UC Santa Cruz

UC Santa Cruz Electronic Theses and Dissertations

Title

Robust Path Back-Tracing Guidance System for Blind People

Permalink

https://escholarship.org/uc/item/6nq496gw

Author

Flores, German H.

Publication Date

2017

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

UNIVERSITY OF CALIFORNIA
SANTA CRUZ

ROBUST PATH BACK-TRACING GUIDANCE SYSTEM FOR BLIND PEOPLE

A dissertation submitted in partial satisfaction of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER ENGINEERING

by

Germán H. Flores

September 2017

The Dissertation of Germán H. Flores
is approved:

Professor Roberto Manduchi, Chair

Professor Sri Kurniawan

Professor Katia Obraczka

Tyrus Miller
Vice Provost and Dean of Graduate Studies

Copyright © by

Germán H. Flores

2017

Table of Contents

List of Figures

List of Tables

Abstract

Robust Path Back-Tracing Guidance System for Blind People

by

Germán H. Flores

Indoor positioning and navigation is an active area of research due to the widespread use of devices equipped with micro-machined electromechanical systems (MEMS) sensors such as gyroscopes, accelerometers, and magnetometers. The MEMS inertial sensors and other popular technologies can provide information to determine the position and orientation of a person relative to a known initial location. A system that is able to produce accurate location data may find multiple applications in location-based services (LBS), safe wayfinding, and other related fields. For instance, a guidance system could be used to provide travel-related information to passengers as they use public transportation, or provide safe navigational directions to people who find themselves in unfamiliar environments. The research presented in this dissertation focuses on building robust systems that provide real-time and reliable travel-related information to help blind or visually impaired travelers reach a destination safely.

In this dissertation, I present two novel navigation systems and an openly accessible and annotated data set of inertial sensor time series data collected from blind people. The first system conveys travel-related information to blind passengers when using public transportation. The system makes use of pre-configured Wi-Fi access points placed in public transit vehicles and at bus stops to convey real-time, multi-modal travel-related information to any passenger, directly on his or her own smart device. The second more complex system helps blind people retrace the path taken inside a building and walk safely back to an initial location. This is ideal for situations in which a blind person is able to reach a certain

location, for example with the assistance of a sighted guide, and needs to find his or her way back to the initial location. This robust path back-tracing guidance system is comprised of a turn detector based on a hidden Markov model (HMM) to robustly detect turns even in the presence of drift in the inertial measurements and noticeable body sway during gait, a step counter that uses filtered inertial sensor data to determine the number of steps walked along a path, and a path matching algorithm to track the user's location. The step counter and turn detection models were trained on sensor data from WeAllWalk, an openly accessible and annotated data set of inertial sensor time series data created from blind walkers using a long cane or a guide dog. This system runs on a smart device and provides the guidance necessary to help a blind person retrace a path. A robust guidance system that supports safe blind wayfinding opens the doors to many blind travelers who would like to travel and move independently as they explore unfamiliar environments.

Dedicated to my parents, sister, and family for all their love, support, and many sacrifices.

Acknowledgments

I would like to express my sincere gratitude to my advisor Prof. Roberto Manduchi for his continuous support of my Ph.D. study, his motivation, support, advice, guidance, and encouragement. His guidance helped me throughout my research and prepared me for the many wonders of the future. Thank you!! I would also like to thank my dissertation committee: Prof. Sri Kurniawan and Prof. Katia Obraczka for their support, encouragement, and advice. I would like to thank my family: my parents, sister, bro, and family members for their support, advice, and love throughout my school career. And all those who participated in my research, thank you for all your time and feedback.

Thank you.

Germán H. Flores

Chapter 1

Introduction

1.1 Motivation

Navigating through unfamiliar outdoor or indoor environments can be unsafe, challenging, and a time consuming task for those who don't have access to maps, GPS-enabled devices, or are unable to see. When visiting a new building, city, or simply going to a hospital or the doctor's office, we all have learned to use maps to search for landmarks to pinpoint the current location and provide a sense of orientation with respect to a landmark, use smart devices equipped with GPS that provide turn by turn directions to get to a location of interest, and sometimes rely on asking other pedestrians to provide guidance or enough information to get to or near a destination. Although these approaches and many others have helped many people get to a destination, these approaches can be extremely challenging and unsafe for persons who are visually impaired. Blind travelers cannot recognize visual landmarks, cannot preview visible portions of a route, and cannot access visual maps. While some blind individuals are able to build fairly precise spatial representations from direct locomotion experience [88], others can develop only a limited understanding of the environment during route traversal

[38]. Technological solutions that can support safe blind wayfinding may be very attractive for increased mobility and independence for people who are blind or visually impaired.

One of the popular areas of research due to the widespread use of devices equipped with micro-machined electromechanical systems (MEMS) sensors such as gyroscopes, accelerometers, and magnetometers is inertial navigation [93]. In inertial navigation, the MEMS inertial sensors provide information that can be used to determine the position and orientation of a target relative to some known initial location. Other systems have also been surveyed [28, 7] and require the use of specialized technologies (e.g., RFID readers, IR transmitters, Bluetooth beacons) and knowledge of the physical environment to determine the location of a target. A reliable navigation system could be built around any of these technologies with the purpose of providing efficient and safe navigation directions to people who find themselves in unfamiliar environments. For example, the system could tell people at a mall how to get to a particular department store or could tell them how to get to the nearest exit in case of an emergency. In other cases, a navigation system could be used to provide travel-related information to passengers as they use public transportation. And in more serious cases, a reliable navigation system could help firefighters exit a burning building when the visibility degrades due to smoke and other burning chemicals. The system could tell them where they are in the building and how many steps and turns to take to exit the building. There are many cases in which a robust navigation system could aid in creating a partial mental map of the environment and help navigate it until a person finds himself or herself at a final destination of interest and out of harms way. While most people may benefit from these systems to travel to an unknown

remote location or find a particular destination, they have tremendous potential to enable way-finding and safe navigation by people with visual impairments.

The World Health Organization (WHO) estimates that at least 285 million people in the world are visually impaired: 39 million are blind and the rest have low vision [72]. Current indoor navigation research has utilized geospatial information to provide virtual boundaries for the navigation trajectories [7], while others have used tracking algorithms that use map information from OpenStreetMap and particle filters [43]. Both applications make use of smartphone sensors and other available information such as Wi-Fi or beacon (Bluetooth Low Energy radio transmitter) readings. While these applications have focused primarily on pedestrian indoor positioning, some indoor navigation work has been done for the blind or visually impaired community [34, 78, 6]. A few accessible positioning systems are already available on the market, including accessible GPS (e.g., the Seeing Eye GPS from Sendero Group) [66] and Wi-Fi positioning (e.g., the AXS system by EO Guidage). A localization system for blind travelers developed by indoo.rs and based on iBeacons is being tested at the San Francisco airport. These self-localization systems aim to measure the precise position of the walker and match it against a map of the environment. These systems usually require the use of an existing framework and map of the site in order to localize a person. However, it might be possible that a simpler topological description of a path in terms of turns and steps may be sufficient to help a blind person navigate in unfamiliar environments.

1.2 Contributions of this Work

This dissertation describes the research completed to develop robust systems that provide real-time and reliable information to blind or visually impaired trav-

elers directly from their smart devices (e.g., iPhone, smart watch, or tablet). In particular, two systems were developed to overcome many of the challenges faced by the visually impaired community when traveling to unfamiliar outdoor and indoor environments. In the first contribution, Ch. 2, I describe a novel system developed for conveying travel-related information to blind passengers when using public transportation. A user study was conducted to evaluate the system, and observations from the experiments and interviews brought to light a number of accessibility issues that need to be addressed when developing a personal navigation system. Personal navigation systems must be able to provide users with accurate spatial and directional information, and must be able to deal with adverse situations (e.g., getting lost, taking the wrong route, or avoiding obstacles). Inspired by this contribution, my focus was directed into developing a robust personal navigation system for indoor environments that would help a blind person retrace a route in any building without the need of specialized hardware or map of the building. In developing an indoor navigation system, where localization is difficult due to the lack of GPS signals in a building and where obstacles (chairs, tables, garbage bins) are inherently present, we must also take into consideration that a person may not have a steady gait, and, if the person is blind, he or she may often exhibit body motion patterns that are very different than those of a sighted person. In addition, the use of a mobility tool (white cane or guide dog) may result in extra upper body rotation and gait differences that may affect the accuracy of a guidance system. To address these problems, the second contribution, described in Ch. 3, introduces an openly accessible and annotated data set of inertial sensor time series data collected from blind people. The primary purpose of creating this first-of-a-kind data set was to be able to (1) learn about the body motion patterns exhibited by blind people, and (2) use this sensor data to benchmark and train

models that could be used for an indoor or outdoor inertial navigation system. Finally, I introduce the robust path back-tracing guidance system, described in Ch. 4, that consists of a turn detector to robustly detect turns, a step counter to count steps, and a path matching algorithm to determine the user's current location. I hope that the research presented in this dissertation inspires the next generation of wayfinding systems to help blind people safely navigate indoor and outdoor environments.

Chapter 2

Public Transit Assistant for Persons with Visual or Cognitive Impairments

This chapter describes a novel system developed for conveying travel-related information to blind passengers when using public transportation. A user study was conducted to evaluate the system, and observations from the experiments and interviews brought to light a number of accessibility issues that need to be addressed when developing a personal navigation system.

