected)
                if(item > currentPeak)
                        currentPeak = item
                else if (item <= min)
                        stepCount++
                        peakDetected = false
                        currentPeak = -1
```

Figure 4. Step tracking Algorithm

To determine a user's location, we have to create an algorithm that can determine the distance a user travels based on their stride. We have two thresholds that will determine if the (x, y, z) accelerometer data is a peak or valley. The peak is the maximum value that passed the max threshold. After the peak is detected and the accelerometer data is decreasing, and it goes below the minimum threshold, then we know that the user has walked a step. We have three vectors. We have a start point, peak, and end point for our algorithm. The distance (foot stride) can be calculated by computing the difference between the start and end point.

Heading Algorithm

The heading determination algorithm serves an important part of our Pedestrian Dead Reckoning system. We are currently using the Android based Rotation Vector Sensor, which creates quaternion estimations of a phone's orientation based on a sensor fusion between the Gyroscope and Magnetometer. To improve our accuracy we can eliminate the Rotation Vector Sensor, and instead make our own quaternion estimations by reading the output from the Gyroscope. Essentially we will start with a current orientation quaternion, which we may either hard code as a default starting orientation, or estimate based on the magnetometer and accelerometer readings. Next we will use the Gyroscope outputs to create a rotation quaternion, which we can add to our current orientation quaternion in order to get a new orientation.

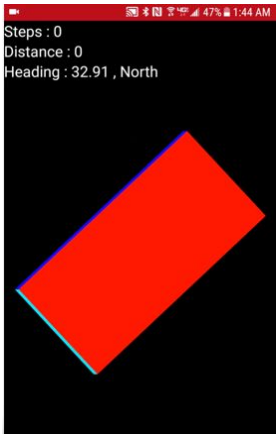


Figure 5 on the left shows a representation of a phone in 3D space. The phone is a rectangle with color coded sides (red is front). We created this simple OpenGL representation to aid in our understanding of quaternion rotation.

Figure 5. OpenGL Visualization

Bluetooth Beacon Algorithm

To re-calibrate a user's position our software will determine if the android device within 1-meter of the Bluetooth beacon. The txPower is the strength of the broadcasting power level constant that is determined by the advertisement protocol. The RSSI value is determined by the Strength of the RSSI value which transmits from the Bluetooth beacon. This algorithm is used to determine when the android device is within 1 meter from the Bluetooth beacon. This algorithm is needed to recalibrate the user's position due to error associated with the sensor readings. Figure 6 (below)

```
calculateDistance(int txPower, double rssi)
    if(rssi == 0)
        return -1
    ratio = rssi * 1.0/txpower
    if(ratio < 1.0)
        return ratio^10
    else
        accuracy = (0.89976)*(ratio^7.0795) + 0.111
        return accuracy
```

Figure 6. Bluetooth Beacon Algorithm

2.5.3 Database

RethinkDB is a scalable, open sourced database for Real-Time applications solutions. This database schema offers a unique solution over traditional database schemas such as MySQL. RethinkDB allows objects to register on change listeners to a specific data set. It is a classic Observer pattern, in which database changes will automatically trigger pre defined updates. We

will utilize this functionality to automatically update HoloLens data whenever a new location coordinate is saved.

RethinkDB is different from other real-time databases such as Google's Firebase or Amazon's Relational Database service because it is open sourced, and does not restrict on storage size or daily limits.

Some of the key functionalities to RethinkDB are outlined below.

1. An advanced query language that supports table joins, subqueries, and massively parallelized distributed computation.
2. An elegant and powerful operations and monitoring API that integrates with the query language and makes scaling RethinkDB dramatically easier.
3. A simple and beautiful administration UI that lets you share and replicate data.

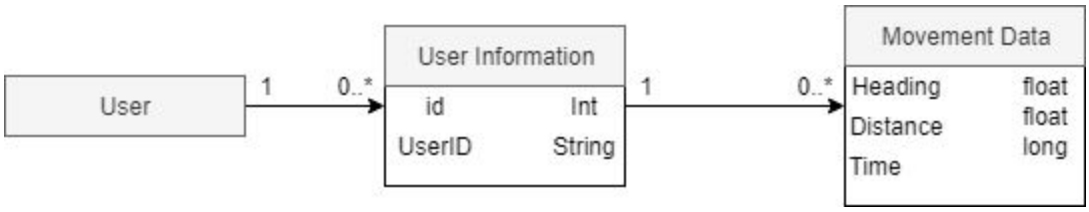


Figure 7. Database Schema Diagram

User Information

The user's id and UserID are stored for unique identification. id is stored as an integer for simple comparison when dealing with multiple users. The UserID however, is a string for readable identification of the user.

Movement Data

Within the Movement Data, three values are stored. Heading is stored as a float that represents the direction the user is facing on a 360 degree scale. Distance is the estimated step size of the user that is also stored as a float. Time represents what time, in milliseconds, that the data was collected on the app; stored as a long, this data can be interpreted to obtain the date and time.

2.5.4 HoloLens

Camera Scripts

The building zone camera script will allow office personal to monitor different zones of the jobsite by using an xbox controller, hand gestures, or voice commands to cycle through the different zones. The 1st person/3rd camera script will allow the office personal to monitor a worker

from the worker's perspective by switching to the avatar representing the worker and toggling between 1st or 3rd person camera angles. Reference the Camera controller script for more information.

Worker/Vehicle Movement Scripts

The player control script will allow the avatar to move from the current position to its new position received from the mobile application and then translate their movement in the HoloLens augmented relativity environment where it can be monitor by the client. When the database script receives a response from rethink database it will send new location information to the Software controller interface Script and determine what avatar or vehicle needs to be retranslated and send the movement script the new X,Y,Z coordinate. The movement script will translate the current gameobject to be move from its current position to its new position.

Particle System Script

The particle system script will render some special effects when an avatar hits an object (wall, vehicle) to notify the supervisor that a collision has occurred. When the avatar collides with game object the game engine will render a particle that will be displayed in the augmented reality. The collider detection script will detect when the avatar is collided with its surrounds (vehicle, Building) and send the information to the Software controller interface Script which will relay the message to the particle system script which will then render the particle effect in augmented reality.

Animation Script

The animation script will animate (idle state, walking state, running state, etc.) objects in the HoloLens depending on their perceived state. We will create animation clips for the avatar objects. Each state will have a transition to the next animation state. Each avatar instance will have an animation script that is an instance to the animator controller component that controls which state the avatar is currently in. The script will pass in the animation instance variables that controls which state is active. This script will be attached to the avatar game object and the state will be controlled by the data received from the mobile application.

Controller Script

The controller input script will allow the supervisor to switch between different zones, workers, or vehicles on the construction site. For our worksite (Durham) it will be decomposed into zones. Each zone in will have a camera associated to it. Each of the cameras will have implement camera control interface that will allow the controller script to toggle between different zones based on the users input from the xbox controller or hand gesture from the HoloLens. If the user presses the change camera to 1st or 3rd person view and there are currently attached to either an avatar or vehicle game object then the script will switch to either third or first person view. This script will be attached to the Software controller interface Script which controls which zone the user is viewing.

Collider Detection Script

The collider detection script will detect when a worker or vehicle has collided with an object on the job site. This script will inform the mobile application if the worker avatar has hit a wall or some other structure. The avatar will have a rigid body component attach to its game object that has collision detection by adding a collider component to the avatar game object. By adding a collider component with the rigid body, we call detect if a collision has occurred. The colliders will be attached to the Durham 3D model and when the avatar collides with the building it can be detected by using Raycast. A Raycast is a ray that send out in space that detects if a GameObject is collided with any other game object. This script will be attached to the avatar game object.

