CSE 453/510
Eyewalk System
Indoor Navigation for the Blind
Part 1 – Formal Requirements

Scott Haseley
Brain Feuerstein
Jitendra Rajendra
Deepak Kinni
Peter Turner
Matthew Elling
Mustafa Motiwala
Aditya Lolla
Frank Bayerbach Jr
Ameya Sawant
1. Problem Statement

The problem which we are approaching is indoor navigation for the visually-impaired. Our goal is to construct a mechanism with which visually-impaired people can safely and reliably navigate a building, without the aid of sighted people. We hope to provide blind people with greater independence, increasing their quality of life, and decreasing the burden on their friends and loved ones.

Navigating in an unfamiliar space poses three distinct challenges:

1. Locating a destination
2. Detecting immediate spatial information
3. Avoiding dynamic obstacles

1.1 Locating a destination

When sighted people enter a space for the first time, we primarily use visual cues to guide us in the right direction. If we are headed to a restroom, we look for the sign and walk directly to it. This same principle applies to finding an office, a store in a mall, water fountains, stairways, and exits. In addition, we have immediate access to emergency information. The eyeWalk device must guide someone who can not see to their destination.

1.2 Detecting Immediate Spatial Information

Sighted people know where the walls are, where the exits are, and where any abrupt changes in the terrain occur. We know if we are in an auditorium, in a tight hallway, or anywhere in between. We know if the wall juts out suddenly or if the floor begins to drop. All of this information is readily available through sight, and we depend on it to safely navigate a path to our destination. The eyeWalk device must provide these details to the user.

1.3 Avoiding Dynamic Obstacles

The unpredictability of moving or non-static obstacles provides a greater challenge than the others. While this is a serious problem for blind people, existing technologies, such as the white cane or guide dog, provide an adequate solution. Therefore, in the initial eyeWalk device, dynamic obstacle detection is not a requirement.

Many emerging technologies show promise in tackling the above challenges, but thus far, there is no complete answer. In this document, we will discuss current solutions to the problem and highlight their shortcomings. We will detail what the eyeWalk device must do and how it will “fill in the gaps” left by the other guidance tools. We also acknowledge that no
computerized solution can completely replace the mechanical assistance items like the white cane or guide dog. That is why this device should be used in conjunction with them.

We will end the document by discussing plans that we have for future releases. Our proposed device can provide more than just guidance assistance for the blind, and we will highlight some potential opportunities.

2. Current Solutions

A number of tools exist to help blind people navigate the world independently. Some help blind people avoid obstacles, while others help them find their way. These tools range from animal solutions to computerized solutions, all of which have both merit and deficiencies. As technology gets better, new and innovative solutions are explored, but it is the problem of indoor navigation that needs the most improvement.

2.1 Guide Dogs
Guide dogs can help a blind person become more mobile, avoid obstacles, and remain safe while traveling. However it is the blind person that leads, not the dog. As such, the animal cannot navigate for the person; only help them on their way. Also, extensive training is required for both the dog and the blind person, which may be a deterrent for some.

2.2 White Canes
White canes are another tool that helps in many of the same ways as guide dogs. According to the National Federation of the Blind, “We believe the long white cane is a means to independence. The white cane has proved a useful tool to millions of blind people in navigating their environments with confidence and safety. It is a tool which allows blind people to travel where and when they want, and as such leads to self-sufficiency”. Canes, however, are only able to detect objects in the immediate vicinity and cannot possibly inform their user of what may lie ahead.

2.3 Braille
Some buildings provide Braille on walls and signs to help blind people with indoor navigation. This is a great help when present, but there are problems with consistency. First, you are never sure which buildings have it and to what extent. It can be embarrassing to feel a wall for Braille as people watch knowing that there is none. Second, there is inconsistency in the dialect used when the Braille is present. If there was a certain degree of consistency, the
Braille system would surely make indoor navigation easier for blind people. Finally, the percent of blind people that actually read Braille is rather low – as low as 10 percent according to the NFB.