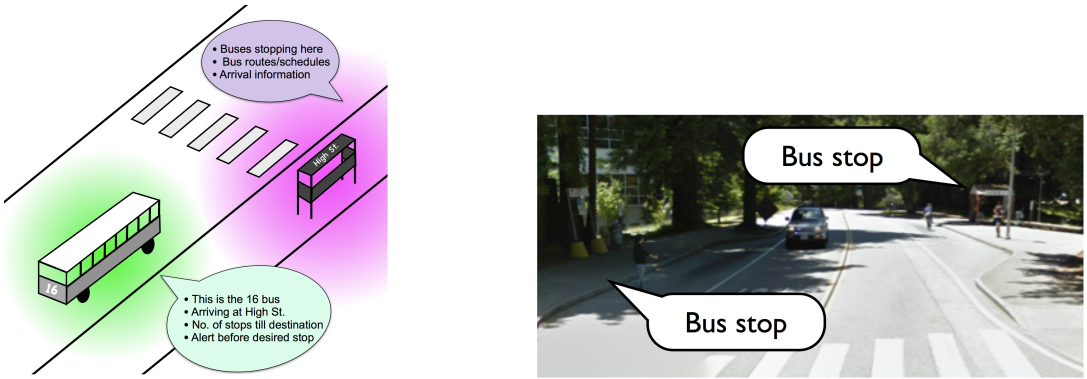
2.1 Introduction

Public transportation is key to independence for those who, for various reasons, cannot drive. At the same time, independent use of public transportation can be challenging for large portions of these individuals. For example, individuals with cognitive disabilities may have problems organizing and executing independent trips [18, 92]. Individuals with visual impairments are also at a disadvantage when

taking public transit [56, 8, 9]. A blind person cannot access printed information at a bus stop; cannot read the number of a bus arriving at a bus station; and, once on a bus, may miss the desired stop if the bus driver does not call all stops, or if the ADA-mandated audible announcement cannot be heard, for example, due to loud background noise. These problems were highlighted by a survey conducted jointly by the LightHouse for the Blind and the City of San Francisco in 2007 with more than 50 blind passengers [56].

We developed a novel approach for conveying real-time, customizable, multi-modal travel-related information to any passenger, directly on his or her own cell phone. Unlike previous research addressing a similar problem [10, 92, 8], our system does not require access to the Internet, and thus does not demand subscription to a data plan. Information is pushed to one's cell phone from Wi-Fi beacons that are placed in public transit vehicles or at bus stops. In addition, users of this system do not need to rely on GPS data from their cell phone (as, for example, in [85]). Note that GPS data can be inaccurate or unavailable in some situations (e.g., signal obstruction due to trees, tunnels, or buildings), and continuous GPS usage quickly drains a cell phone's battery.

Our Public Transportation Assistant (PTA) consists of placing Wi-Fi beacons at bus stops as well as within bus vehicles (see Fig. 2.1a). Upon arriving at a bus station, users of this technology receive an acoustic signal from their cell phone, indicating that a connection with the local beacon was established. At that point, the user can interrogate the system to obtain information about that bus stop and about incoming bus vehicles. As soon as the desired bus arrives, a new connection is established with the in-bus beacon and maintained while the passenger is riding the bus, and more information, this time related to the specific bus ride, is made available. For example, the user is informed well in advance



(a) **(b)**

Figure 2.1: (a) Conceptual system representation. Wi-Fi beacons are placed at a bus stop and inside a bus, providing different types of information to a user carrying a client software application in a mobile device. (b) The location of the bus stops considered in our experiments in the UCSC campus.

when the bus is approaching a desired bus stop, in time to get ready to exit the bus. Information is presented to the user in the form of synthetic speech. At any time, users can interrogate the system to hear the latest announcement as well as any other available information.

In addition to reporting the technical development of the system, we also report on a user study we conducted with four blind participants who operated our system while traveling by bus through a specific route in our campus. Each participant used our PTA system in a very realistic scenario, which involved catching two bus vehicles equipped with an AP, each time selecting a specific destination, and exiting at the correct stop. After the experiment, each tester participated in a semi-structured interview that focused on his or her experience with the public transit system (in particular, any accessibility issues) and on his or her opinion of the system they just tested. These interviews, together with observations from the experiments, shed light on the major problems faced by blind travelers using public transit, and provided a critical assessment of the functionalities of our PTA system.

2.2 Related Work

Improving information access for cognitive or sensorially impaired travelers is the object of active research [8, 41]. Other systems and applications for the assistance of blind travelers taking public transportation have been described in the literature. For example, Ubibus [9] was designed to help a blind person catch the correct bus. Likewise, the Bluetooth-based application described in [59] ensures that the user is informed when a bus is arriving at a stop. Previous work [86] used Bluetooth beacons placed at bus stops to alert the user about the arrival of a desired bus. Use of Bluetooth beacons was also considered in [15] to provide information to a blind pedestrian about the status of a traffic light. The Travel Assistance Device (TAD) [10] and the app described by Silva et al. [85] both rely on the GPS in the user's smartphone to determine the user's position and to alert the user when the bus is approaching the desired bus stop. Unlike this previous work, our system is designed from the ground up to assist users throughout the whole travel, from the time they arrive at the initial bus stop until they reach their destination. Hara et al. [42] describe a crowd-sourcing approach to building a database with the layout descriptions of bus stop locations. This information can be extremely useful to blind passengers, as also noted in our interviews.

2.3 System Description

The system consists of two main units: a client and server as shown in Fig. 2.2. The server is the access point application that communicates with the client to transmit relevant bus information. The server was designed to transmit specific information when a client is within its transmission range and the user has requested information. Users of this system interact with a client application,

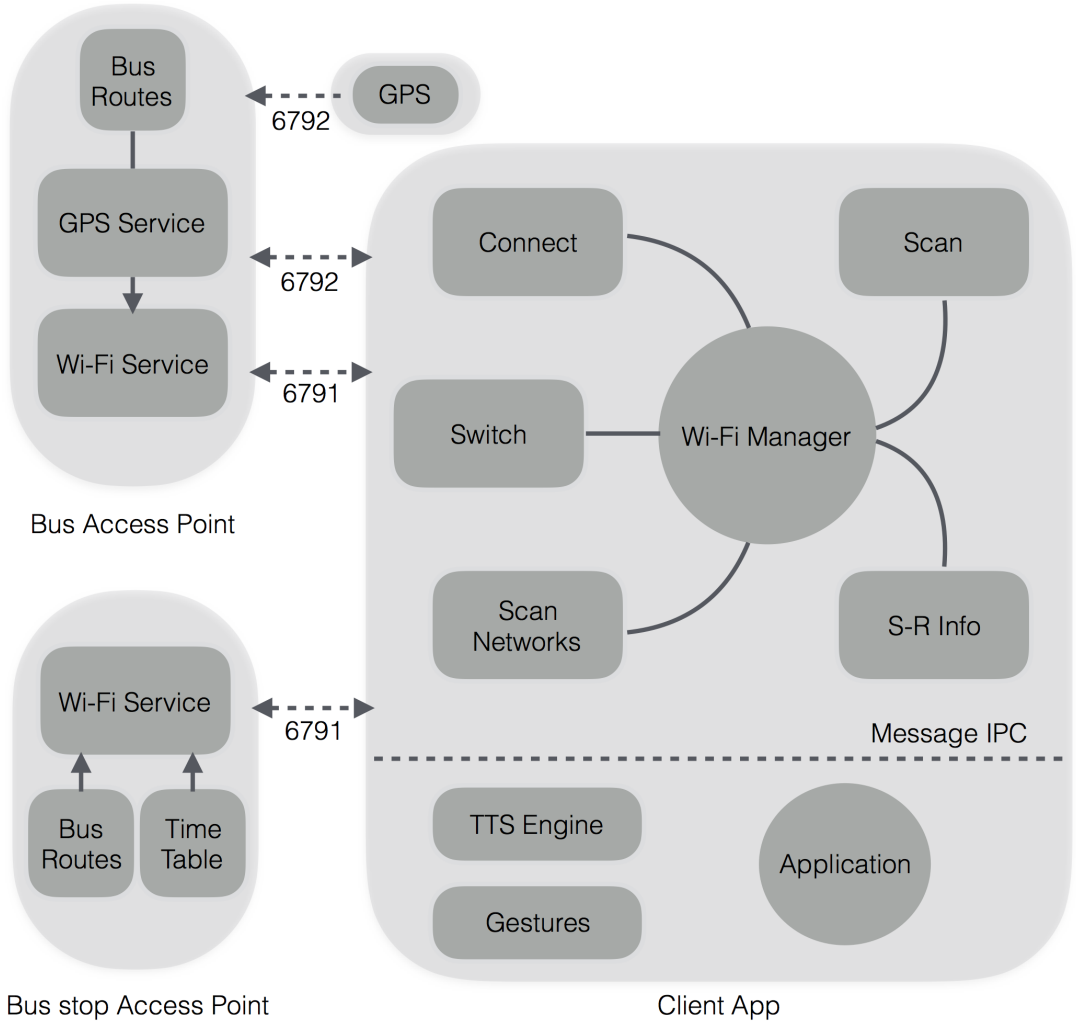


Figure 2.2: Client and Server system architecture. The client components are all implemented and deployed in an Android device, whereas the server components are all implemented in a router.

written in Java and implemented in an Android mobile device (a Nexus 7 tablet). Both modules, client and server, were designed and implemented to be modular, responsive, and intuitive, allowing the user to receive relevant information when desired in a convenient modality.

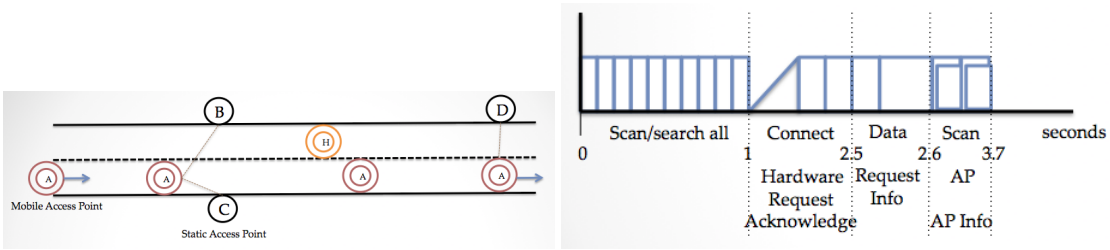
2.3.1 Server

In our current system, Wi-Fi beacons are implemented in TP-LINK routers re-programmed with OpenWrt, a Linux-based operating system that provides a writable root file system with package management and other configurable scripts and tools. Routers are configured as Access Points (APs), and a global static primary IP address is hard-coded into each of the APs.

For our current prototype, two types of APs were reprogrammed and reconfigured: a bus stop AP and an in-bus AP. These APs look and work exactly in the same way, except for the type of information sent to the client. Bus stop APs, which are placed at bus stops, send bus routes numbers and other information such as the address or the name of the bus stop. In-bus APs, which are placed inside a bus, send the bus identifier and information about the bus stops encountered in the route. The information sent by the APs helps the client recognize the type of AP that is within range and perform adequate actions such as prompting the user to select a bus or alerting the user that their selected bus is within range and about to arrive. Note that an in-bus AP traveling on a bus may come within range of other bus stop APs (located on one or both sides of the street), or other moving in-bus APs (see Fig. 2.3a). The client must be able to differentiate between these situations and perform appropriate actions. The information exchanged must be short and compact to allow for fast lossless data transmission even when several users may be using the client application at a bus stop or inside a bus at the same time.