UI Menu Scripts

UI menu scripts will receive finger gestures and voice commands. These commands will be used for switching between zones and view perspectives. The UI Menu script will receive input from user wearing the HoloLens. The Software controller interface Script will receive the input and send it to the UI interface controller. The UI interface controller will have an array of UI components that implements the Menu UI interface and will call the interface method. Each component of the UI menu will perform the operation that is requested from the user.

Database Script

The Database script will be used to send and receive data from the database. If the database sends data to the HoloLens, this script will parse the data received, and send it to the player or vehicle script in order to render the appropriate translation. If the Database Script receives date from the rethinkDB it will parse the data and send it to the Software controller interface script. The software interface script will send that data to either the avatar or vehicle script and it will translate the correct game object to its new location.

2.5.5 Modeling

3D Modeling

The HoloLens renders 3D models of the worksite in augmented reality using the Mapbox SDK, a mapping platform for Unity3D. Our prototype involves tracking user locations inside of Iowa state campus buildings as a proof of concept. To be able to monitor individuals on the HoloLens we need to have 3D model of the campus building so that it can be rendered in augmented reality.

To location information about the campus building about the campus building our team search the Iowa state website www.fpm.iastate.edu. The site helped us locate a link to 3D SketchUP models from Trimble Warehouse that had 3D models of the campus buildings. The model of each building is in a .skp file format along with a texture pack for each campus building.

Unity requires that the 3D model be in the format of .obj or .aed to import it into its IDE. SketchUP Pro offers a converter that can convert .skp files to .aed and .obj model. Once a model file is converted and loaded into Unity3D, it can take the texture s files and apply them to the polygons of the 3D model. The model can then be geolocated to the exact location on the map platform to be rendered in augmented reality.

To be able to monitor individual workers and vehicles on the job site we need to create some vehicle and character models. Unity offers these models on the asset store and can be purchased and imported into our project.

Figure 8. 3D Model of Durham

2.5.6 Web

We will be using a web platform to show a 2D rendering of Durham, and the recorded trajectories of our 3 devices. The technologies we will be using include, but are not limited to, HTML, CSS, Canvas, JavaScript, and JQuery.

Front-End Architecture

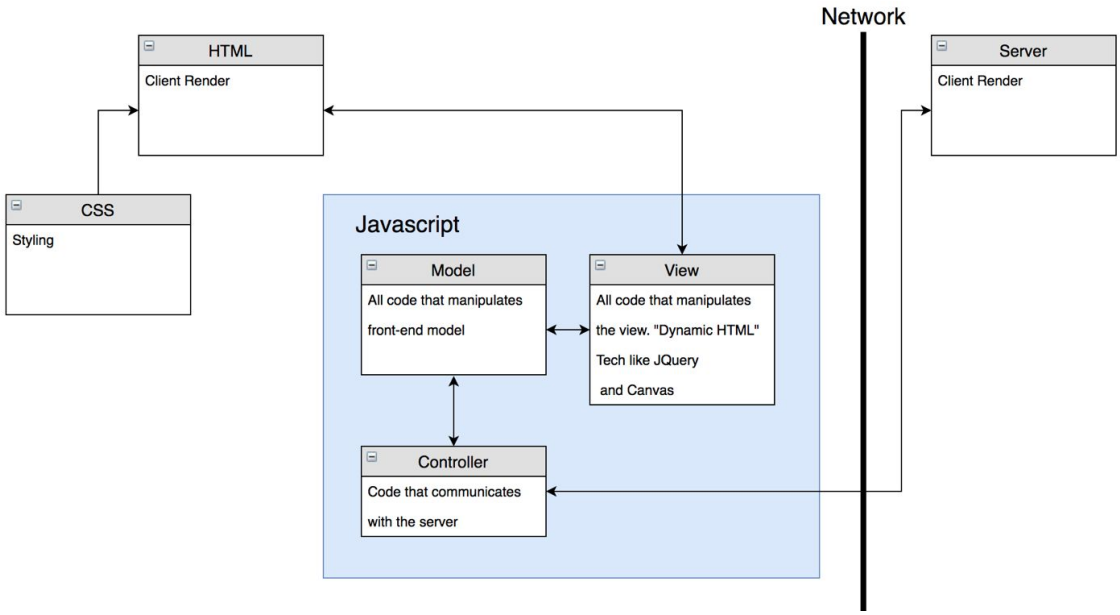


Figure 9. Front-End Architecture

The frontend architecture will be using HTML5 with CSS to render a Durham floor model canvas on the webpage. When the client makes a URL request to this website it will send a request to the server. The model will render the Durham floor plan that the user will interact with. Any requests the user makes will be sent through the controller. On the view, there will be a dropdown that will allow you to switch between users. When the client has selected a user, and hits the submit button, the controller will make an API call to the server and collect all of the location data from the mobile application. This will be rendered on the front-end for the client to observe.

Server Architecture

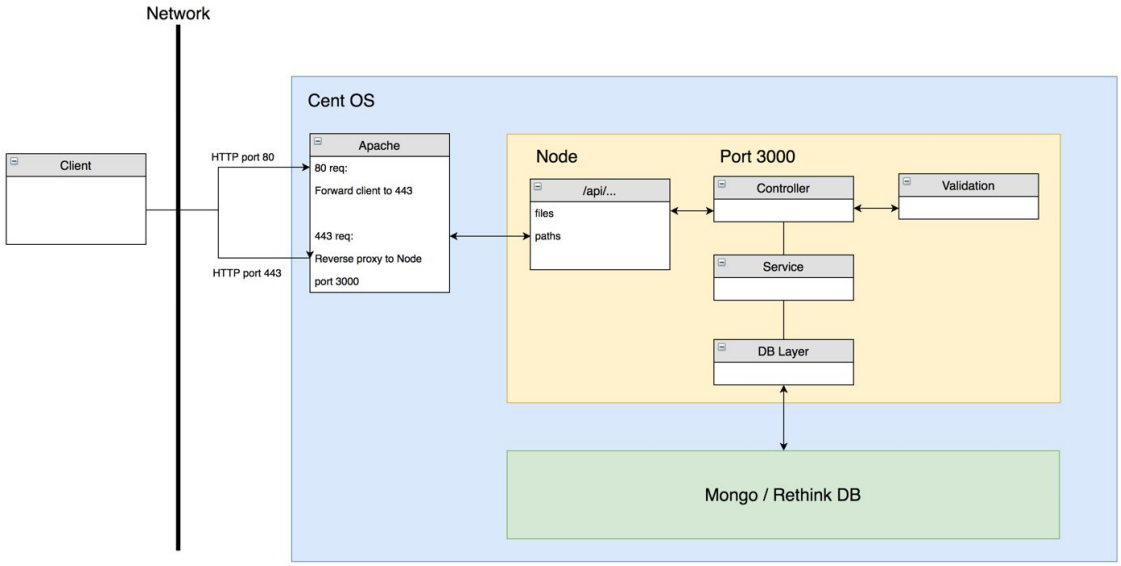


Figure 10. Server Architecture

The server architecture is fairly simple. The client will communicate with the server either through port 80 or port 443. If the client connects through port 80, they will be redirected to port 443 (HTTPS). Since Apache handles SSL very well, we are using that as our point of contact. Once the request is decrypted, it is sent to Node through a reverse proxy to Node running on a closed port, 3000. Node runs through the request described in the back-end architecture. When a database call is initiated, Node communicates with either MongoDB or RethinkDB, which are running on their own closed ports.