2.4 GPS
Most technological solutions are focused around GPS. At present, dozens of great mobile applications exist for GPS navigation, but they are mostly isolated to outdoor navigation. For example, Google’s Intersection Explorer can provide a virtual map of your surroundings that can be explored eyes-free. While this without doubt assists in increased self-sufficiency for blind people, it is limited in its reach. Once you are at your destination, the assistance stops.

2.5 Sonar
Research was done in the way of sonar for indoor navigation. The idea was that glasses, canes, or stand alone devices could be equipped with sonar to provide audible clues about a person’s surroundings. Sound would reflect off of objects in a room and could be interpreted by a blind person to learn about their surroundings. The disadvantages to this technology are that it is both expensive and has a high learning curve. While there are some commercially available products available, it is not mainstream.

2.6 Laser Canes
Laser canes are a commercially available tool to assist in navigation. The cane has a laser sensor attached that helps detect obstacles and obstructions and provides feedback through sound or vibration.

2.7 vOICe System
The vOICe System is an innovative and experimental system that converts images from a digital camera (such as a cell phone) into sound. vOICe generates “soundscapes” based on the video images being processed. Panning, pitch, and loudness are used to give audio indication of a person’s environment. The learning curve for this system is very high and it can take months of training to gain anything from the technology. There are, however, people that use it on a daily basis and the idea of the system has enormous potential. Learning software can be installed for free on a Windows PC. http://www.seeingwithsound.com/winvoice.htm

2.8 Google Indoor Maps
As seen here, http://www.engadget.com/2011/11/29/google-maps-6-0-hits-android-adds-indoor-navigation-for-retail/, google has implemented their own indoor navigation system. It uses the GPS and network locator that are present in the phones to track a person's movement on a
map of the floor plan. As quoted from the above link: “The newly added indoor maps don’t quite offer the turn-by-turn navigation you’ve come to know and depend upon”. While google’s indoor navigation system is sure to become a staple of their map services, it doesn't have the fine precision and accuracy that a blind person requires to navigate.

2.9 Summary
None of the existing solutions provide an “all-in-one” approach, tackling all aspects of inner building navigation. Only braille provides details about the type of environment/room, but has serious shortcomings. If braille was ubiquitous, and consistent in content and form, it would be a real solution to this problem. It also provides no help in navigating the physical space. White canes and guide dogs are good solutions to immediate navigation, but have a short range and provide limited feedback. The eyeWalk device will address both of the navigation issues, as detailed in the next section.

3. Requirements – User

3.1 Detecting Objects
The user must be able to detect the presence of any fixed objects by pointing the device toward it and requesting information. Objects must be differentiated by their type. A minimum list of types follows:

- Men’s Bathrooms
- Women’s Bathrooms
- Emergency Exits
- Fire Extinguishers/Emergency Equipment
- Exits
- Stairways
- Elevators
- Escalators
- Benches
- Drinking fountains
- Information Desks
- Known hazards and obstacles
- Walls
- Hallways
- Room Entrances
3.2 Obtaining Directions
The user must be able to acquire directions to any destination on the current map. The directions need to be spoken to the user through the device audio output. Destinations include all objects that are currently on the map. By default, the user should be able to find the physically nearest instance of a particular type with ease. Directions will be delivered in a manner consistent with GPS navigation directions.

3.3 User Interface
The primary method for entering input to the system will be through a cycling audible menu system. The user will select an option by tapping the screen once their choice has been spoken. In addition, the user will also be able to enter voice input to access directions. When the user is in “Find Mode,” they will be able to speak their desired destination and receive directions immediately, without navigating the menus.

All output will be spoken to the user through the device’s headset. Information is given to the user in a controlled manner, and they are never subject to a constant stream of feedback, as this would prove distracting to the user.