2.3.2 Client

The client is an Android application written in Java that incorporates the following hardware and software technologies: Wi-Fi, touch and gesture detec-



(a) **(b)**

Figure 2.3: (a) Possible mobile and static access point configurations. The black circles, B, C, and D, represent access points that were placed at a bus stop. The red and orange double circles, A and H, represent access points that have been placed in a bus. Moving from left to right, a bus access point can be by itself, come near two bus stop access points, come near another bus access point, or arrive at a location where only one bus stop access point is present. (b) Sequence diagram of events to scan, connect, and send data between the client and the server.

tion, text-to-speech (TTS) engine, socket and message communication protocols, database management system (DBMS), and Android services. Upon arrival at a bus stop, the user and the system must perform several actions that include: Start the Android application; scan and detect nearby access points; provide guidance to the user in order to select a bus stop and a bus to board; provide information about the bus routes and arrival time; detect all of the user's touch screen inputs: gestures or single and double taps; query of the local database to retrieve bus arrival information; provide the user with auditory feedback; listen for in-bus access points; connect, disconnect and switch between bus stop and in-bus access points and vice versa; stay connected to an access point; listen for other bus stop or in-bus access points. Some of these actions must occur asynchronously and without interrupting other actions that are concurrently working or about to start. In order to fulfill this requirement, the client unit was subdivided into five main subcomponents: user activity, Wi-Fi manager service, schedule and instructions service, gesture recognition engine, and text-to-speech engine. The Wi-Fi manager service is the most complex component since it has to manage several

threads (actions) and must be running at all times in order to listen, connect, and disconnect from access points that are within range.

These components must follow an order of execution determined by the user activity. For example, connection and initial communication with an access point must occur before transmission of data. Fig. 2.3b shows an example of a sequence of phases leading to a connection. In this scenario, it took exactly 1 second to find an access point, 1.5 seconds to fully connect to the access point, about 0.1 seconds to transmit handshake data, and 0.5 seconds to scan for selected access points. In general, the scan, connect, and data transfer phases take shorter times than in this example, depending on multiple factors including the distance to the access point and the presence of nearby obstacles.

2.4 User Interface

The client interface is designed to be effective at communicating proper instructions to blind users, guiding users to their desired destination, and providing intuitive usage modalities. It uses multi-touch interaction techniques, text-to-speech, and tactile feedback. The user inputs data through single and double screen taps; simple instructions and information are facilitated via speech; and verbal "Yes" or "No" words or non-speech sounds are used to provide feedback as the user single or double taps the touch screen.

Interaction with the system occurs in two main situations: when the user arrives at a bus stop and wants to be informed once a desired bus has arrived; and when the user is in the bus and wants to be informed about when to exit the bus. A typical information exchange in the first case would proceed as follows: Upon arrival at a bus stop, the client automatically detects nearby APs and iterates through each one of them, prompting the user to select one if multiple

bus stop APs have been found. Once selected, the AP at the bus stop transmits relevant bus information such as the AP location and bus routes. The system then prompts the user to select one of the bus lines that he or she wishes to board. Once the bus line has been selected, the system listens and waits until the selected bus comes within range, at which point the system disconnects from the bus stop AP, connects to the bus AP, and alerts the user that the bus is arriving. In the second case, when the user is in the bus, the system is already connected to the bus AP. The system provides periodic updates such as arrival time, next stop name, confirmation of arrival, and it allows the user to inquire about the current route.

The set of instructions and confirmations that are used to guide the user during interaction with the system were implemented in a hash map structure to allow for fast and easy retrieval and expansion. The client application sequentially iterates through these sets or dictionaries as a state machine, moving from a state to the next, and speaking the correct instruction, question, or confirmation given the current state of the system. For example, when the user opens the application, the system greets the user by speaking "Welcome", and then it asks the user "Do you wish to connect to network X?" Depending on the user response, the system provides a proper confirmation such as "Bus N has been selected" or "The arrival time is ..." At any given state of the system, a phrase or word is retrieved from the dictionary, parsed to fill in any unknown information such as the bus number or the network name, and then sent to the TTS engine, which speaks it. Figure 2.4 shows the ordered sequence of events that must take place from the time the user starts the application to the time the user arrives at the destination.

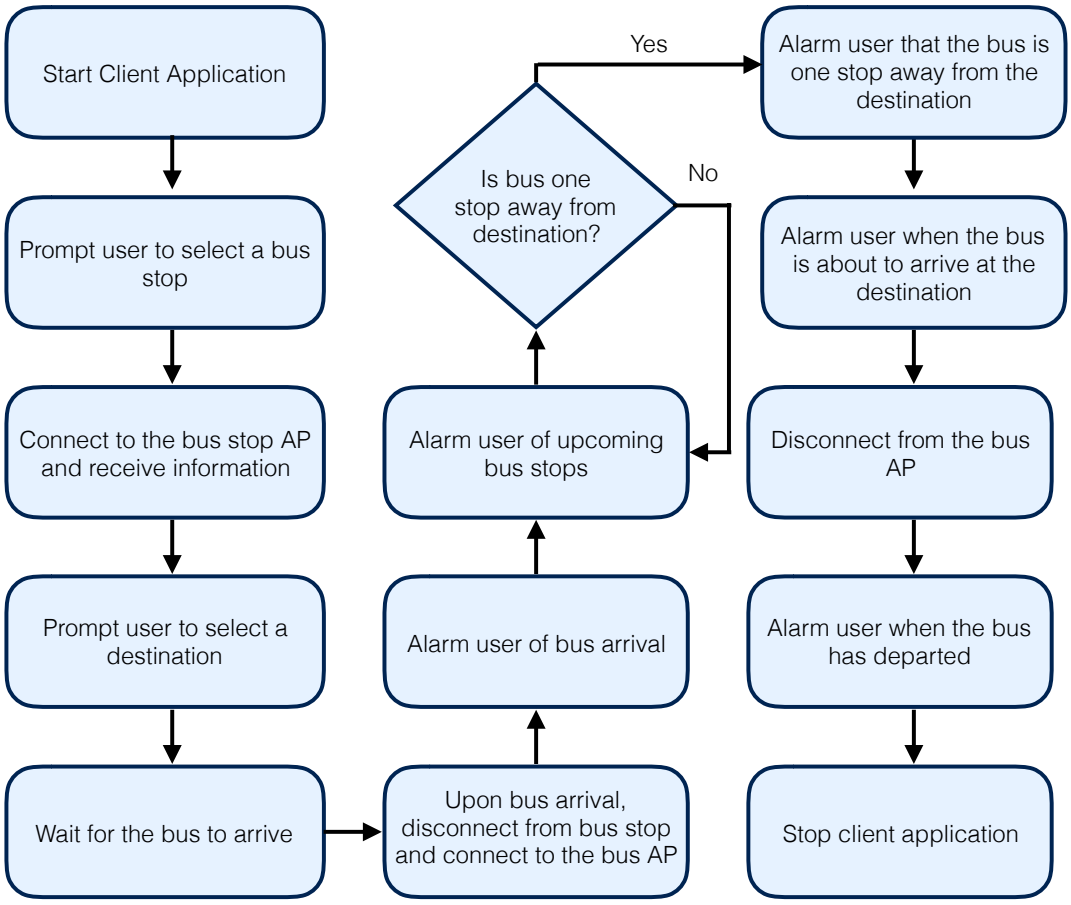


Figure 2.4: Client application flowchart showing the sequence of events taken by the user to reach the final destination.

2.5 Experiments

To test the system and its various components under different conditions, we conducted a total of 41 trials comprising the following scenarios: (1) Walking from and to an access point in open and busy areas; (2) remaining in the bus for at least 20 minutes; (3) testing the system for multiple situations at a bus stop.

Scenario 1. An AP was placed in open and in busy areas, with the user carrying the client system walking away and towards the access point. Open areas are areas with no buildings or large objects present (e.g., an open field). Busy

areas are characterized by buildings and large objects surrounding the access point (e.g., a street surrounded by houses and trees). In both cases, we investigated any Wi-Fi connectivity issues that may occur due to the obstruction or disruption of the radio signal, and obtained estimate ranges for access point discovery and connectivity.

Scenario 2. We placed the AP inside a bus, with the user remaining in the same bus while connected to the AP for extended periods of time. These tests were designed to ensure that the application remained connected to the access point continuously in realistic situations.

Scenario 3. We tested the system in four different situations with a potential user arriving at a bus stop, and with a bus subsequently arriving at the same stop. More precisely, we tested (a) connection to a bus stop AP upon user arrival to the bus stop, (b) connection to a in-bus AP when no bus stop AP was present, (c) switch from the bus stop AP to a in-bus AP upon bus arrival, and (d) connection to a in-bus AP as the user entered the bus, then remaining connected to the in-bus AP for the duration of the trip. In all of the situations mentioned above, the system was tested for discovery and connection time, connection switching performance, and transmission range. A single router, shown in Fig. 2.5a, was placed at the bus stop, while another router was placed in a shuttle bus that came to the bus stop every 20 minutes. (The UCSC Dept. of Parking Transportation and the UCSC Police were notified in advance of the experiments.) The "open area" tests were performed at the UCSC West Field (a large open field without trees), while the tests in the "busy area" were performed at the Science Hill bus stop (Fig. 2.1b). Transmission range measurements from and to the access point were recorded for both environments.

(a) **(b)**

Figure 2.5: (a) Access point prototype. The system consists of a pre-configured router and a 12V battery. This system was placed in a bus stop and inside a bus. (b) Transmission range of a pre-configured access point. The effective range is the range in which the client is able to connect to the server, send information and remain connected; the actual range is the range in which the client is able to detect the server but is not able to connect and transmit information. Any access point located at a distance greater than 211ft is considered out of range.

The application was installed in a Nexus 7 tablet and a record of the trials was kept. A trial was declared successful if the client was able to connect to an access point within a reasonable amount of time and if the client and server were able to communicate with each other without a single data packet loss. In general, results show that there were no discovery, connection time and transmission range issues for 5 out of the 6 different conditions. Only in one condition (connection switch when the bus approaches the bus stop) the system was not able to communicate properly with the in-bus access point for the initial trials; proper revision of the software led to successful connection trials in subsequent trials.