Backend Architecture

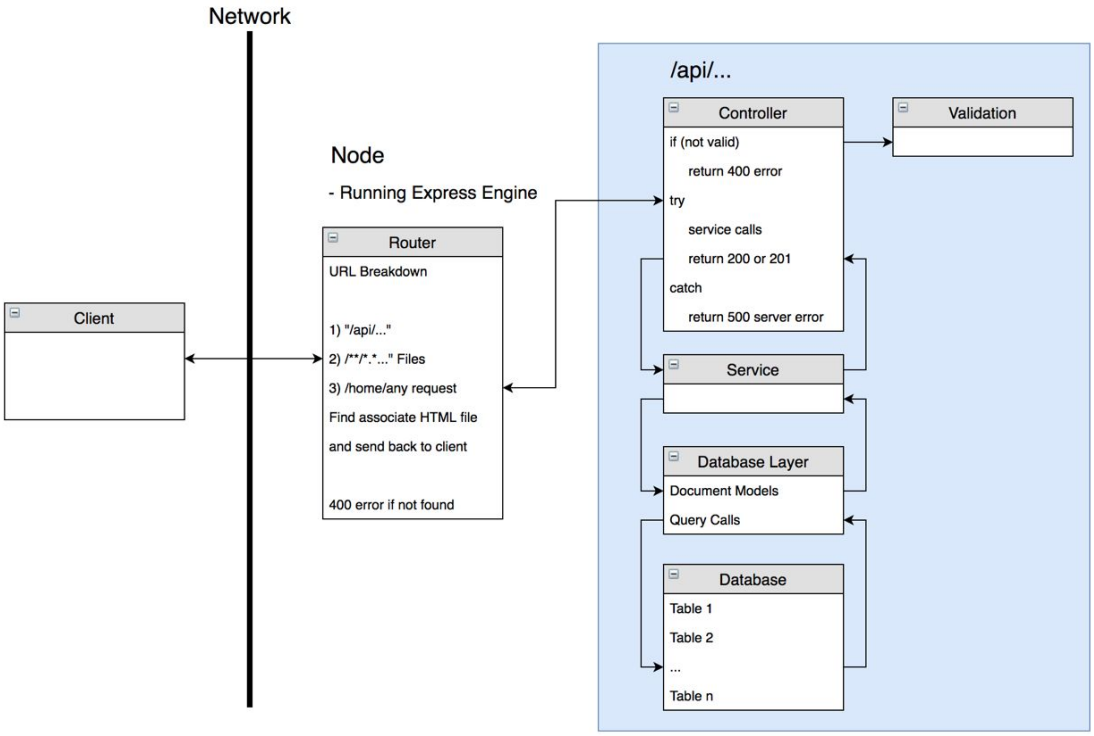


Figure 11. Backend Architecture

When the client sends requests to the API, it will go through a layered architecture. The way our layered architecture will work is exactly as the name might suggest. The request will go to our router, where it will be directed to send back webpages, or handle API requests. If it's an API request, it will go to our controller, where it will be validated and then sent off to the service layer. If the validation layer fails, it will send back a 400 response to the client. The service layer will manipulate the data as needed and communicate to the database layer. The database layer will create the database calls and do any CRUD operations. It will pull from the database whether that is MongoDB, or RethinkDB. Once the data has been collected, it will go back up the layer chain. The service will have the data at this point and manipulate as needed. It will send it back to the controller where it will send a 200 response if successful, or a 500 response if there was any internal error along the way. The validation layer will defend against injection attacks by making sure everything meets the requirements. This is a strict format, and at no point should layers cross communicate, i.e. the controller should not talk to the database layer.

2.6 Technology Considerations

Cross Platform vs Native

One of the problems surrounding mobile development is that the two main platforms (Android and iOS) run their applications on different languages. It would be great if we could code an application in one language, and then just automatically transform it for any platform we want to use. This is what Xamarin and React Native do.
Xamarin is an open source software that uses the .Net framework. Xamarin uses the C# programming language within the .Net framework. React Native is another open source software which uses JavaScript.

Pros of React Native

1. Developers who develop in reactive native can create apps for the web, iOS, and Android platforms.
2. Does not require multiple teams for multiple platforms.
3. Allows for rapid prototyping
4. Saves development time
5. Quickly solves cross-platform bugs

Cons of React Native

1. It is a fairly new software, so it has less support and documentation.
2. It is impossible to write a truly native feeling app in react native without also having knowledge of the native operating system.
3. Limited access to platform specific APIs.
4. Steep learning curve.
5. High security vulnerability.
6. Not suited for computationally intensive tasks.

Pros of Xamarin

1. Xamarin performance is very close to native performance.
2. Platform scalability
3. Xamarin is open sourced.
4. Provides its own IDE

Cons of Xamarin

1. Delayed support for new iOS/Android software updates.
2. Knowledge of the native platform language (Android, iOS) is required.
3. Xamarin applications are larger than native applications.
4. Compatibility issues with third party libraries.

5. Limited open source libraries

Advantages of Native Android

1. Better user experience.
2. Optimized for Android.
3. Compiles faster.
4. Unrestricted access to Android API's.
5. Third party compatibility.
6. More control.
7. Update support through the Google Play Store.

Disadvantages of Native Android

1. Applications written for Android, only run on Android phones.

Disadvantages of Cross-Platform

1. Lack of documentation.
2. Restricted access to Android frameworks, libraries, and hardware sensors.

iOS vs Android

iOS mobile devices have Accelerometer, Gyroscope, Pedometer, Magnetometer, GPS, Barometric, Proximity, Facial Recognition, Radio, Bluetooth, and Fingerprint sensors. Android devices have Accelerometer, Gyroscope, Magnetometer, Heart Rate, Light, Fingerprint, Proximity, Temperature and Bluetooth sensors. The devices available for our project are all Android. Our client has given our team two Android devices (HTC One, Galaxy Note).

Pros of Using an Android Device

1. Team experience.
2. Android is open source.
3. Android is programmed in Java.
4. Highly customizable screens
5. Expandable storage.

Cons of Using an Android Device

1. Complex layout schemes.
2. High fragmentation due to many different Android devices running on many different API's.

Pros of Using an iOS Device

1. Fingerprint /Face recognition increase security.
2. iOS using swift programming language is easy to read.

3. Swift is open-source.

Cons of Using an iOS device

1. iOS is not open source.
2. Integration issues with projects written in Objective-C
3. Internal storage thus you cannot upgrade the hard drive unless you buy a new device.
4. iOS framework documentation isn't as detailed.
5. XCode IDE has a steep learning curve.

Our team chose Android because as a whole, we have more experience working with Android than iOS. Also, Android has more open source resources, better documentation, and typically more affordable phones than devices running iOS.

Backend

Our solution revolves around storing and monitoring worker coordinate data. Since our solution requires having an access point to transfer the coordinate data to our database, our application must be able to store data internally when a network connection is not available. We will use Realm for this internal storage process.

For our external database we have considered MySQL, RethinkDB, MongoDB, and MariaDB. The advantages and disadvantages are as follows.

