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

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. 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 recalibration points and reset the users 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 3 individuals following separate trajectories which span through both indoor and outdoor locations (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 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 (Truong et a).

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.

$$\mathrm{RSS} = \mathrm{RSSO} - 10 \cdot p \cdot \log_{10}\left(\frac{d}{d_0}\right) + v,$$

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

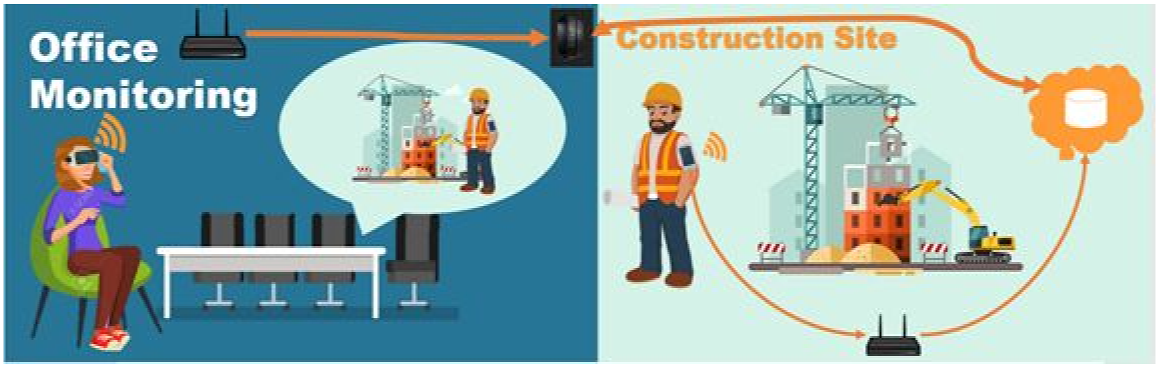

Figure 1. Concept Diagram

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

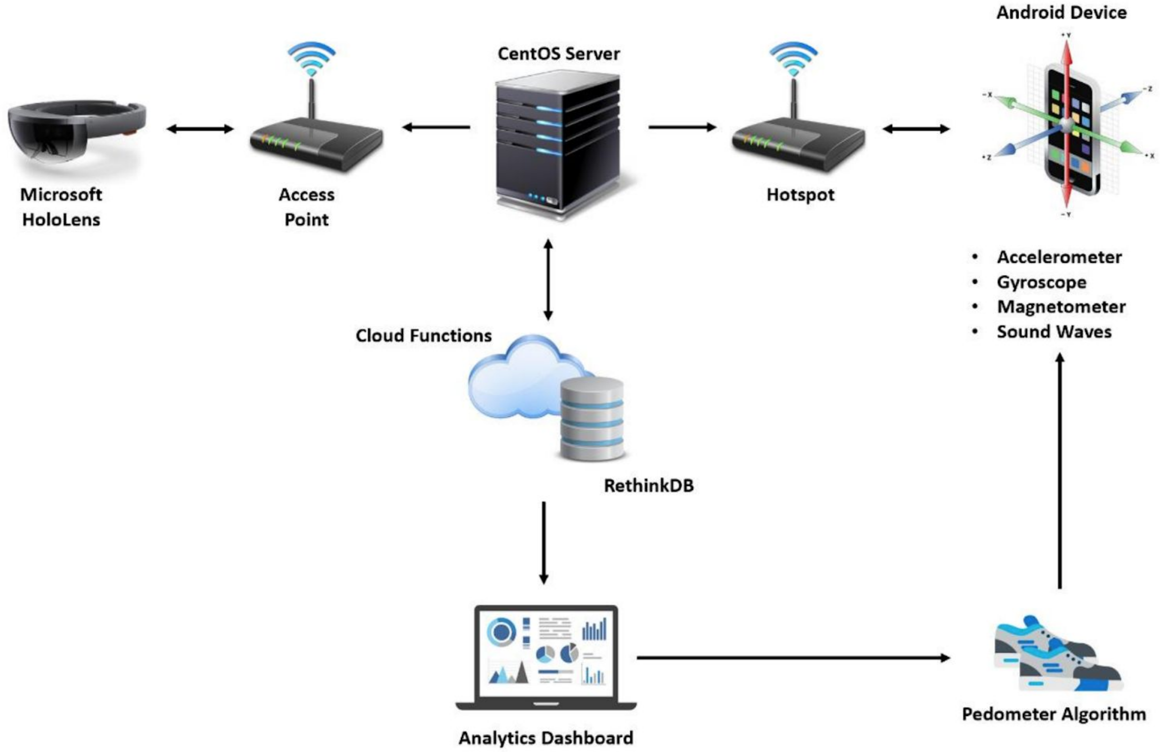

Figure 2. Architecture Diagram

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 Mobile Application

Android OS

Android OS is the world's most popular mobile operating system. The operating system uses the Linux kernel and supports applications written in Java and Kotlin. Android has great documentation and support which makes it easy for new developers to create applications. The primary IDE for Android development is Android Studio.