The device must distinguish multiple types of taps on the screen: short taps (or single taps), long taps, and double taps. From any mode, the user may enter the main menu by performing a long tap. Menu options are accepted through a single tap. When the user enters a menu, a series of choices will be spoken through the headset. There should be a configurable-length pause interval between choices. The device will perform this cyclically until the user makes a choice. They can accept a choice by tapping the touchscreen.

In addition, the device should also know the dimensions of each room in the building, the total number of rooms and floors in the building, as well as any building-specific information that may be relevant. It should also maintain a brief blurb about the content of the building, whether it is commercial or residential, the address, etc. If it is a public space and has a closing time, this should be included.

4. Requirements – Customer
As our client is the same as our user, these requirements are the same.

5. Requirements – System
The system will be designed for an Android smart-phone, which must meet the hardware requirements detailed in 5.1. It is broken into a few separate components: the device software, the server which distributes the maps, and the software which we use to create the
maps. Upon starting a new session, the device downloads the appropriate map from the
distribution server (the specific map is determined by the GPS coordinates, and can be
helped by the positioning of QR Codes near the entrance to identify the building and the
entrance). The accelerometer in the phone tracks the user's movement as they move through
the building. QR Codes are placed in strategic locations (near entrances, elevators, corners),
to “hotfix” the user's position. This will help eliminate any tracking errors due to the innacuracy
of the accelerometer. The specific functions of the system are detailed in 5.2.

5.1 Hardware
In order to deliver an adequate system, the following hardware requirements are necessary:
• Headset – This is used for all sound spoken to the user. Feedback from the device is
delivered this way.
• Microphone – If the user opts for voice-recognition, this is how they will input their
requests to the device.
• Touchscreen – While the device could be designed with a button-based interface, we
prefer the use of a touchscreen.
• GPS – The device needs to know which building the user is in.
• The device which delivers the software must be:
  ◦ Lightweight – It should not burden the user.
  ◦ Unobtrusive – It should not be too large, awkward, or cumbersome. The user
    should be able to perform their everyday tasks without interference from the device.
  ◦ Inexpensive
Due to the above requirements, Android-based smartphones are the target platform.

5.2 Modes of Operation
The device will operate in different modes, all of which should be selectable through a menu
as described above. The session begins in idle mode, and can enter direction or detection
mode. Direction and detection mode provide different ways to obtain information about the
surroundings. Direction mode provides directions to locations, while detection mode detects
uses a “point and click” style to obtain spatial information. Find mode is an alternative input
mode where the user speaks into the device's microphone and enters “Direction-mode”
requests that way.

5.2.1 Direction Mode
In this mode, the user can get safe passage instructions to any point available in the building
(see information requirements below). The user will select a starting point which is either the
user's current location, a point on the map that is selectable through a spoken list of options,
or a location that the user speaks into the device. The user will then select a destination through a spoken list of options or a destination that the user speaks into the device. The directions will then be spoken to the user like traditional GPS instructions. For example, “Turn Left, Walk 10 feet...” and continues to guide the user as they move.

5.2.2 Detection Mode
In this mode, the user will have the ability to get information about what is around them by pointing the device in a particular direction, perform a tap, and receive audio feedback about what is nearby in that direction. Beginning with the closest item, it will receive the exact distance (in units of measurement) from the device to the item. Any obstacles will be mentioned. The device should give a list of items within 5 feet, and then 10 feet, and so on from that closest item. For example, the devices is pointed in one direction and clicked, and the feedback is “Wall ahead 13 feet. Bathroom 4 feet to the right of the wall.”

5.2.3 Normal Operational Mode (Idle Mode)
This is the mode the system will be in when the user is simply moving about the building. It will track the user’s movement and give alerts based on the user’s preferences.

5.2.4 Find Mode
This is a mode in which the user can request information from the system by speaking into the device. The user will say “Find” and the name of any point in the building and receive audio feedback with directions to the nearest location. For example, if the user says “Find Exit”, the system should speak directions to the nearest exit. This mode should be accessible at any time from the Idle Mode. Any item in the Information Requirements should be available in the Find Mode.