Internal Storage Advantages

1. Designed for small lightweight operations.
2. Reads and writes without no extra configuration.
3. Stored in a single file.
4. Increased performance on reading operations

Internal Storage Disadvantages

1. Designed for small datasets.
2. Writing and reading operations can't run concurrently.

MYSQL Advantages

1. Host-based verification.
2. Flexible privilege and password system.
3. Security encryption for all password traffic.

MySQL Disadvantages

1. No built-in support for XML or OLAP.
2. Speed deficiency on big tables (not good for agile).

MongoDB Advantages

1. Database engine supports JSON.
2. It can store any structure of data.
3. The data Schema can be written without any downtime.
4. Supports JSON.
5. Encrypted storage engine.

MongoDB Disadvantages

1. It was not designed to handle relational data models.
2. Setting up MongoDB is a long process.
3. Default settings are not secure.

MariaDB Advantages

1. Allows high scalability with easy integration.
2. Real-Time Access.
3. Fast query performance and processing.
4. Has the core functionality of MySQL.
5. Encryption for network, server and application levels.
6. Fast and stable.
7. Variety of plugins.

MariaDB Disadvantages

1. MariaDB is poorly supported on some operating systems (works best for Linux).
2. Migrating from MariaDB to MYSQL is difficult.
3. Not supported by hosting environment.

RethinkDB Advantages

1. Can listen for changes to data.
2. Supports sharing, parallel queries and MVCC.
3. Powerful query language, (Node driver for JavaScript developers).
4. Compatible with JavaScript (Node.js), Python, PHP, Ruby, C, C#, C++, Objective-C, Java.
5. Designed to be accessed from an application server.
6. Atomic updates.
7. Easy to setup and learn.
8. Open source.

RethinkDB Disadvantages

1. It's not ACID-compliant, doesn't have schema, it stores the field name of each document individually which can impact compression.
2. Mongo is 3x faster then Rethink for querying.
3. You need to setup your own auth and user accounts.

We chose RethinkDB because of the change feeds the architecture provides. The reason why this is important is because the HoloLens and website need to constantly update the worker's positions in real-time. If we used a different database, then both the HoloLens and website would have to constantly make pull requests to check for changes in data. By using RethinkDB our website/HoloLens team can subscribe to the change feed and will automatically receive an event when a location coordinate has been updated. This saves us from making unnecessary requests to the database.

We didn't go with mongoDB because our solution doesn't require a lot of pull requests unless the HoloLens and web operator want to render an early event. MongoDB has great performance when pulling documents of data from their database but since we don't need that functionality for our solution it is not feasible for our project. MYSQL database is very nice for storing data in a structure format but it doesn't offer change feeds meaning which it receives an update from the application it will send to the subscribed applications.

HoloLens

Possible AR technologies currently available are the Microsoft HoloLens, Google Glass, and Magic Leap. The Microsoft HoloLens operates by rendering 3D images into the visible world around you.

Advantages of HoloLens

1. Comfortable to wear.
2. Gesture recognition.
3. Voice command recognition.
4. Eye-detection based selection.
5. Hands free experience.
6. Unity development.

Disadvantages of HoloLens

1. Gesture recognition is very tricky and only a few gestures are available.
2. Costs $3000 for the developer edition.

3. Field of view gets distorted once you start moving.
4. Requires Windows OS for development.

Disadvantages of Google Glass

1. In 2017, Google only launched the Google Glass Enterprise Edition to companies like Boeing.
2. Google Glass Developer Edition was discontinued in 2015.

Advantages of Magic Leap

1. Much wider field of view than the HoloLens.
2. Two different size glasses which depend on your interpupillary distance.
3. Lighter hardware on your head.
4. Supports Mac development.

Disadvantages of Magic Leap

1. Costs $2,295 for developer edition.
2. Light pack computing device needs to be attached to your belt and has a USB-C cord attached to the headset.
3. If the computing device breaks it will cost an extra $495 to replace.
4. The leap glasses level needs to be aligned almost perfectly.
5. Leap technology tracking and sense recognition is more jittery than HoloLens. Space mating doesn't work properly.
6. Problem when glasses have to scan a room. Menus can go through walls.
7. Excessive heat on your head from wearing the device.

Our client is working on several projects on the Microsoft HoloLens to increase visualization solutions for his business. Our client product line involves around optical solutions. Therefore, our team will be using the Microsoft HoloLens because it provides visible solution that stills allows the user to interact with the world around them.

2.7 Safety Considerations

For the scope of our project, there will be no safety issues so long as the users behave normally. Even in the case of the project being implemented on a construction site, there would be no "new" safety consideration. While a construction site may have dangers, this project won't introduce any new risks.

2.8 Task Approach

At the beginning of the semester, our client proposed an idea for a product to track an individual's location in situations where GPS is not available. We will accomplish this task by developing an Android application. This app will access the phone's sensors and make position estimations based on received data.

2.9 Possible Risks And Risk Management

Few risks are foreseen for this project. Due to the need for this system to have many phones in a construction area for several hours each day, there is a risk to the devices being dropped or damaged. This threat can be minimized by using padded and secured armbands on each individuals right arm. The use of these arm bands should not pose any "new" risk to the users' safety. There is also the risk our client is taking when purchasing supplies for our project. This can be lessened by avoiding expensive purchases whenever possible, either finding cheaper alternatives or using equipment we already own.

2.10 Project Proposed Milestones and Evaluation Criteria

- Meet client

Meeting with the client would include meeting them, but also figuring out what their needs are. Emphasizing what their problem is, and figuring out how we can help.

- Develop relationship with client to continue product production

Continue talking with the client to build a relationship. This is needed for continued success of the project.

- Ideating - conceptualizing ideas for the solution

Generate ideas on how we can solve the clients issue. These ideas require client feedback. We want to be sure our solution meets our clients needs.

- Design requirements

Functional and nonfunctional requirements as well as project limitations.

- Software designs

Design documentation and revision.

- Prepare for product demo

Present data to client for approval to continue onto the next phase (The product demo).

- Prototyping

Create an iteration of the project which covers functional needs.

- Reiteration

If prototype fails to meet expectations, continue to iterate on the prototype, in order to improve.

- Work on project

This is the actual design and coding phase of the full project. Kanban, sprints, and or other project management will be utilized to complete internal milestones.

- Testing

Once the project is approved, we will conduct testing to ensure quality.

- Go Live

This is the final project delivered to the client. All needs have been met at this point.

2.11 Project Tracking Procedures

- Trello
  - For managing Kanban tasks
- Github
  - For tracking code production and revision
- Weekly Status Reports
  - For monitoring weekly accomplishments, and updating timeline as needed
- Slack
  - For group messaging
- Google Docs
  - Shared documents for weekly reports

2.12 Expected Results and Validation

Functional Requirement:	Test Plan:	Verification:	Validation:

Indoor/outdoor user tracking	We will setup 3 android devices at certain starting locations for tracking. We will provide the user with a specific path to walk which includes a start point and an end point. We will record their path and display it on Mapbox and compare their path with the test path.	Integration Testing: The AR team will design unit tests: An avatar game object instance gets created for each individual set up to be monitored via our application. Compare the avatar path with the test path and report the results The Web team will design Unit tests: Will store the user's path from the start point to their end point and compare their path with the test path and report their results. The backend team will create unit tests: Verifies that the schema tables get the correct user path from the mobile application. The test from all 3 teams will have code walkthroughs and reviews to verify that the unit tests are tracking 3 individuals according to their path trajectories.	AR validation: The tester shall verify that there are 3 avatars that are rendered in the 3D worksite. They will verify that their path matches the test path specified by the test plan and it is rendered in the correct location on the map. Web Validation: The tester will verify visually that the users path taken has the same longitude and latitude coordinates as the test plan. Database Validation: It will verify that the table schemas recorded all the location data from the 3 users' path from the mobile device. After the tests are completed, the tester will do a report analysis of the errors that they found and will be reviewed by the development team.