Android Studio

Android Studio provides tools for building apps on any device running the Android OS. It includes a visual layout editor for easy drag and drop view manipulation, phone emulators for testing applications on different virtual devices, and built in code completion for faster development. In addition to all this, Android Studio also provides real-time CPU, Network, and Memory profiling.

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.

Volley

Volley is an HTTP library that allows Android to asynchronously send and receive data over a network. Our team will use volley to push sensor and location data to the RethinkDB.

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.

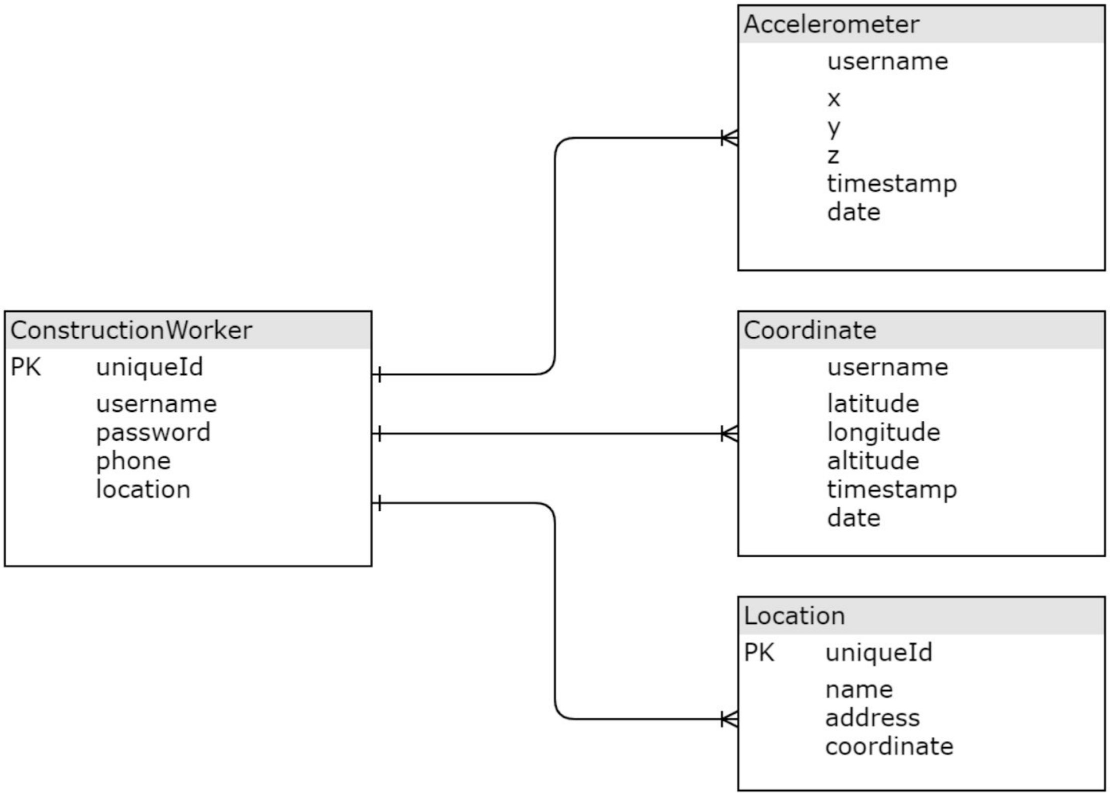

Figure 3. Database Schema Diagram

The database will have 4 schemas:

1. Construction Worker
2. Accelerometer
3. Coordinate
4. Location

Construction Worker Table

The construction table is used to store user specific information. This table will be used to map specific a device to a specific site location.

Accelerometer Table

The accelerometer table is used to store the 3D accelerometer data collected from the phone's sensor.

Coordinate Table

The coordinate table is used to store the x and y coordinates which map to our 2D and 3D visual displays. Mapping all coordinates between a specific timestamp for a given user will display that user's trajectory in the given interval.

Location Table

The Location table is used to store the name, address, and location (latitude/longitude) of each job site. It will contain the location information of all job sites currently being monitored. By cross referencing a Construction Worker's uniqueID with a Location's uniqueID, we can determine which site location a worker is attached to.