5.2.5 Where am I? Mode
While not a true operation mode, the user will be able to ask the device where they are, and the proper container description should be returned to them. In addition, they should hear the floor number and the building name. Containers are defined as areas of points, such as a hallway, room, or stairway.

5.3 Configuration/Flexibility
The software device will function smoothly and efficiently “out of the box.” Great care will go into choosing the default values and which options are enabled. This is especially important when designing for people who can not see, as their desire to navigate through the depths of configuration and options menus is understandably low. From any screen, the user can access the main menu through a long tap. One option from that menu will be Configuration, which will then provide the options listed below.
5.3.1 Operation Modes (Detection/Direction)
The user will have the option of using either Direction or Detection mode to detect obstacles and find destinations. Their interfaces are very different and they provide complementary information to the user. It will be extremely easy to switch between the two, depending on the nature of the information that the user requires. Detailed descriptions of the different functional modes can be found below, in section 6.

5.3.2 Find Mode (for entering voice)
When enabled, this allows the user to switch to “Find Mode” by providing some simple input (ie. double tap) from any previous mode. While in Find Mode, the user will speak “Find <destination>” and they will immediately switch to direction mode and receive directions to the nearest destination of that type. It mainly functions as a quick shortcut to find the nearest destination which suits their needs.

5.3.3 Screen Output Mode (for visual output)
The visual output shows the user move through the space. They will be in the center of the screen, and the surrounding obstacles will move and change dynamically as they navigate. Since the blind population has little use for visual output, this will be primarily useful to help in debugging the software, but may also be useful for low-vision users. The distance which is shown on the screen can be configured here.

5.3.4 Multiple Languages
The default language will be English, and that will be the only language in our initial release. Our first subsequent version, however, will support Spanish at the very minimum. We hope to add many additional languages quickly, if there is demand for it.

5.3.5 Unit of measurement (Feet vs Meters)
Different users will have a preference for the units that distances are measured in. The default will be the American “feet” and “inches,” but there will be support for metric, as well.

5.3.6 Voice Style for Feedback
The user will be able to choose what the voice sounds like which provides all of the output. There will initially be two voice, one male, and one female. More can be added down the line, if there is sufficient demand for variety.

5.3.7 Alerts
When alerts are enabled, the user will be notified when they are within a certain distance of
some obstacle or destination. These are turned off by default and are fully customizable. The user can dictate which types of destinations they receive alerts for, what the distance is when they receive the alert, and how they are notified of the alert. They also can configure the time between alerts, so they aren't continuously warned every 2 seconds. This could be very useful as a warning if the user enters a “buffer zone” near walls or other hazards. Safety Alerts could also notify the user through vibration (also configurable).

5.3.8 Building-Specific Obstacles/Destinations
While there will be a set of destinations and obstacles that are common to most buildings, some will require custom data. When we create a map, we will have the opportunity to add any type of obstacle by providing the proper parameters. (The Destination/Obstacle data type is detailed below, in section 7).

5.3.9 Configuration Defaults
Alerts: disabled; Time between Alerts: 1 min; Notification Method: Vibrate Only; Alert Destinations: Walls (all other destinations disabled by default) Voice: Woman; Language: English; Measurement Unit: Feet; Display Distance: 20 ft.

6. Testing

6.1 Testing Methodology
In order to accurately test the software, we will employ an iterative design methodology. The complete system is broken down into the following parts:

1) Device Software
   1. Direction Mode
   2. Detection Mode
   3. Accelerometer Movement
   4. QR Code location hot-fixing

2) Map Making Software

Each component will be constructed and tested independently of the system. We will then integrate one component at a time, and perform a complete system test. By only integrating one component at a time, and exhaustively testing the software after each integration, we should be able to easily trap, isolate, and fix errors before they get lost in the system.