Movement Sensitivity	We will turn on the mobile application and then have the user perform normal walking and running speeds.	Usability Mobile Test: The user will launch the application and perform a walking movement and running movement and report the results. The mobile team will have a walkthrough to verify how this test is going to meet moving sensitivity requirement.	Tester: The user will turn on the mobile application and will perform running and walking movement and look to see if the mobile device detects their walking and running speeds. After the tests are completed, the tester will do a report analysis of the errors that they found and will be reviewed by the development team.
Store tracking to Rethink DB	We will connect the mobile device to a Wi-Fi and will then perform a step and make a http post request to the database.	Mobile Unit Test: Verifies that the mobile application does an HTTP request when the device is connected to a WI-FI access point. Backend unit Test: Verifies that the data received from mobile application matches with the recorded data from the mobile application. The backend and mobile team will have code walkthroughs and reviews to verify that they the unit tests are fulfilling sending data through an access point according to the specifications of the storing tracking information requirement.	Mobile Tester: Verifies that the application does an HTTP request to the server and the connection was successful. Backend Tester: The tester will verify that the data recorded on the phone matches with the data stored on the database table scheme. After the tests are completed the tester will do a report analysis of the errors that they found and will be reviewed by the development team.

Distance Accuracy	We will give a user a new starting location to start the application at. We will instruct him/her to move 1 meter and see if the new current location is within meter.	Mobile Unit Test: Create a unit test that starts at Location A and moves to location B and then verifies the user new location is within our 1-meter accuracy requirement. The mobile test team will perform a code walkthrough that explains how their test meets the specification of the distance accuracy requirement. The review team will check documents and files to ensure that the code meets our coding standards.	Mobile Tester: Verifies that the movement from Location A to Location B is within 1-meter accuracy. After the tests are completed the tester will do a report analysis of the errors that they found and will be reviewed by the development team
Delay Accuracy	We will use a timestamp when the mobile application has gathered all the sensor data and starts to perform the smoothing/prediction algorithm while the software creates a 5 second delay. After the delay is completed the software will create another time stamp and compute the time difference.	Mobile Unit Test: Create a unit test that takes the time stamp after the data has been collected and after the x second delay has been completed. Then compute the time difference and compare it with delay requirement. The mobile test team will perform a code walkthrough that explains how their test meets the specification of the delay accuracy requirement. The review team will look over the documents and files and confirm it meets our coding standards.	Mobile Tester: Verifies that the delay time of the application is within the 5 second threshold by running the test script and report the results. After the test is completed, the tester will complete a report analysis of the errors that were found. The development team will review these documents.

Drift Accuracy	We will start a user at a starting point by Durham. We will then have the user move 2 meters in any direction and record the position and then do a computation of the possible positions and see if the user is within the 1 meter of the computed location.	Mobile Unit Test: Create a unit test that takes the starting position of the test plan and then computes all the possible new position paths the user could had taken and then compare those position with the user's current position and report the results. The mobile test team will perform a code walkthrough that explains how their new positions will meets the specification for the drift accuracy requirement. The reviewers will look at the code to see if it meets our coding standards.	Mobile Testers: Verifies the users new position is within 1-meter of the possible computed positions that was computed. After the test is completed, the tester will complete a report analysis of the errors that were found. The development team will review these documents.
HoloLens Monitoring	The HoloLens will receive the new recorded location data from the Rethink database. It will then render the users new position in the augmented reality map. The user that is wearing the device should see the avatar move from the user's current position to the new position that was received.	AR Unit Test: Create a unit test that takes the newly recorded data from RethinkDB and calculate the new position that the avatar transform should read after unity renders the new position. Compare the avatar's new position with the computed position and report the results. The AR team will perform a code walkthrough that explains how their new computed transform position meets the	AR Testers: Verifies the new transform position the avatar moves to in the AR map is the same as the newly computed position. After the test is completed, the tester will complete a report analysis of the errors that were found. The development team will review these documents and address the software errors that were reported.

		specification of the HoloLens monitoring requirement. The reviewers will look at the code and see if it meets the unity coding standards.	
Monitor Accuracy	We will create a timestamp and send a location packet by using a http request to the database. When the database has received the packet, it will create another timestamp. The database will then send the location packet to the website and HoloLens software. The HoloLens and website will then create a new timestamp and calculate the difference and see if it is within 10 second threshold.	AR Unit Test: Create a unit test that take the recorded timestamp from the mobile application and create a new timestamp. Then compute the difference and compare it with the 10 second requirement and report the results. Web Unit Test: Create a unit test that takes the recoded timestamp from the mobile application and creates a new timestamp. Then compute the difference and compare it with the 10 second requirement and reports the results. Backend Unit Test: Create a unit test that takes the recoded timestamp from the mobile application and create a new timestamp. Then compute the difference and compare it with the 10 second requirement and reports the results. The AR, Web, and Backend will perform a code walkthrough that explains how their time	AR Testers: Verifies the that recorded location is being displayed on the map in AR within the 10 second threshold from the time it was recorded on the mobile device. Web Testers: Verifies the that recorded location is being displayed on the website map within the 10 second threshold from the time it was recorded on the mobile device. Backend Testers: Verifies the that recorded location is being displayed on the website map within the 10 second threshold from the time it was recorded on the mobile device. After the test is completed, the tester will complete a report analysis of the errors that were found. The development team will review these documents and address the software

		calculation meets the monitor accuracy requirement. The reviews will look at the code and see if it meets the coding standards.	errors that were reported.
Bluetooth Sensor	The android device will be paired with the beacon so that they our connected to each other. We will set the beacon at a location x inside the jobsite. When the android device receives a signal from the beacon, it will send a location update to the android device.	Mobile Unit Test: Create a unit test that takes the updated the location from the beacon and then compares it with the current location that is stored on the device. It should compare these two and report the result. The Mobile team will perform a code walkthrough that explains how their comparisons meet the Bluetooth sensor requirement. The reviews will look at the code to make sure that the developers are meeting the coding standards.	Mobile Tests: The user will stand within 1 meter of the beacon to trigger the location update. They will then verify that the mobile devices current location is getting updated. After the test is completed, the tester will complete a report analysis of the errors that were found. The development team will review these documents and address the software errors that were reported.

Android Low Battery Notification	The android device battery must be drained to 10% of its total capacity. Then we will run the application on the android device.	Usability Mobile Test: Let the device get below 10 percent of battery and display a low battery notification. The Mobile team will perform a code walkthrough that explains how their notification meets the Android low battery notification requirement. The reviews will look at the code to make sure that the developers are meeting the coding standards.	Mobile Tester: Tester should see if the low battery notification displays when the devices battery is below 10 percent. After the test is completed, the tester will complete a report analysis of the errors that were found. The development team will review these documents and address the software errors that were reported
HoloLens Low Battery Notification	The HoloLens battery must be drained to 10% of its total capacity and then run the application on the device.	Usability AR Test: Let the device get below 10 percent of battery and display a low battery notification. The AR team will perform a code walkthrough that explains how their notification meets the HoloLens low battery notification requirement. The reviews will look at the code to make sure that the developers are meeting the coding standards.	AR Tester: Tester should see if the low battery notification displays when the devices battery is below 10 percent. After the test is completed, the tester will complete a report analysis of the errors that were found. The development team will review these documents and address the software errors that were reported

Table 4. Function Requirements Test Plan

2.13.1 Non-Functional Test Plan

Non-Functional Requirements	Test Plan	Verification	Validation
Battery Life Cycle	A user will turn on the mobile device and launch the application as a background process to monitor the battery and CPU usage throughout an 8-hour work day.	Usability Mobile test: The user will launch the app and monitor the mobile application every hour to report the results. The Mobile team will explain why the monitoring test meets the Battery Life Cycle requirement.	Mobile tester: The user will launch the mobile application and monitor the Android profiler every hour for 8 hours. After the test is completed, the tester will complete a report analysis of the CPU usage. The development team will review these documents and address the software battery issues, if any.