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.

Worker/Vehicle Movement Scripts

The following two scripts will respond directly to coordinate table updates in the RethinkDB.

The player control script will be used to move the avatar from its current position to some newly received position.

The vehicle control script will similarly be used to move a vehicle object from its current position to some newly received 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.

Animation Script

The animation script will animate (idle state, walking state, running state, etc.) objects in the HoloLens depending on their perceived state.

Controller Script

The controller input script will allow the supervisor to switch between different zones, workers, or vehicles on the construction site. It will also provide a playback feature for viewing previously recorded data.

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. Essentially, this event will tell the mobile device that we are experiencing error and need to recalibrate the user's location.

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.

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.

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 Durham as a proof of concept. To be able to monitor individuals on the HoloLens, we need a 3D model of Durham to represent our work site.

To locate information about Durham (our work site), our team searched the Iowa State website www.fpm.iastate.edu. The website helped us locate a link to 3d SketchUp models from Trimble Warehouse. This is where we found our model of Durham.

Unity requires a 3d model to be in the format of .obj or .aed. SketchUp Pro offers a converter to convert .skp files to these formats. Once a model file is converted and loaded into Unity3d, it can begin taking the various textures files and applying them to the polygons of the 3D model. The model can then be geolocated to the exact location on the map platform, and rendered in augmented reality.

In order to represent workers and vehicles on the job site, we need to create some vehicle and character models. Unity offers these models in the asset store for purchase.

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.

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.

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 API's.
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. Xamarin is open sourced.

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.

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. 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. Internal storage thus you cannot upgrade the hard drive unless you buy a new device.
3. iOS framework documentation is harder to find then Android.
4. 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.

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.

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.

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

By the end of the project, we expect to have a phone application capable of tracking a person as they move around a space. The accuracy should be within 1 meter of their actual location. The app will be able to store this location and movement data locally when network connections are not available, and conversely send data to the server when network connections are available. From there, the information will be saved in a RethinkDB, and distributed to the HoloLens application.

2.13 Test Plan

Functional Requirement:	**Test Plan:**	**Verification:**	**Validation:**

Indoor/Outdoor User Tracking	We will setup 3 Android devices at different starting locations. We will provide the user with a specific path to walk which includes a start point and an endpoint. We will then record their path and display it on Mapbox, comparing the accuracy of our prediction estimates to the true path.	**Integration Testing:** *The AR team will design unit tests:* 3 avatar game object instances get created when 3 individuals are using the mobile 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 tests 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. After the tests are completed, the tester will do a report analysis of the errors to 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 movements.	*Usability Mobile test:* The user will launch the application, perform a walking movement and a running movement, then report the results. The mobile team will have a walkthrough to verify how this test is going to meet moving sensitivity requirements.	*Tester:* The user will turn on the mobile application, perform running and walking movements, and verify the mobile device detects the differences. After the tests are completed, the tester will do a report analysis of the errors found to be reviewed by the development team.

Store data to RethinkDB	We will connect the mobile device to WiFi and will then perform HTTP requests to save data in our database.	*Mobile unit test:* Verifies that the mobile application does an HTTP request when the device is connected to a WiFi access point. *Backend unit test:* Verifies that the data received from the mobile application matches with the recorded data from the mobile application. The backend and mobile teams will have code walkthroughs and reviews to verify that the unit tests are succeeding.	*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. After the tests are completed the tester will do a report analysis of the errors found to 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 predicted location is within a meter.	*Mobile unit test:* Create a unit test which starts at location A, moves to location B, and verifies that the predicted 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 found to 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 timestamp and compute the difference.	*Mobile unit test:* Create a unit test that takes the timestamp 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 to confirm tests are meeting the coding standards.	*Mobile tester:* Verifies that the delay time of the application is within the 5 second threshold by running the test script and reporting 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.

HoloLens Monitoring	The HoloLens will receive the new recorded location data from the Rethink database. It will then render device positions 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.	*AR unit test:* Create a unit test that takes the newly recorded data from RethinkDB and calculates 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 computed position transform meets the specification of the HoloLens monitoring requirement. The reviewers will look at the code and see if it meets the Unity coding standards.	*AR testers:* Verifies the avatar's new position matches the computed position. After the test is finished, 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