An example of connection sequence with timing information is reported in Fig. 2.3b. Typical transmission ranges are shown in Fig. 2.5b. It was observed that the effective range (the range in which the client is able to connect to the server,

send information, and remain connected) was approximately 55 yards (165 feet). The actual range (the range in which the client is able to detect the server but not able to connect or transmit information) was approximately 70 yards (210 feet).

2.6 User Study

To evaluate the PTA system, we conducted a user study with four blind participants (three males and one female, ages: 55-67). Participants were offered the option to use earphones during the tests, rather than listen to the tablet's speaker. Only two participants decided to user earphones. We also instrumented three bus stops in our university campus and one bus vehicle. After a participant had a chance to ask questions about the system and the experiment, he or she was accompanied by the authors to the first instrumented bus stop. The app was then started and the tablet handed to the participant, who was instructed to select a specific final destination, using the system's tap and swipe interface. While waiting for the bus, participants were encouraged to occasionally interrogate the system, asking for the waiting time till bus arrival. Once the bus arrived and the participant received confirmation by the system that this was indeed the desired bus, the participant was accompanied inside the bus, where he or she took a seat in one of the front seats reserved for people with disabilities. During the trip, the participant was informed by the system about each upcoming bus stop. Note that the same information was also announced by the speakers in the bus; however, our participants were able to hear the announcement multiple times, if desired, from the tablet. The participant was asked to pull the stop request cord to call the final stop when he or she determined that the stop was approaching. Once arrived at the destination, the participant was accompanied outside the bus, where he or she waited until the system announced that it had disconnected from the bus AP,

Figure 2.6: Blind participants using the system during the user studies.

at which point the system would go into standby mode. The participant was then accompanied to another instrumented bus stop, located across the street, and asked to wake up the application again (by a tap-and-hold gesture). The entire process was then started again, with a different final destination. The whole test (including waiting for the bus to arrive and traversing the route eastward and westward) took between one and two hours. At the end of the test, each participant participated in a semi-structured interview that was audio recorded. Fig. 2.6 shows our participants using the system as they selected a final destination or were waiting for the system to announce that the bus was arriving at the bus stop.

2.7 User Study Discussion

The experiments and interviews conducted with blind users of our system have brought to light a number of accessibility problems with the public transportation systems, along with possible technological solutions. The participants' shared experience clearly shows that using public transportation can be challenging for people who are blind. Missing the bus or the desired stop are relatively frequent occurrences for blind travelers.

Our PTA system was designed to provide the following functionalities: (1) inform the user about the bus lines through a specific stop, timetables, and possibly real-time information; (2) allow the user to select a bus line and desired destination stop; (3) inform the user when the desired bus vehicle arrives at the stop; (4) provide real-time location information en route, with specific (and possibly customizable) warnings as the bus approaches the final destination. Users of this system can have the system repeat the information as many times as they want, which can be very useful for someone with a hearing impairment, in noisy or confusing situations, or when one would benefit to hear the same information repeated multiple times for confirmation. Our main design goal was to provide the user with enhanced information awareness during a bus ride, in hopes that this would make the travel experience safer and more comfortable.

By and large, the system worked as expected, with the exception of a single connectivity problem, and some difficulties by two participants using the tap-and-hold interface. Both issues can be easily resolved, the first one by changing the logic to determine when to announce a connection, and the second one by substituting tap-and-hold with another gesture, such as triple-tap. Semi-structure interviews were conducted after each study and the questions focused on the use of pubic transportation, the perceived accessibility problems and strategies when using the public transportation system, and about the functionalities and usability provided by the PTA system.

In terms of accessibility and usability of public transportation, when boarding a bus, all our participants ask the bus driver for the bus number or they listen to the acoustic announcement made when the bus stops at the bus stop and opens the doors. When determining when to exit the bus, most of the participants listen to the acoustic announcements spoken when the bus gets near a bus stop, or some

let the driver know their final destination. When boarding or disembarking the bus, some of the participants mentioned that the buses sometimes don't get too close to the sidewalk. This leads to the participant having to walk on the street before walking on the sidewalk. In addition, if the participant is sitting at the bus stop bench and does not stand up, the bus may not stop and keep going. It was also mentioned that some bus stops are not easy to locate, unless a bench or bus stop shelter is found.

In terms of the functionalities of the PTA system, our participants enjoyed all functionalities offered by the system and generally gave us very encouraging reviews. The main criticism, shared by all participants, was directed at the short advance notice given by the system when the desired bus is arriving at the bus stop. While they considered the bus arrival warning to be a very useful feature, they would prefer that this warning, currently produced a few seconds before the bus pulls in, were given earlier. Unfortunately, by its own nature, the current system is unable to produce a warning at a much earlier time, as connection with the incoming in-bus AP is needed before the warning can be generated. Detecting the bus arrival with longer notice would require polling a real-time online bus tracker such as OneBusAway [29] or NextBus. In addition, the participants also mentioned that they would like the system to help them find the exact location of the bus stop, a functionality that our current technology cannot offer (at best, our system can provide a very approximate estimate of the distance to the Access Point, based on the received signal strength indicator or RSSI). Among the other functionalities provided by the system, the warning produced before the bus reached the final destination and the ability to be able to select the destination bus stop was appreciated by all the participants.

2.8 Conclusions

We have described the design and implementation of a system that provides personalized, real-time public transit information to a person who, due to a visual impairment or cognitive disability, may have difficulty using a bus. While our system was shown to adequately support all desired functionalities (except for what was noted above), other technological solutions are possible. In fact, one main disadvantage of the chosen technology is the need for installation of Wi-Fi Access Points at bus stops and within bus vehicles. In some cases, Wi-Fi APs are already installed, for example, on long-haul bus lines. Installation of bus stop with Wi-Fi APs have also been planned in some cities (e.g., San Francisco). In the case of existing APs, it is conceivable that these could be upgraded to offer similar services as our system.

Several of the same functionalities offered by our PTA system could be provided by a smartphone app that uses GPS data for localization, and has access to timetables and possibly real time information from the Internet. This solution would not call for any special infrastructure, but it would need good Internet connectivity and GPS signal. (Note that our system does not require Internet connectivity or access to the user's smartphone GPS.) In addition, a purely smartphone-based application may not be able to notify the user in real time when the desired bus has arrived at a bus stop. Regardless of the technology ultimately chosen, we believe that our experiments have shown that a personal travel assistant, implemented as a smartphone app, has great potential to improve travel-related information access for blind users, and that our study has highlighted the main functionalities that such a system needs to offer to be really useful to blind travelers.

Chapter 3

WeAllWalk: An Annotated Data Set of Inertial Sensor Time Series from Blind Walkers

This chapter introduces an openly accessible and annotated data set of inertial sensor time series data collected from blind people. The primary purpose of creating this first-of-a-kind data set was to be able to (1) learn about the body motion patterns exhibited by blind people, and (2) use this sensor data to benchmark or train a model that could be used for an indoor or outdoor inertial navigation system.

3.1 Introduction

For someone who cannot see, tasks such as finding one's own location or figuring out how to reach a certain location in a building can be daunting, especially if this person is not familiar with the building layout or if he or she has poor orientation skills. Lacking access to visual landmarks, a blind traveler can quickly

become disoriented; and if he or she at some point finds himself or herself being lost, tracing back their own steps can be equally challenging. For this reason, many blind individuals do not visit new places (office buildings, hospitals, or schools) without a sighted guide who can show them around and lead them to the desired destination. Without the ability to travel independently, people in this community may miss opportunities for education, employment, leisure, socialization, and participation.

Personal navigation systems are designed to provide their users with spatial information and directions when traveling to new places. While outdoor navigation is to some extent already solved by the use of GPS, this is not an option for indoor navigation, and various technologies are being explored. Of course, systems for indoor navigation are useful not only for blind travelers, but also for anyone who may need directional information at times. Indeed, there is increasing commercial interest in technology that may help people locate a shop in a mall, a room in a building, or one's own car in a parking lot. Several research groups have started building assistive applications on top of this technology, adapting it to the particular needs of specific communities of users.

This contribution focuses on systems that support indoor wayfinding using dead reckoning from inertial sensors. This approach has the advantage that it requires no external infrastructure (as with iBeacons or similar technologies) or use of a camera (as with image-based technologies). Note that, until wearable cameras are socially accepted and widely used, users of a camera-based localization system would need to take pictures of the environment with their cell phone, something that for a blind person may be difficult and possibly awkward in social settings. In contrast, inertial sensing can be conducted with a smartphone conveniently tucked in one's shirt or pants pocket.

Dead reckoning uses data from the inertial sensors (and from magnetic sensors, when the data they produce is reliable) to estimate the trajectory taken by the user. In theory, data from a tri-axial accelerometer could be doubly integrated to obtain its location. In practice, this is only possible with sensors attached to the walker's feet; by detecting when one's foot is resting on the ground, it is possible to perform a zero velocity update, thus largely limiting errors due to drift. When the sensors are worn elsewhere on one's body or garments, a safer strategy is to use them for step counting, and to indirectly recover one's position using an estimated stride length, as well as orientation information from the gyroscope. Various versions of this approach have been used to track a person walking in a place with known geometry (obtained, for example, from a floor plan). Even when the geometry of the environment is not known, it is possible to use dead reckoning (e.g., by means of step counting and robust turn detection) to help a person re-trace a path taken in a building.

Step counting and turn detection with a smartphone placed in one's clothing can be computed reliably if one walks with a steady gait and in mostly rectilinear paths. Blind individuals, however, often exhibit body motion patterns during gait that are markedly different than those of sighted people [44] (e.g., due to "scuttling" [27]). In addition, cane users, who are trained to execute the 2-point touch or constant sliding technique [14], swing their cane-holding arm left and right, resulting in additional upper body rotation. As already observed by other authors [27], step counting may be difficult (or require specific parameter tuning) to work robustly with these individuals and for any smartphone placement. Likewise, blind individuals, especially when walking in large spaces, and unless they use a guide dog, do not always walk on straight paths with sharp and clearly

detectable turns. Rather, they often veer involuntarily, and need to correct their path when they realize that they are getting close to a wall or an obstacle.