GPS Sensor	We will turn off the GPS location service on the mobile device and create a pre-planned walking path around Durham. We will record the path walked, compare it with the preplanned walking path and show the differences.	Mobile Unit Test Create a unit test which takes the pre-planned route, compares it with the user's route and reports the results. The mobile team will perform a code walkthrough that explains how their unit test meets the GPS sensor requirement. The reviewers will look at the code to ensure the developers are meeting the coding standards.	Unit test where you create a walking path, perform the walk, compare the expected path with the actual path and return the result
Look and Feel	We will place the mobile device inside of the armband and wear the device for 8 hours.	Usability Test: We will create a test that requires the tester to wear the mobile device for a normal work day and report the results.	Mobile Tester: The tester will wear the device around his/her arm for 8 hours and then report the comfortability to the development team.
Environment	We will launch the HoloLens application at the clients facility inside of their conference room.	Usability Test: We will create a test that requires the tester to wear the HoloLens inside of the clients facility.	AR Tester The tester will go to the client's facility and see if their facility has the proper environment for running the HoloLens application. The tester will complete a report analysis of their findings to the HoloLens development team.

Table 5. Non-Functional Requirements Test Plan

2.13.2 IEEE Standards

Our team reviewed several IEEE standards that would apply to our project. The IEEE Technical Committee Report in Recommended practices for Burst Measurements in the Time domain is about looking at peak ranges of noise caused by bursts. It shows a study how the results that are recording from reading sensor data can be analyze. This can be done by recording the magnitude and duration of each burst. This standard is intended to classify the noise of the transmitted data from signals that are send from hardware. Our team can use this standard because our solution will have to deal with noise surround beacons, and mobile sensors. The accuracy of the beacon distance signals from the device can have a lot of error due to RSSI values that is transmitted from the beacon to the mobile device. Based on this article, nice can be measured by the under bust and lower bust intervals and a bust duration. You can also view different burst magnitude of sensor such as instantaneous average and peak which is the highest output value. We are using a similar method for determine our step count by looking burst of peak and valleys to determine a user stride length. We can apply this IEEE article for the beacon noise as well to determine the peaks and values of noise in order to find the best time accuracy to achieve a functional requirement. The standard also talks about the awareness of energy busts that can be applied to our accelerometer data which interferes with signals and create noise.

The IEEE International Standard- Systems and software engineering Life Cycle Processes us fir defining requirements and the process of the software undergoes throughout the stages of development. The process starts from a project blastoff meeting that will discover the components, scope and identifying your stakeholders. There are two types of requirements: Functional and nonfunctional that our team will need to consider. Non-Functional are quality requirements that specify the characteristic of your system. Functional are the implementation requirements that serve as a contract to both the developers and stakeholders. Requirements are the language that is used to clarify what the system must meet after the software life cycle is completed. The requirements should have attributes that contain identification information to allow traceability. Requirements are intended to solve the client's business problem. It defines the requirements that your software must meet and any restrictions that your software should satisfy. The System and software life cycle process can apply to our project because we have to meet with our client which is like the project blast off and define the functional and non-functional requirements along with the constraints of our project. We spend several meetings to refine the requirements to make sure that they are accurate and can be traced and tested.

The IEEE Standard for Software Unity Testing is to specify a standard approach for unit testing that is used for a based line for evaluating critical software components. The IEEE standard unit testing process has three phases:

1) Perform the test planning

2) Acquire the test set
3) Measure the test unit

The Standard gives you guidance for implementing test procedures against your requirements to ensure that your software meets your client expectation.

The document discusses a general approach on how you can come up with a schedule and what resources your testing team will require to meet the requirements specifications. Once you have developed a schedule and an idea how to apply unity test such as what technology is required, how long will it take to write tests, you can define a test set. This is important because you can test for every case. For example, if you have to test all possible combinations of 10 items, you will have 10,1000 combinations of test that you must test. instead you should define a set set that covers the whole domain. Which a domain agreed upon, the next step is to execute and evaluate the results to determine how to fix them. The Standard for Software Unit testing will help our team define coverage for meeting our 1-meter accuracy requirement. This will involve input from may sensors including:

1) Magnetometer
2) Gyroscope
3) Accelerometer
4) Pedometer.

If we were to write test for all the possible combinations for these sensors our team would not be able to finish the project in two semesters. Not only there is a lot of combination but there is a lot of noise associated with the sensors which can output different readings that would be impossible to test. Therefore, we need to define a test set that covers the primary or average value of each sensor and possible outliers.

3 Timeline, Resources, and Challenges

3.1 Project Timeline

Fall Timeline

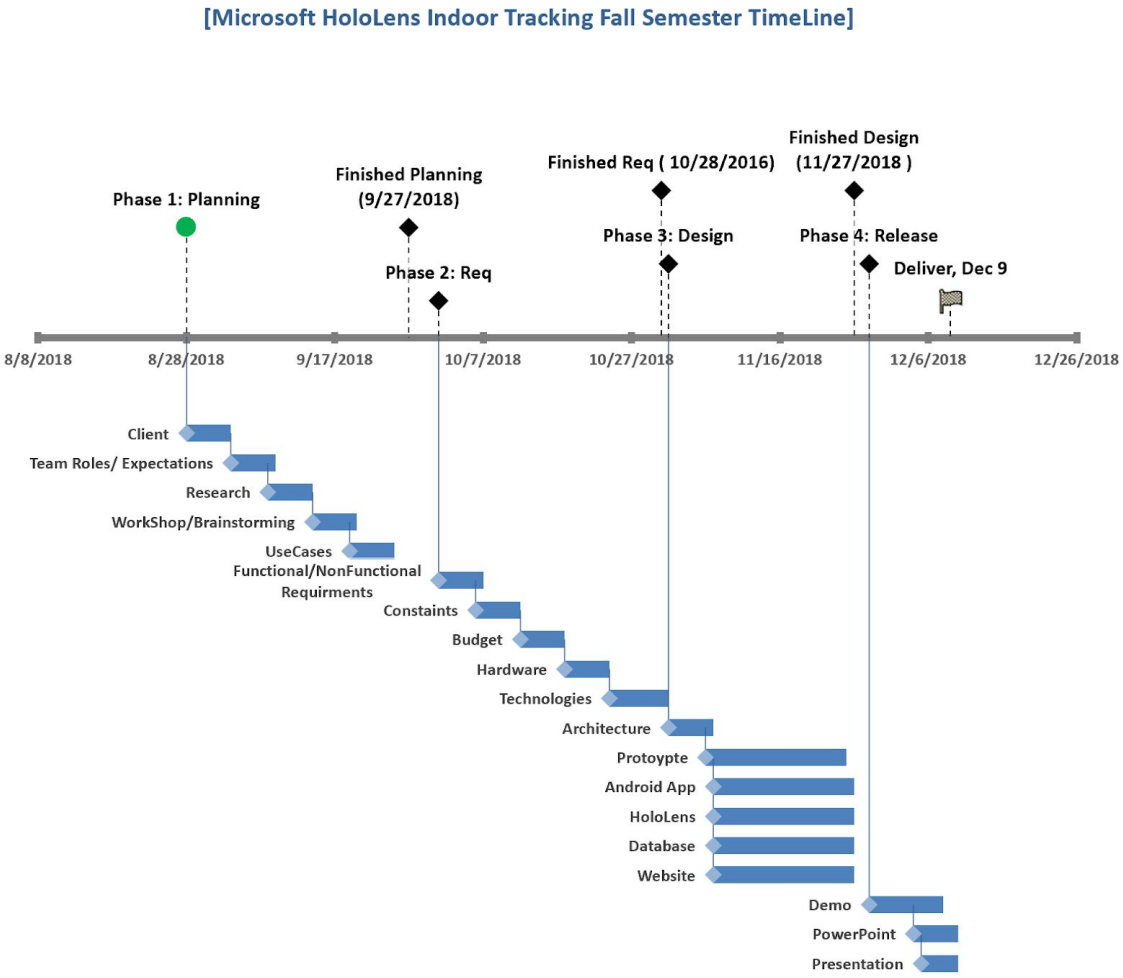


Figure 12. Fall 2018 Timeline

Phase 1: Planning

Client

We met with our client and learned about his company, Optical Operations. We talked about the goal of the project which entails interfacing with the sensors of a cell phone and displaying 3D

renderings on the HoloLens. The scope of the project is to create an indoor tracking solution using a mobile device without the use of GPS. The deliverables of our project include 3 parts:

1. Android Application
2. Website
3. HoloLens 3D monitor environment.

Team Roles and Expectations

Our team held a meeting to identify team member strengths and weaknesses. We then used that data to create team roles and expectations for the fall 2018 semester. After creating roles for our project, we divided up tasks based on the agreed upon roles.

Research

Our project consists of tracking users both indoors and outdoors without GPS. In this phase our team will perform a market analysis of what technologies already exist by researching indoor tracking conferences, competitions, and scholarly articles.

Workshop:

At this point of our timeline we identified the previous work literature and developed a good idea of the different technologies commonly used to track devices indoors. Our team used brainstorming techniques to identify the central idea of our project.

Use Cases:

We used a concept diagram to identify how the construction workers and office monitors will interact with our system.

Phase 2: Requirements

Requirements

During the meeting with our client we discussed how the last capstone group used WiFi tracking with a raspberry pi zero. We learned that our project would revolve around interfacing with the sensors of the phone along with creating a 3D environment on the HoloLens. Please reference table 1 for more details.

Constraints

Our goal is to provide an indoor/outdoor tracking solution for construction workers to wear during a work day. The constraints to our solution must not impede the workers ability to perform on the job site. Please reference table 3 for more details.

Budget

For our solution we will require a Bluetooth beacon for location recalibration. Also, we will need a server to store the location data. If the phone sensors don't reach the 1 meter functional requirement, we may need additional hardware

Technologies

The technologies we will be using for this project are the following: RethinkDB, Unity IDE, Java, Android Studio IDE, C#, Java, NodeJS, CS, and PHP.

Phase 3: Prototyping

Android Application

The Android Application prototype will use the Bluetooth, accelerometer, gyroscope, and magnetometer sensors to estimate the device's location. We will implement a login system to track specific users and their work locations. The prototype will send a connection request to the google server and send the users location in real-time. See Section 2.5 for more details.

Phase 4: Release

Demo

After we have successfully created our prototype, we will demo the project to our client and receive feedback.

Power Point

Create a powerpoint for a presentation.

Presentation

Present to the board of facility members.

Spring Timeline

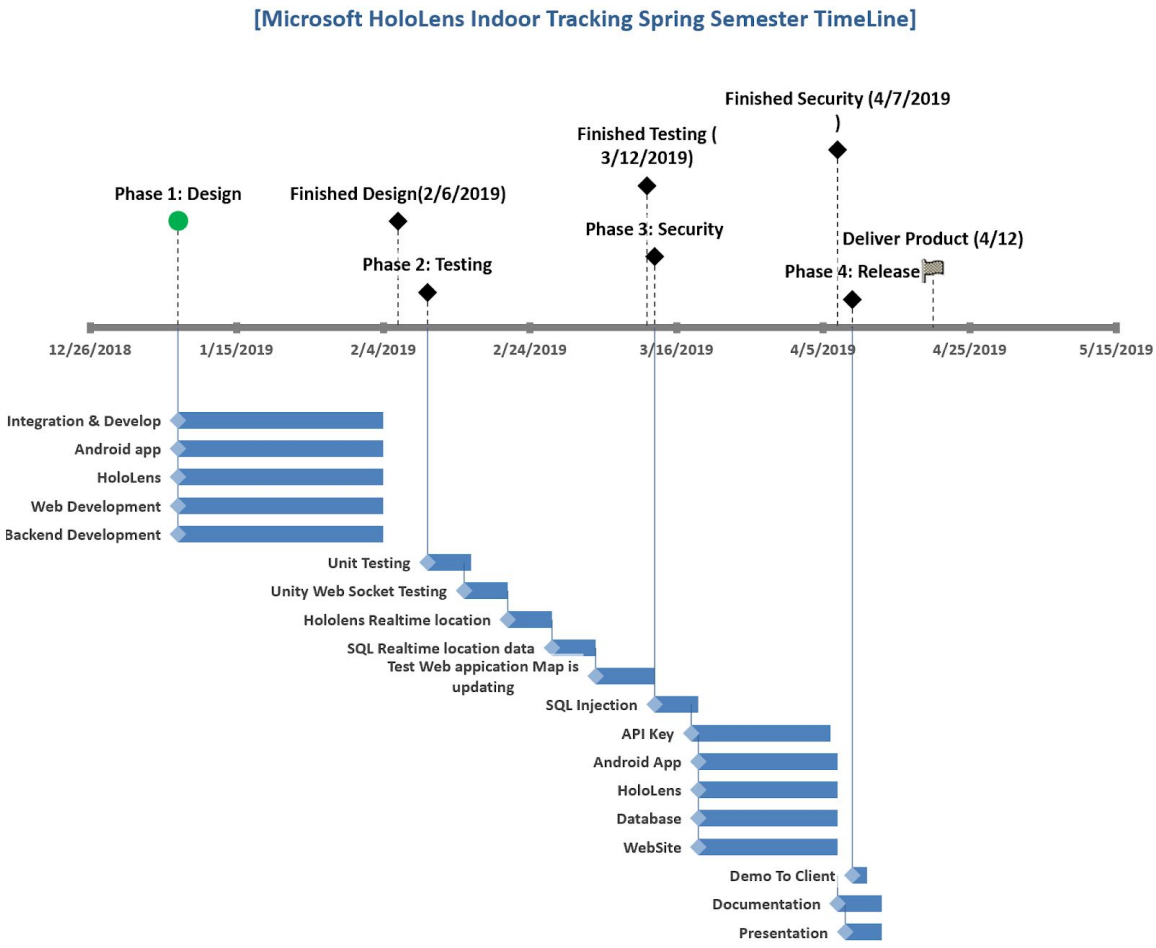


Figure 13. Spring 2019 Timeline

Phase 1: Design

Android App

The Android application will continue to implement algorithms for improving tracking accuracy.

HoloLens

The HoloLens prototype will create a 3D model of Durham and import it into the Unity IDE environment. We will be using Mapbox SDK as the mapping platform to render our model on the map. We will be developing scripts to provide the functionality needed to update and display location changes in the HoloLens. See section 2.5 for more details.

Database

The database will use a Google Cloud Server as is provided by the client, along with RethinkDB. This server will continuously send user location data to the HoloLens in real-time. See section 2.5 for more details.

Website

The website prototype will show the floor plan of Durham and display the user's current location. It will then update device positions as it receives new information. It will also have a data analytics dashboard to display the peak values of the accelerometer data. The data will be used by the Android team to develop more robust algorithms for step detection and location estimation.