6.2 Unit Tests
Each software component will undergo a series of tests specific to the functionality and
features of that component.

6.2.1 Direction Mode Testing
To test direction mode, our primary concern is that the directions provided to the user are accurate. Inaccuracies can be classified as follows:
   1) Guiding to the wrong destination
   2) Providing incorrect distances per each “leg”
   3) Providing incorrect angles per each “leg”
We will provide test cases, with predetermined distances and angles, and then run the system on those cases and identify inaccuracies. In addition to inaccuracies, we also need to answer the following questions:
   1) If no destination of the desired type is present, does the software handle this gracefully?
   2) If more than one destination is present, is it easy and clear which choice the user is making?

6.2.2 Detection Mode Testing
In order to test detection mode, we need to make sure that we are picking up the proper item from the map, and that the information that is provided is complete and accurate. In particular, we need to make sure of the following:
   1) We test all angles to the object, including the border cases, (0 and 180 degrees), head on (90 degrees), and a few in between.
   2) We ensure that objects can not be detected through walls.
   3) All data is read properly, so that the correct description, type, and distance is provided to the user.

6.2.3 Accelerometer Testing
We will be using the accelerometer to track our basic movements, although we do expect some error to accumulate due to inaccuracy of the sensor. Our tests need to make sure that we are doing our best to minimize these errors, although the QR Codes are designed to be a strong backup to location tracking.

6.2.4 QR Code Testing
QR Codes will be placed at key locations within the building, and we will be using a camera to detect these codes, read them, and compute our location and facing-angle based on the location of the codes. We need to make sure that these codes are 1) Detected under all
appropriate conditions, and 2) Delivering complete and accurate information.

To test that the codes are detected under all conditions means we need to test the camera moving past at different speeds, from a very slow movement, to a brisk walk. We need to test that the camera can read the codes from all angles, from 0 to 90 to 180 degrees, and points in between. We also need to make sure the camera can detect codes from distances very close and very far (up to 20ft).

Since these codes are only used for movement tracking and not for obstacle detection, we are only concerned with the following data being conveyed properly: x position and y position.

Finally, to prove that the QR Codes work perfectly, we need to make sure that the position is properly updated based on the information gathered from the codes. This will be tested simultaneously with the tests for speed, angle, and distance of codes.

6.2.5 Map Maker Testing
Our software to create the maps needs to be tested for the accuracy and completeness of the maps that are created. Because the data and measurements in the maps come directly from floor plans, we can test this in two phases.

Phase 1 – Test that the floor plans are accurate to the actual buildings. This is not something that we will do for all floor plans, but it will give us a starting point to find errors.

Phase 2 – If the floor plans are accurate, then we need to make sure that the data in the floor plans is translated accurately through the map generating software. We can do this by testing the device software on our newly created maps, to make sure that all objects, walls, and hallways are spaced appropriately. We also need to test that the map aligns properly with the north axis, as is required by the compass sensor in the device, that the scale is set properly, and that the maps are linked appropriately when the user moves between floors.

Our maps are going to declare certain areas to belong to certain containers (ie Hallway, Room 1, etc), and so we need to test that the container data is always accurate.

6.3 Integrated System Testing
Once all components are tested individually and tested on integration, we will perform a series of complete system tests. Each of these will take the system from beginning to end, testing the accuracy of all of the components. Testing begins by starting the application, and
making sure that the user is placed at the correct location. As the user moves through the map, we will observe the registered position, and make sure that any error accumulated by the accelerometer is handled by the QR Codes. We will test detection on each obstacle in the map, from different angles, and make sure that all of the QR codes that we encounter are registered. We will vary the movement past the codes to test speed, distance, and angle. We will request directions to each location on the map, and we will vary the starting positions from inside of hallways to rooms, to borders of the map. Any errors that we detect will be recorded and addressed before we move on to the next test.