We introduce a new, openly accessible and annotated data set of inertial sensor time series collected from blind individuals walking through relatively long and complex paths in realistic conditions, and carrying two smartphones in different locations on their clothing. The primary purpose for creating this data set was to allow other researchers to benchmark their algorithms (step counting, turn detectors, or other) on a common ground. This follows the example of other similar data sets (described in Sec. 3.2.3), with the critical difference that our WeAllWalk data was obtained from blind walkers, using either a long cane or a guide dog. More importantly, this data set does not just contain measurements from people walking on a straight line, as in previous collections [17, 71]. Instead, our participants walked on multiple paths with different levels of complexity, including turns at 45°, 90°, and 180°, as well as through doors that needed to be opened. While walking, our participants occasionally veered off the straight path, got caught in wall openings, and collided with obstacles. These events (which are faithfully recorded in the WeAllWalk data set) are to be expected when walking without sight. We carefully annotated our measurement time series, indicating the start and end time of each such event. In addition, we provide ground truth data in the form of heel strike times, measured by accessory inertial sensors clipped to the participants' shoes. We believe that this annotated data is representative of typical situations encountered by blind walkers, and that it should be very useful for anyone who wants to test their dead reckoning algorithms in realistic scenarios.

This chapter is organized as follows: After the related work, Sec. 3.2, we introduce the WeAllWalk data set in Sec. 3.3. We describe the sensor platform, the paths and their characteristics; we also introduce the participants to this study,

the procedures that were followed, and our criteria to annotate the data collected. And Sec. 3.4 has the conclusions. The WeAllWalk inertial sensor time series data set is available at `http://n2t.net/ark:/b7291/d1cc7g`. It is released under the terms of the Creative Commons Attribution license (CC-BY-4.0).

3.2 Related Work

3.2.1 Indoor Navigation via Inertial Sensing

There has been increasing interest over the past decade in personal navigation systems that support users in determining their location and in finding a path to a desired destination. While outdoor localization can be obtained, at least with an accuracy of a few meters via GPS, this is not possible indoors, where the GPS signal becomes too weak for detection. Indoor navigation represents the "last frontier," with whole conferences devoted to this subject [45, 91]. A variety of techniques have been proposed [28] for indoor localization, including radio-frequency triangulation [96], image-based recognition [53], Bluetooth beacons [68], visual markers placed in specific locations [25], and dead-reckoning using inertial sensors (see survey by Yang et al. [97]). The use of inertial sensors for blind indoor wayfinding has also been considered by several authors [22, 26, 27, 55, 69, 76, 79, 97].

3.2.2 Step counting

Automatic step counting (e.g., for physical activity tracking) has received considerable attention by the research and industry communities. Commercial pedometers use sensors that can be embedded in shoes (e.g., the Adidas Micropacers), in a smartwatch, in a smartphone, and attached to ankles or a belt. We refer

the reader to [95] for a review of different sensing modalities for step counting and other physical activity monitoring. A variety of algorithms have been proposed for stride event detection from inertial sensor time series; an excellent review of some of the main algorithms is presented in [17]. Sensor placement certainly has a role in the characteristics of the data collected. For example, ankle or foot worn sensors usually provide more accurate step counting [35] than waist worn sensors. However, step counting accuracy does not seem to be greatly affected by the specific location of the sensor on other parts of the body [39, 17] (including on head-mounted displays [4]).

Whereas the vast majority of step counting algorithms have been developed for able-bodied ambulators, some authors have addressed the performances of these algorithms with sensors carried by people with some level of mobility impairment. For example, [64] evaluated different algorithms with ten mobility-impaired geriatric patients, while [98] designed and tested robust stride event detectors for users with Parkinson's disease. In both cases, participants carried an accelerometer on a belt around their waist. While none of the blind individuals who contributed to the WeAllWalk data set could be considered to have mobility impairment, use of a long cane or a guide dog may result in a gait pattern that is quite different than for sighted walkers.

3.2.3 Similar Datasets

We are aware of two existing openly accessible data sets with inertial time series collected from walkers carrying a smartphone; these data sets are briefly described below. Other similar data sets exist, but with different sensors and body placement (e.g., foot-mounted sensors [3]) which are not directly relevant to our intended use case. The Walk Detection and Step Counting on Unconstrained

Smartphones dataset [17] consists of time annotated sensor traces (accelerometer, gyroscope, and magnetometer) obtained from 27 participants walking a route at three different walking paces, and carrying one or two smartphones placed in various positions while walking. The OU-ISIR Gait Database [71] consists of walking data from 744 participants wearing four sensors (three units with accelerometer and gyroscope, and one smartphone containing an accelerometer) located in a belt around the participants' waist. Participants walked on straight paths at varying inclinations.

WeAllWalk differs from these prior data sets in two main aspects. First, it contains data from blind walkers, both using a long cane and a guide dog. Second, the paths traversed by our participants are much more complex and realistic than the straight routes considered in the previous data sets. The routes in WeAllWalk include turns at corridor junctions, active door openings, as well as sporadic stops or short re-routings due to involuntary collisions with objects or walls, as should be expected during regular blind ambulation. We carefully annotated the time series to identify intervals corresponding to walking in a straight line, taking a turn, or opening a door, as well as specific "features," such as when the walker stopped for a short moment, bumped into an obstacle or a wall, or deviated momentarily from the path, because for example, he or she missed a door or got stuck in a wall opening.

(a)

(b)

Figure 3.1: Examples of placement of the CPRO shoe-mounted sensor for ground truth step detection. The sensor is contained in the white small case, attached to a plastic padded clip, and clipped to the back of the shoe.

3.3 The WeAllWalk Dataset

3.3.1 Sensor Platform

3.3.1.1 Sensors

Our participants carried two smartphones (Apple iPhone 6s), placed in different locations on their garments. Each smartphone recorded data from its tri-axial accelerometers, gyroscopes, and magnetometers. Data was sampled at a rate of 25 Hz. In addition, we recorded derived data produced by the iOS' Core Motion framework via proprietary sensor fusion algorithms. This derived data includes the estimated direction of the gravity force, the device's actual acceleration (obtained by subtracting the estimated gravity acceleration from the data measured by the accelerometer), the corrected magnetic field, and the device's attitude (the 3-D rotation of the device with respect to a static reference frame). Each data sample was time-stamped with the clock of the phone that originated it.

In addition to the smartphones, our participants carried two small inertial sensor units clipped to their shoes (see Fig. 3.1). We would like to emphasize that we do not assume or expect that blind walkers would wear these shoe-mounted

sensors in their daily life. These sensors were added for the sole purpose of enabling ground truth step counting (since placement at the foot level enables robust step detection [35]). Algorithms for step counting from inertial sensors in the smartphone can then be benchmarked against this ground truth data. We used MetaWear-CPRO units (shown in Fig. 3.1), which contain a 16-bit tri-axial accelerometer and gyroscope IMU from Bosch (BMI160). The accelerometers can work at a programmable range of $\pm 2g$, $\pm 4g$, $\pm 8g$, or $\pm 16g$, whereas the gyroscope can work between ranges of $\pm 125°/s$, $\pm 250°/s$, $\pm 500°/s$, $\pm 1000°/s$, or $\pm 2000°/s$. For the experiments, we set the accelerometer range to $\pm 2g$, and the gyroscope range to $\pm 500°/s$. The inertial sensor time series measured by the shoe-mounted sensors (sampled at 25 Hz) are recorded together with data from the smartphones, and later processed to detect the ground truth heel strike times. Foot strike events for each foot are detected from these sensors using data from the Y-axis gyroscope (as in [81]) using a modified version of the UPTIME algorithm [2] (see Fig. 3.2).

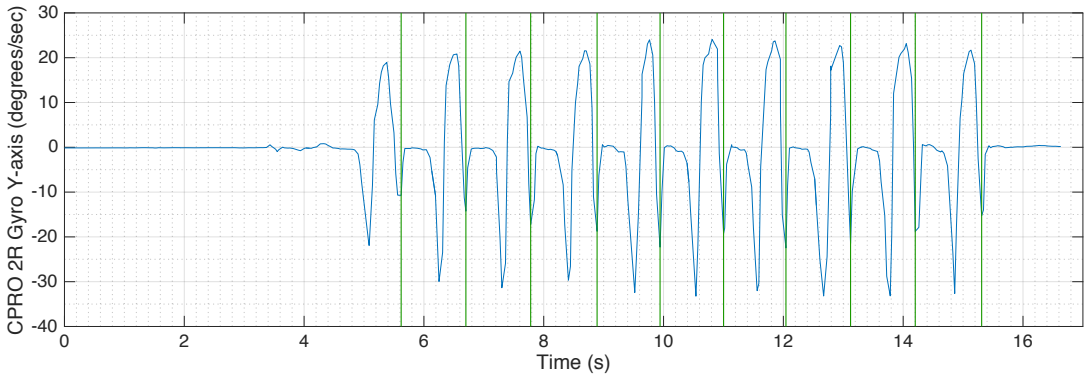


Figure 3.2: Time series of measurements from the Y-axis gyroscope in the CPRO sensor clipped to one of our participants' right shoe during the calibration pretrial. The green vertical lines represent heel strike times.

All of the devices carried by our participants (two iPhone 6s and two foot-mounted sensors) are controlled via Bluetooth by a single iPhone 5 (called "control phone") carried by one of the experimenters. The system makes use of the Multi-

peer Connectivity framework to communicate between multiple iOS devices, and the MetaWear iOS Objective-C API to communicate with the MetaWear-CPRO sensors. The "control phone" is paired with each of the smartphones carried by the participant in order to broadcast commands to them, as well as receive status updates (e.g., acknowledgement that a command was received, or battery life status from the MetaWear-CPRO sensors). The smartphones carried by the participant are then paired with each of the MetaWear-CPRO sensors, one per smartphone. This pairing is done remotely from the "control phone." Once the "control phone" is paired with the two smartphones carried by the participant, and each one of these is paired with a MetaWear CPRO sensor, the "control phone" broadcasts a series of commands. Some of these commands include starting and stopping the inertial sensors, synchronizing the smartphones and the MetaWear-CPRO sensors so that all sensor readings reference the same starting time, saving all the sensor data from the smartphones and the MetaWear-CPRO sensors, restarting the system for the next experiment, and more. All of this is done remotely and without having to physically interact with the smartphones carried by the participants during the experiment (e.g., having to take the smartphones out at the end of each trial in order to restart the system or save the data). All the sensor data from the MetaWear-CPRO sensors is streamed to the smartphone paired with and recorded along with all the sensor data produced by the iOS Core Motion framework.