Phase 2: Testing

The Android team will use the Espresso testing framework and UI Automator. For the HoloLens our software will be using the Test Runner framework which allows us to test in both an Edit and Run mode. This allows us to create different tests that invoke Monobehaviour methods. For the website we will be using Cypress. The utilization of these testing frameworks allows us to implement a test-driven development methodology.

Phase 3: Security

To prevent SQL injection our software will sanitize data before sending it to the server. In addition, we will use prepared statements to prevent attackers from introducing undefined queries. Our backend will use whitelisting on input validation.

Phase 4: Release

During the release phase we will work with the client on creating a demo for all three parts of our project (Android app, HoloLens, Website) and will eventually transfer our finished project to the client’s server. We will work on preparing a final poster and make the necessary changes to our documentation.

3.2 Feasibility Assessment

Project Overview

Our client had a previous team that used RSSI Triangulation (WiFi) to discover a user's location. They measured the intensity of the signal strength from multiple receiver locations to triangulate the user’s position. The team turned the position into latitude and longitude coordinates which affected their decimal accuracy and made the solution very inaccurate.

Our project uses the phones secondary sensors to discover the user's location. This is more feasible because it does not rely on multiple access points. There are many annual conferences and competitions which address methods for tracking device locations with different technologies. Sadly, there is not a perfect solution. Therefore, it can be expected that our prototype may not meet the accuracy requirements specified in table 1 and identified by the client. The good news is that there are solutions which do exist, and are able to meet our requirements. Our team can learn from these solutions.

Technology Overview

Our project is using new AR technology with the Microsoft HoloLens. The technology is still very young, and primarily lacking in documentation. The Android application is going to use the phone sensors to predict an individual's location within 1 meter of accuracy. The Android API allows our application access the different hardware and software sensors implemented in Android phones. While the API has good documentation on how to communicate with the hardware layer, our team may have difficulties integrating the sensors with mathematical formulas. For the backend our team will be using RethinkDB which is an open source software. Our backend team has used other forms of databases but is unfamiliar with RethinkDB, so there will be a bit of a learning curve. Despite this, we feel that the project is feasible due to our teams diverse experience in web, mobile and game development.

Cost

The Microsoft HoloLens costs $3000 for the developer edition. The average Android phone costs about $100. Additional hardware may require another $100 or less.

Schedule

The schedule for our project has been created in a way that considers our team's class schedules and other extra activities. We have created a schedule in a modular way that allows a fair amount of time to create the required task for each phase.

Summary

Based on the information outlined in the feasibility report, we can conclude that we are capable of creating a prototype for tracking device location without GPS; however, we may run into several setbacks regarding the 1-meter accuracy.

3.3 Personnel Effort Requirements

PERSONNEL	EFFORT

Anthony	Web Development/Security
Ben	Android Application Development
Ryan	Android Development, Database Management/Research
Cory	Research/Logistics
Travis	HoloLens development
Jose	Web Development

Table 6. Personnel Effort Requirements

3.4 Other Resource Requirements

- Android phones for testing
- Server space for website
- HoloLens for testing

3.5 Financial Requirements

Item	Description	Cost	Reference
HTC One	We need this Android device for testing.	FREE	Adviser
Microsoft HoloLens	The HoloLens is a requirement for our solution as requested by our client.	FREE	Client (Optical Operations)
Android Device Armband	The armband is a requirement for our project.	$39.99	Amazon

Google Cloud Server	The Google Cloud Server is a requirement so that our client will be able to access our data after we complete the project.	$50/month	Client (Optical Operations)
Bluetooth I-Beacon	The beacon is a requirement because it is needed to recalibrate the users location on the Android device. This will help improve the accuracy of our result and help maintain the meter requirement.	$30	Amazon
Durham Building Asset	The Durham model asset will allow us to create the school testing grounds on the HoloLens Environment. It is needed so that an office employee can monitor our test users.	FREE	SketchUp Pro Software
Microsoft Pro	The Microsoft Pro operating system is needed to develop with the HoloLens emulator.	$100	Microsoft Store

Table 7. Financial Requirements

3.6 Risks

Team communication is one of the risks that our team faced. When we met in the beginning of the semester, we discussed roles and responsibility that each team member would be responsible for. We pick roles and responsibilities way too early in the development process. We didn't have clear understanding of the scope of our project which made it difficult to set development roles for each team member. Because of not have a clear understanding of the team roles we were not able to meet deadlines due to the complexity of each component of our project. We didn't use our team communication effectively (slack) which made it difficult to gauge the progress of the give task assigned to each team member.

Our team collaboration on the design of our solution was lacking due to the poor communication. We did not effectively discuss a plan for our team to use to solve our client's problem. We understand the scope of our project and the overall goal but since the whole team communication relied on team leader, we didn't have a clear direction of our project. this made it difficult because they would research and went down different plans and stead of Collaboratory discussing all the different possibilities our solution could entail and picking the best solution that would meet are 1-meter requirement.

4 Closure Materials

4.1 Conclusion

We are confident in our ability to design and implement this project within our proposed time frame. We have already begun certain phases of development, including sensor implementation and data filtering. By the end of the Fall semester we will have a prototype of the Android application with tracking algorithms implemented. By the end of the Spring semester we will have added the Web and HoloLens functionality, and provide an interface for viewing the trajectories of 3 separate smartphones.

4.2 References

1. A. Jimenez and F. Seco, “Finding objects using UWB or BLE localization technology: A museum-like use case,” 2017 International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2017.
2. L.-H. Chen, E. H.-K. Wu, M.-H. Jin, and G.-H. Chen, “Intelligent Fusion of Wi-Fi and Inertial Sensor-Based Positioning Systems for Indoor Pedestrian Navigation,” IEEE Sensors Journal, vol. 14, no. 11, pp. 4034–4042, June 2014.
3. Z. Chen, H. Zou, H. Jiang, Q. Zhu, Y. Soh, and L. Xie, “Fusion of WiFi, Smartphone Sensors and Landmarks Using the Kalman Filter for Indoor Localization,” Sensors, vol. 15, no. 1, pp. 715–732, Jan. 2015.
4. A. Gallagher, Y. Matsuoka, and W.-T. Ang, “An efficient real-time human posture tracking algorithm using low-cost inertial and magnetic sensors,” 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566), Feb. 2005.
5. Y. Jin, H.-S. Toh, W.-S. Soh, and W.-C. Wong, “A robust dead-reckoning pedestrian tracking system with low cost sensors,” 2011 IEEE International Conference on Pervasive Computing and Communications (PerCom), May 2011.
6. M. Oner, J. A. Pulcifer-Stump, P. Seeling, and T. Kaya, “Towards the run and walk activity classification through step detection - An android application,” 2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 2012.
7. K. Park, H. Shin, and H. Cha, “Smartphone-based pedestrian tracking in indoor corridor environments,” Personal and Ubiquitous Computing, vol. 17, no. 2, pp. 359–370, Feb. 2013.
8. P. Truong, J. Lee, A.-R. Kwon, and G.-M. Jeong, “Stride Counting in Human Walking and Walking Distance Estimation Using Insole Sensors,” Sensors, vol. 16, no. 6, p. 823, June 2016.
9. IEEE Technical Committee Report on Recommended Practices for Burst Measurements in the Time Domain," in IEEE No 257-1964 , vol., no., pp.1-12, 15 May 1964
10. ISO/IEC/IEEE International Standard - Systems and software engineering -- Life cycle processes --Requirements engineering," in ISO/IEC/IEEE 29148:2011(E) , vol., no., pp.1-94, 1 Dec. 2011
11. "IEEE Standard for Software Unit Testing," in ANSI/IEEE Std 1008-1987 , vol., no., pp.1-28, 17 June 1987

4.3 Appendices