3 Timeline, Resources, and Challenges

3.1 Project Timeline

Fall Timeline

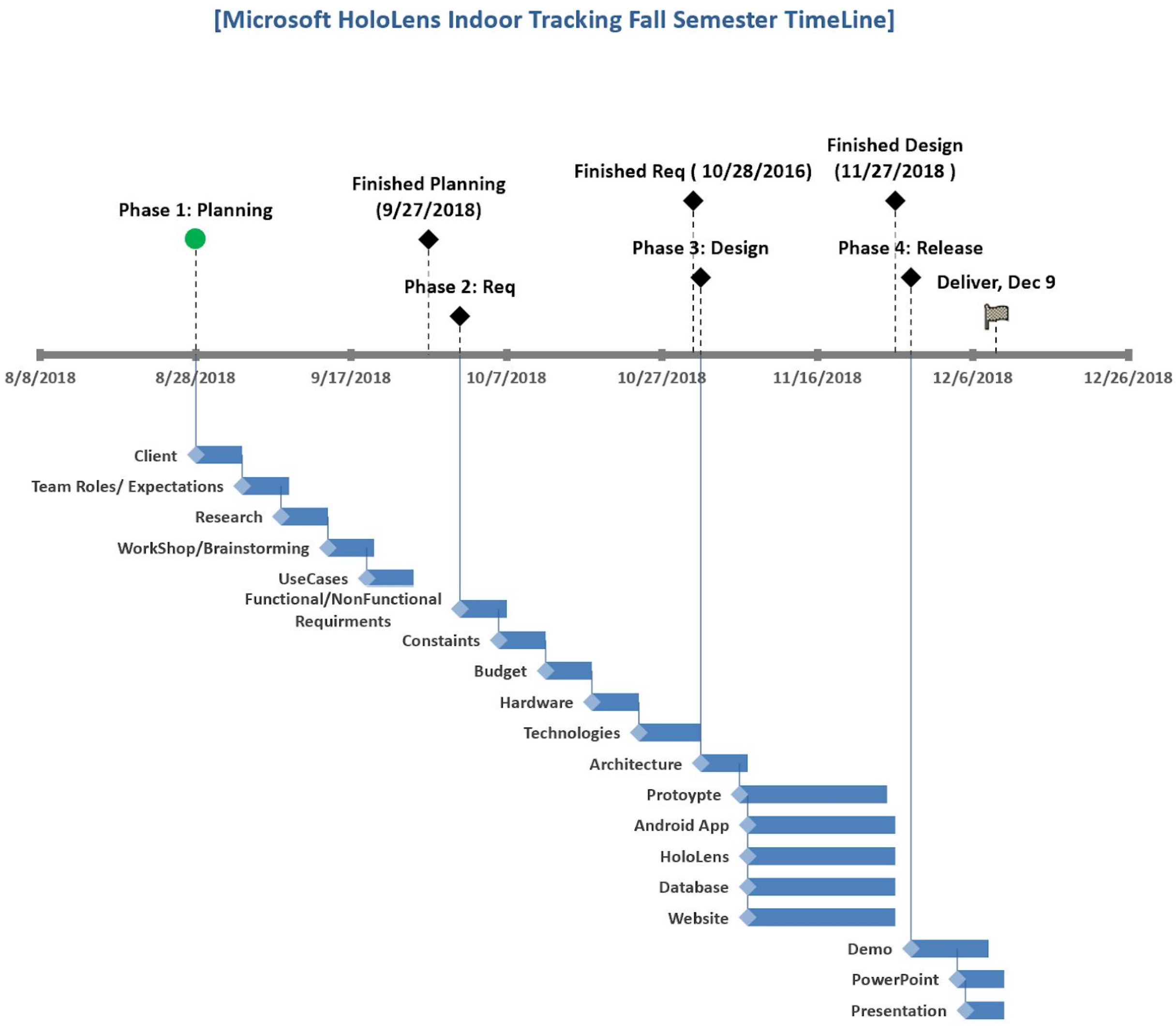

Figure 4. Fall 2018 Timeline

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

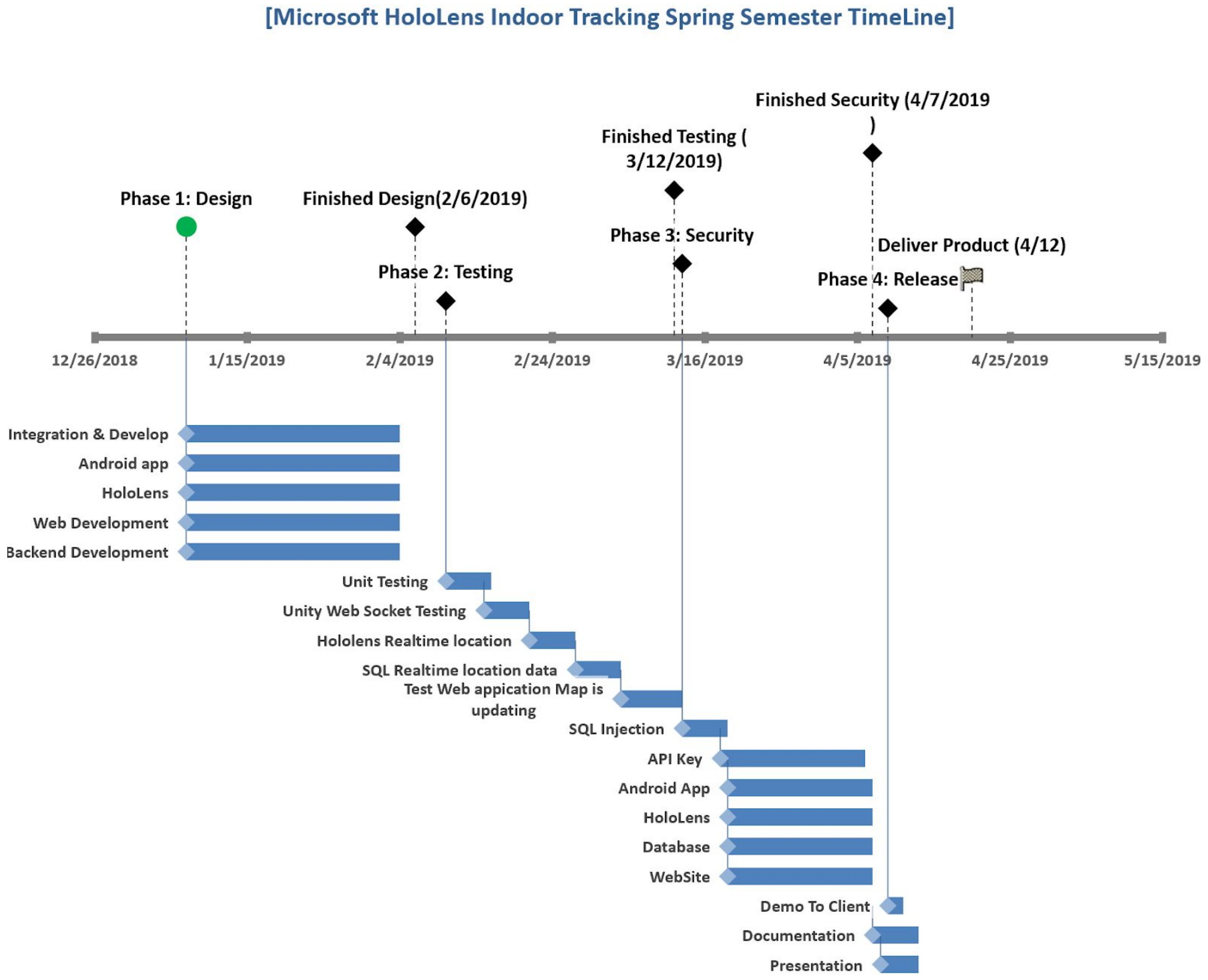

Figure 5. Spring 2019 Timeline

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

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

A. R. Jiménez 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), Sapporo, 2017, pp. 1-8.

Chen, Lyu-Han, et al. "Intelligent fusion of Wi-Fi and inertial sensor-based positioning systems for indoor pedestrian navigation." *IEEE Sensors Journal* 14.11 (2014): 4034-4042.

Chen, Zhenghua, et al. "Fusion of WiFi, smartphone sensors and landmarks using the Kalman filter for indoor localization." *Sensors* 15.1 (2015): 715-732.

Gallagher, Anthony, Yoky Matsuoka, and Wei-Tech Ang. "An efficient real-time human posture tracking algorithm using low-cost inertial and magnetic sensors." *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*. Vol. 3. IEEE, 2004.

Jin, Yunye, et al. "A robust dead-reckoning pedestrian tracking system with low cost sensors." *Pervasive Computing and Communications (PerCom), 2011 IEEE International Conference on*. IEEE, 2011.

Oner, Melis, et al. "Towards the run and walk activity classification through step detection-an Android application." *Conf. Proc. IEEE Eng. Med. Biol. Soc*. Vol. 2012. 2012.

Park, Kwanghyo, Hyojeong Shin, and Hojung Cha. "Smartphone-based pedestrian tracking in indoor corridor environments." *Personal and ubiquitous computing* 17.2 (2013): 359-370.

Truong, P, Lee, J, and Jeong, G. (2016, June, 4). Stride Counting in Human Walking and Walking Distance Estimation Using Insole Sensors. Retrieved from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4934249/

4.3 Appendices