3.3.1.2 Calibration Pre-trial

Before starting to walk on the prescribed paths, each participant went through a "calibration" pre-trial, which consisted of walking along a straight corridor for twenty steps. The approximate time of each heel strike for each foot was recorded

manually by an experimenter (by tapping on the screen of the control phone each time the participant placed a foot on the ground). This pre-trial phase is used to calibrate the parameters of the ground truth step detector from the shoe-mounted sensors described earlier (this calibration is performed off-line after data collection). The ground-truth step detector is assumed to be well calibrated when the steps events it identifies correlate well with the steps that have been manually recorded by the experimenter. Note that the manual step input is performed only during the pre-trial.

3.3.2 Paths

Our participants walked on 6 different paths in two different buildings in our campus (labeled as E2 and BE). The first paths (T1 to T4) are located in the E2 building, while the last two (T5 and T6) are located in the BE building. Floor maps of the buildings and the routes are shown in Fig. 3.3. Paths were all indoors and on level terrain (one path contained a stretch on an outdoor terrace). We decided against including staircases in the paths, due to safety concerns. Routes where chosen to have a variety of lengths and complexities. The shortest path was about 75 meters long and only included one 90° turn; the longest path was about 300 meters long along an 8-shaped route, included seven turns and required the participant to open three doors. One path included a 180° turn, while three paths included a 45° turn. The path order was designed in such a way that the end point of a path in the sequence corresponded to the start point of the next path.

For most of the time, participants walked in a corridor (with the width of the corridor varying from 120 cm to 210 cm), but some paths included traversal of an open space (an elevator hall or an entrance hall) as well as a passage next to a

Figure 3.3: Top two rows: the floor plans of E2 and BE, respectively. Third row: the trajectories taken by our participants. T1-T4 were located in E2, T5-T6 in BE. Fourth row: Detailed annotated map of trajectory T2 walked by the participants.

stairwell. In some cases, a turn was preceded or followed by a door that needed to be opened. In these cases, we informed the participant in advance of the presence of a door, and of whether the door had to be opened by pushing on a crowd bar, or be pulled open by a handle. In two places, the path went through a door that required substantial force for opening; in these cases, one experimenter opened the door for the participant.

Floor surface varied from industrial carpet to linoleum to rugged concrete (in the outside terrace). In addition, two industrial flat mats were placed in an elevator hall, and a metal plate was placed across a corridor. Some of our participants got their cane tip or their shoe briefly stuck at the edge of these floor coverings. Most environments were devoid of obstacles, although a few corridors contained large pillars, couches, chairs, tables, garbage bins, and obstacles in the form of appliances, which were kept on one side of the corridor. In these situations, we advised the participant to keep close to the opposite side of the corridor. Some corridors contained openings to rooms or to other corridors, and a few participants occasionally moved close to these openings and got caught in the wall corner; this typically caused a short stop before the participant was able to get back to the intended route. At times, the participant also had to stop and move to the side to avoid walking into people who were standing in the corridor or were walking towards the participant. On the day participant P6 visited, some corridors were encumbered by one or more ladders due to ongoing work. In this case, we directed the participant by voice to avoid the ladder.

3.3.3 Participants and Procedure

Eight blind volunteers and five sighted volunteers participated in the study. Note that the focus of this data set is on blind walkers; we added data from sighted participants only as a "control," for comparison in identical settings.

3.3.3.1 Blind Participants

Participant P1 is a 66-year-old woman who has been blind since she was very young. She has a guide dog, a Labrador Retriever, who is functioning, but close to retirement. P1 feels that her dog is becoming distracted and is not as good as he used to be at staying away from obstacles, and for this reason she recently took some classes to refresh her long cane skills. She walked paths T1 to T4 with the dog first, then again with the cane. She then walked paths T5 and T6, with the dog first, and then again with the cane. She felt that using the dog allowed her to walk on a straight line, while she tended to veer while walking with the cane; this was confirmed by our observations. Her dog, which she held on a harness with her left hand, often pushed her close to the right side of the corridor. When walking with the cane, P1 sometimes got stuck in a wall opening and had to walk away from it to resume her path. She slides her pencil-tipped cane left and right, synchronized with her gait.

Participant P2 (aged 46) lost her sight over the past five years due to diabetic retinopathy (the diabetes also caused some neuropathy at her feet). She uses a long cane for mobility, although she is looking forward to receiving a guide dog in the near future. She is still perfecting her mobility skills, and feels that she is not moving as gracefully as other people in her condition. Her cane has a ball tip; she slides it left and right, synchronized with her gait. She often hit a sidewall with

her cane, and sometimes bumped into obstacles along the way (e.g., a garbage bin or a chair).

At 26 years of age, P3 was the youngest participant in our study and she has been blind since birth. An expert cane user, she had a guide dog in the past. She, however, admits that her orientation skills are poor, so she was glad to hear that this study required no route memorization. P3 was able to walk on straight paths without much veering; however, she did get caught in a wall opening a few times. She uses a cane with a ball tip, sliding it on the floor in a swinging motion that, however, is generally not synchronized with her gait.

Participant P4 is a 65-year-old woman who lost her sight soon after birth. She didn't bring her cane, and thus was tested only with her dog, an energetic German Shepherd, who walked very fast as she held the harness with her left hand. The guide dog followed P4's commands faithfully, although at one point, while in the stairwell that joins two corridors, the dog almost started leading P4 downstairs instead of walking straight past the staircase. P4 explained that the dog might have been wanting to walk to P4's husband, who was waiting downstairs in the parking lot.

Participant P5, aged 59, is a man who lost his sight at 18 months of age. He never had a guide dog, and is not interested in one. He is an expert traveler, with excellent orientation skills. He often travels independently by public transit. He had a peculiar way of using his pencil-tipped cane. Instead of swinging his cane left and right, he holds it at an angle in front of him, and taps it on the ground at regular intervals. He explained that, by listening to the sound and its echo, he could tell the presence of nearby surfaces. He walked, for the most part, with very little veering.

Participant P6 is a 68-year-old man. He lost his sight due to a traumatic brain injury as a teenager. P6 used a telescopic cane with a round metallic glide tip, which he maneuvers in a swinging motion synchronized with his gait. He slid the cane on the floor except for the outside terrace with rugged concrete surface, where he instead tapped it (2-point touch). P6 explained to us that he normally uses a different, heavier cane when walking outdoors. He was able to walk in straight lines and avoided almost all obstacles, without hitting any wall or being caught in wall openings.

Participant P7 is a man, aged 46, who has been blind since birth. He has excellent orientation skills and regularly travels even long distances using public transportation. He uses a single piece long cane with round metallic glide tip, which he slides on the floor in a swinging movement synchronized with his gait. In our trials, he walked with little veering. In a couple of occasions, he bumped his shoulder into large obstacles along the path.

Participant P8 is a 69-year-old woman who lost her sight progressively during her young age. Similar to P1, she walked all paths twice, one time with her guide dog and the other time using a long cane (pencil tip). She is a proficient traveler, yet she often times veered off the straight direction when walking in a corridor and had to correct her path.

3.3.3.2 Smartphone placements

Each participant was asked to choose a comfortable location for the two smartphones used in order to take inertial measurements during the trials. Preferences varied: sometimes a smartphone was placed in the front or back pants pocket, while in other cases it was placed in a holster clipped to the participant's belt, in a jacket pocket at waist or breast height, or tucked under the participant's

shirt at shoulder level. Tab. 3.1 shows the different placements for our blind participants. Informal surveys (e.g., [46]) have shown that the majority of people keep their phone in their pocket, and for this reason we didn't consider placement of the smartphones in the participants' handbag or backpack. In addition, step counting with smartphone in a handbag was shown to be inaccurate [17] due to extra swinging of the bag. We also didn't consider the case of a smartphone held in one's hand while walking, as this may be inconvenient for blind people who already have one hand occupied holding a cane or a guide dog.

Table 3.1: Blind participants list.

ID	Mobility Tool	Phone 1 Placement	Phone 2 Placement
P1	Cane | Dog	Left breast pocket	Jacket right side pocket
P2	Cane	Jacket left side pocket	Tucked under shirt on right shoulder
P3	Cane	Pants left front pocket	Pants right back pocket
P4	Dog	Jacket left side pocket	Pants right back pocket
P5	Cane	Pants left front pocket	Pants right back pocket
P6	Cane	Holster clipped to front right belt	Pants left front pocket
P7	Cane	Jacket right side pocket	Pants left front pocket
P8	Cane | Dog	Pants right front pocket	Pants left front pocket

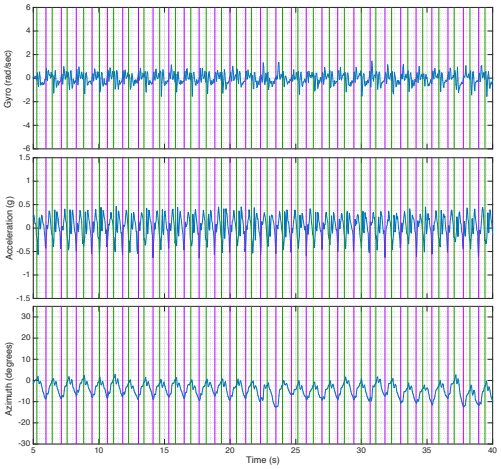
3.3.3.3 Procedure

After signing the IRB-approved consent form, each participant was shown the CPRO sensors in their clip cases, and asked to clip each sensor case to the back, if possible, or to the side of their shoe (see Fig. 3.1). (Note that participants were advised in advance of their visit to wear comfortable shoes, and to wear clothing

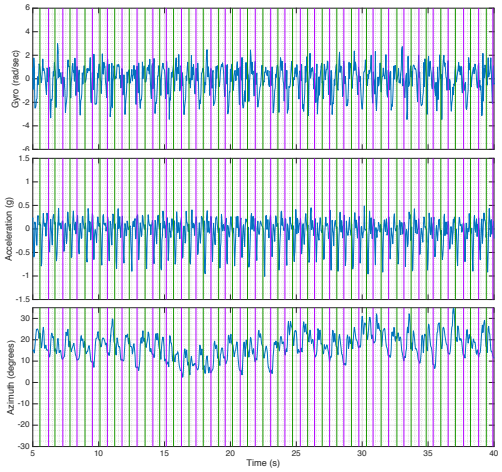
with pockets.) Then, the participant was asked to position the two smartphones, as discussed in Sec. 3.3.3.2. Participants were advised not to pay attention to any speech produced by the smartphones (which were programmed to utter short synthetic speech verification sentences upon successful pairing with the control phone). Participants were also advised to begin walking when prompted by an experimenter, and to walk straight until asked by the experimenter to turn left or right (or, in the case of path T5, to turn around), to push or pull open a door, or to stop at the end of the path. These were the only verbal directions provided to the participants, except for occasional safety warnings (e.g., as mentioned earlier, participants were advised to walk closer to one side of a corridor if there were obstacles on the other side).

No training on the use of the system was necessary, since the task was for the participants to simply walk naturally. Each participant first went through the pre-trial described in Sec. 3.3.1.2 for ground truth calibration. Then, he or she was accompanied to the start position of the first path, and asked to start walking in the designated direction. Before the start of each path, participants were oriented to face the correct direction; this was particularly important for paths T2, T5 and T6, which started with diagonal traversal of an entrance or elevator hall. All trials with blind participants were supervised by two experimenters. One of the experimenters managed the start and end of data collection from all sensor platforms via the control phone, and recorded videos of all sessions by means of a GoPro HERO Session camera attached to a head strap. The other experimenter walked at a close distance behind or sometimes in front of the participant, and was in charge of ensuring the participant's safety.

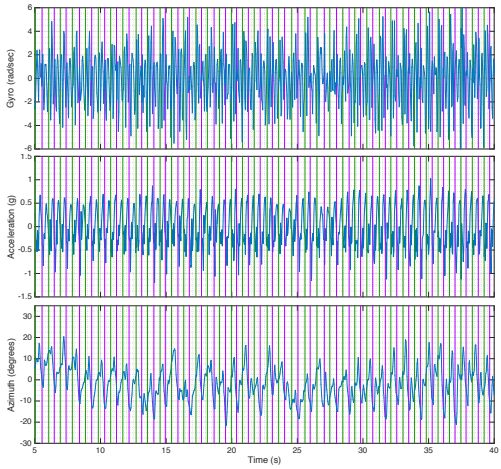
Fig. 3.4 shows an example of time series data collected during a straight path in route T3 for three individuals: a sighted participant, a blind participant using



(a) A sighted participant



(b) Participant P3 using a long cane



(c) Participant P6 using a guide dog

Figure 3.4: Time series of measurements from accelerometer, gyroscope, and azimuth. The magenta and green vertical lines mark the left and right foot strikes.

a long cane, and a blind participant using a guide dog. For the accelerometer and the gyroscope sensors, the first and second subfigures plot a linear combination of the time series from the three axes, corresponding to the principal component. The azimuth data (angle around the vertical) was obtained from the iOS Core-Motion Framework, and is defined with respect to an arbitrary horizontal axis. (Note that the magnetometer is not used for this purpose, as we found that it decreases the quality of the azimuth in indoor environments.) The plots also display the heel strikes times (shown by vertical lines) for each foot. Observation of the azimuth time series provides some insight into the gait characteristics of each individual. In particular, the sighted walker maintained a steady heading direction (with oscillation due to natural body swinging). The azimuth time series of the blind walker with a cane shows a more variable pattern, with variation in heading direction as large as 20 degrees. The blind participant using a guide dog maintained a more stable heading direction, but with a wider swinging action.

3.3.4 Data Annotation

After completion of all trials for a participant, the data from all sensors was offloaded to a desktop computer for post-processing. In particular, all data streams were synchronized as discussed in Sec. 3.3.1.1. The video streams collected from the GoPro camera were also synchronized to the same time base used for the sensors. The heel strikes times for each foot (computed by the CPRO sensors, Sec. 3.3.1.1) were recorded.

The time lapse during traversal of a route was divided (by visual inspection of the video) into contiguous intervals, where each interval corresponds to either a straight segment in the path, or to a "turn" event. For example, traversal of route T1 (shown in Fig. 3.3) was divided into seven contiguous time intervals,

Figure 3.5: Four of our blind participants dealing with specific situations. Top two images: being caught in a wall opening. Bottom left: pushing a door open. Bottom right: avoiding an obstacle (a ladder) in the way.

corresponding to four straight patches interleaved with three 90° turns. The cardinal direction of each straight path, or of the paths joined by a turn, was recorded in the annotation file, together with the start and end time of each interval, and with the number of steps taken during the interval. In addition to the segmentation into straight paths and turns, we created annotations of particular events such as opening a door, bumping into an obstacle, being caught in a door opening, or stopping momentarily (see Fig. 3.5). These events are normally associated with anomalous characteristics in otherwise regular inertial data time series (see Figs. 3.6, 3.7, 3.8). Also note from these figures that when participants are engaged in tasks such as opening a door, the shoe-mounted sensors sometimes detected "phantom steps" when in fact the participants were simply balancing themselves on their feet. We did not manually remove these phantom steps, as they occurred only sporadically in our study. All of the data was annotated by one experimenter and independently checked and verified by another experimenter. The annotation file, which is stored using the Extensible Markup Language (XML) format, also includes other relevant information such as the type of mobility tool used, as well as some general gait pattern observations.

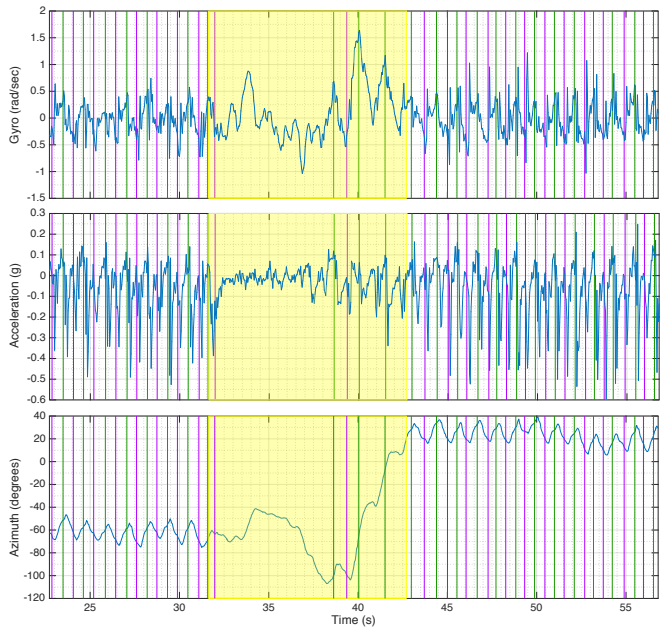


Figure 3.6: Sensor data from a participant pulling a door open then making a left turn.

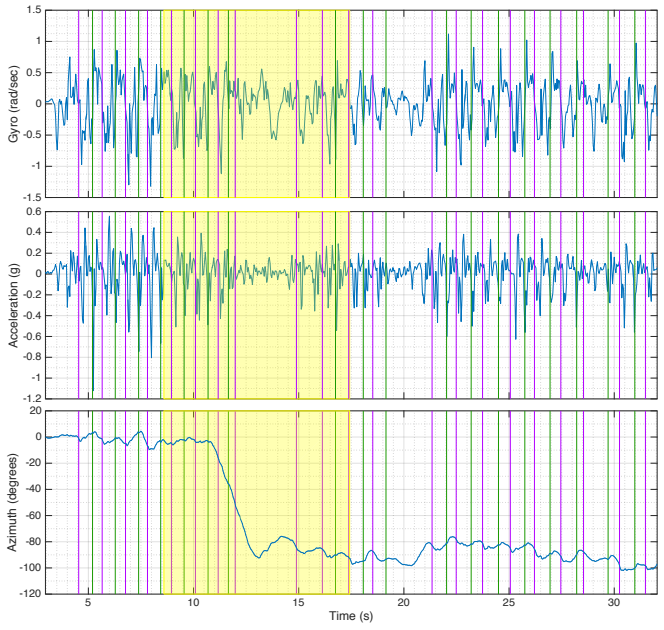


Figure 3.7: Sensor data from a participant making a right turn then pushing a door open.

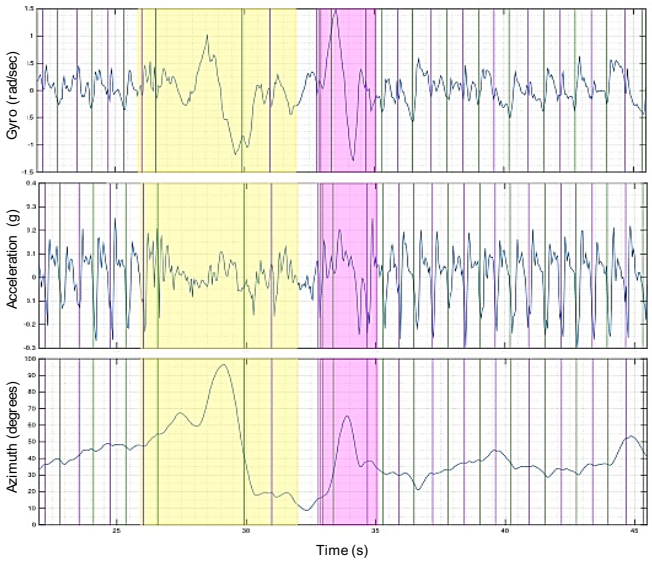


Figure 3.8: Sensor data from participants in different situations. being caught in a door opening (yellow area) then hitting her arm against the wall (magenta area).

3.4 Conclusion

We have introduced a new data set with inertial sensor time series collected from blind walkers. The participants walked through fairly long and complex routes; on their way, they sometimes had to open doors and avoid obstacles. The data has been subdivided into straight paths and turns, and carefully annotated, with special events (such as bumping into an obstacle) individually identified and marked. While we believe that this data can be useful to several researchers who are interested in personal mobility, we are also aware of some of its shortcomings. For example, although our participants were asked to walk naturally, they didn't have to find their way independently (as they were instructed when to turn). Participants may also have felt self-aware, as they were being followed and observed, and thus may not have been fully natural (for example, they may have put extra effort to avoid obstacles). All of our routes were indoors, and thus our data is not

representative of outdoor ambulation. As one of our participants explained, some blind travelers pay attention to different things when walking indoors and outdoors. For example, when walking indoors, they may be careful to avoid obstacles such as a door left ajar; while walking outdoors, typical concerns include the condition of the pavement, and the possibility of a hole or a curb. Nevertheless, this data set allows other researchers to benchmark their algorithms (step counting, turn detectors, or other) on a common ground, and we believe that this annotated data is representative of typical situations encountered by blind walkers.

Chapter 4

Robust Path Back-Tracing Guidance System

Inspired by the work described in Ch. 2, this chapter describes a novel system we developed to support safe indoor wayfinding for the blind or visually impaired. The system consists of a turn detector to robustly detect turns, a step counter to count the steps taken along a path, and a path matching algorithm to determine the user's current location. The WeAllWalk inertial sensor data set described in Ch. 3 is used to train the models described in this chapter in order to develop a robust path back-tracing guidance system.

4.1 Introduction

One of the main challenges for the blind or visually impaired is the ability to safely and independently navigate in indoor environments. Consider for example the case of a blind person going to a doctor's appointment as illustrated in Figure 4.1. Upon arrival at the hospital's waiting room, the patient is faced with the challenge of going from the waiting room to the doctor's office. A typical scenario

might consist of a sighted receptionist accompanying the patient from the waiting room to the doctor's office. Once the appointment has concluded, the same receptionist might come back and walk the patient back to the same waiting room. However, in some cases, the same receptionist might not be able to help the blind patient go back to the waiting room, or the blind patient might try to find the way back on his or her own (by remembering the turns and steps taken, as well as noises during the walk to the doctor's office). In either case, the patient might arrive at a different waiting room altogether. There could be many different possibilities, all with the possible risks and concerns for the blind patient's safety and whereabouts. Given this scenario and the challenges of building a robust system to help blind or visually impaired persons safely back trace the path to go back to the location from which they started, we designed a safe return system that uses the inertial sensors in a smartphone device to count steps and detect turns during traversal of a path in a building.

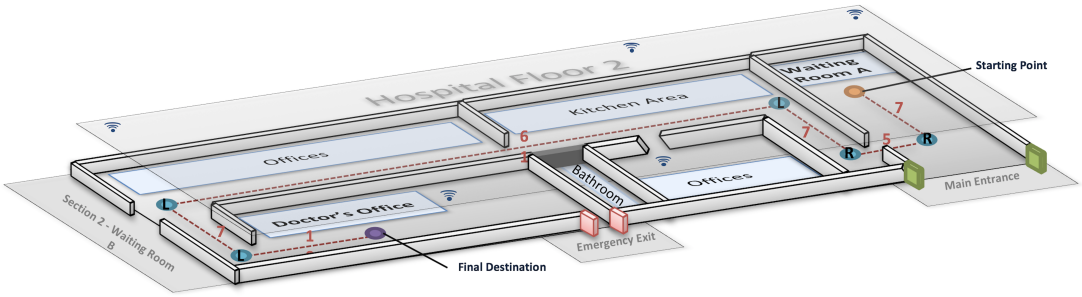


Figure 4.1: Hypothetical path traversed by a blind patient at a hospital. Upon arrival at waiting room A, the receptionist guides the blind patient to the doctor's office, a few corridors down. Then after the appointment has ended, the blind patient needs to traverse the same route in order to go back to Waiting Room A.

In this chapter, we describe the robust path back-tracing guidance navigation system for the visually impaired that consists of 3 components: (1) a turn detection module to robustly detect turns (Sec. 4.3); (2) a step counting module to accurately count the number of steps taken along the path (Sec. 4.2); and (3) the

safe return module that uses a path matching algorithm (Sec. 4.4.3) to compare the steps and turns on the way to a destination with the steps and turns on the way back to the starting location. These three modules work in unison to provide the guidance necessary to reach the starting location. The advantages of using a simplified route representation that consists of steps and turns is that it can be obtained without any prior knowledge of the building's floor plan, and that an assistive system of this kind may be sufficient to efficiently and safely help a blind person retrace the route taken inside a building and walk back safely to the starting location.

4.2 Step Counting

4.2.1 Introduction

Pedometer systems are widely available in the market today to monitor different kinds of motion activities for medical, localization or recreational activities. A previous survey [21] reviews standalone pedometer systems such as the Omron HJ112 and the HJ 720-ITC, as well as smartphones with pedometer functionality such as the Nokia 5500 Sport. These modern systems typically make use of miniaturized electro-mechanical (MEMS) sensors to sense and measure all sorts of physical modalities (e.g., pressure, motion, rotation) [24]. One of the sensors widely used in step counting techniques is the accelerometer. Accelerometers measure linear acceleration in the x, y, and z direction with respect to the device orientation as shown in Figure 4.2. The oscillating and cyclic motion of the accelerometer readings can be analyzed and processed for motion and step detection. Gyroscope readings, which measure the angular rate of rotation around each

of the orthogonal axes, have also been explored and used for step counting with promising results [48, 63].

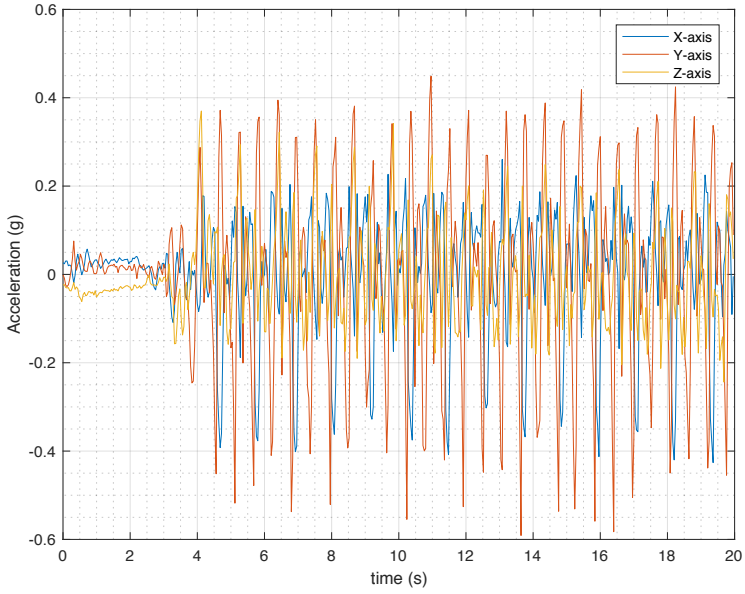


Figure 4.2: Accelerometer time series readings collected by an iPhone6 kept in the walker's side pocket.

In this section, we explore some of the techniques that have been proposed in literate to detect steps based on accelerometer or gyroscope sensor measurements. In particular, we implement and experiment with a total of six different techniques to detect steps (see Tab. 4.1 for the list of the selected techniques). All the algorithms investigated in this section have a set of parameters that must be configured prior to using them in order to obtain optimal step counting results. Therefore, we make use of the WeAllWalk dataset described in Ch. 3 to train all the models and we benchmarked the different step counting algorithms using two different metrics. Based on these results, we chose the best step counting algorithm to use in the user study described in Sec. 4.4.5.

4.2.2 Related Work

Various step counting algorithms based on accelerometer readings have been surveyed and evaluated in order to identify the most reliable smartphone-based step counting [17] technique. Some algorithms simply apply a threshold to some property (e.g., signal magnitude) of the filtered accelerometer data. When the magnitude or intensity of the property exceeds a threshold, the algorithm registers that a step was taken [51, 70]. Other popular step counting algorithms search for peaks, valleys or zero crossings on the filtered accelerometer or gyroscope data and equate each finding to one or more steps [48, 73, 71, 20, 80, 84]. Furthermore, more complex algorithms based on probabilistic models (hidden Markov model (HMM), particle and Kalman filters) have been proposed to either locate and discern between the different gait movement phases (e.g., heel strike or heel off) or filter the signal to isolate and search for local maxima or minima [63, 90, 52]. Some of these algorithms have reported step accuracy rates between 96%-100% [48], 98.9% [20], and 99.6% [80]. Step accuracy in this case is defined as comparing the estimated number of steps calculated by a given algorithm to the exact number of steps taken by the individual. A negative percentage means that the algorithm under counted steps and a positive one means that the algorithm over counted steps. Problems in step counting accuracy start to occur when the user stops walking and starts moving the phone in all sorts of directions (e.g., shaking the phone) or when the phone's orientation is changed while the user is in motion (e.g., answering a phone call while walking). These unwanted sensor readings tend to be classified as steps by most step counting algorithms. Therefore, most of the aforementioned algorithms have some sort of device placement limitation (e.g., trousers vs. handbag) and they have only been tested with data collected in the best ideal situations and from sighted individuals.

Two main factors play a critical role in making a step counting technique accurate: (1) the placement of the pedometer relative to the human body, and (2) the ability to detect and deal with unwanted movement of the pedometer while the person is walking (e.g., jolts, jerks or any other motion not classified as walking). Placing the pedometer in the optimal location allows for reliable step counts; detecting and dealing with unwanted movement of the pedometer while a person is walking decreases the chances of over counting steps. Pedometer accuracy by placing inertial sensors at three different locations (pocket, head and hand) has been previously investigated [5]. This study did not find any significant difference in step count accuracy between sensor locations. However, their study was limited to only these placement locations and as observed in the research we performed in [30], visually impaired individuals tend to place their smartphones in different locations comfortable and easily accessible to them (e.g., front shirt pocket, pants back pocket). In the case of detecting and dealing with unwanted movement, some techniques have been employed to mainly detect patterns that shouldn't be classified as steps [17, 89, 74]. To deal with some of these issues, some systems wait a few seconds after movement has been detected to start recording the number of steps in order to prevent misjudgment of steps [94]. This is not ideal in situations where the user needs to make sudden stops, such as a blind person who needs to stop very frequently in order to reorient himself or herself in unfamiliar environments. For these cases, the systems do not accurately record the exact number of steps taken by the person. Another system uses artificial neural networks to learn, detect and prevent the counting of steps when the device is shaken [89]. However, their implementation was designed to detect motion not characterized as walking (e.g., unintentionally shaking the phone) and the accuracy of the system was not measured.

4.2.3 Algorithms

Given the existing research on pedometer techniques, we implemented and evaluate a total of six different step counting algorithms, some of which were ranked the best in the survey done by [17], and one which we found suitable due to its reported high accuracy rate and use of gyroscope rather than accelerometer readings as used by most pedometer techniques [48]. Table 4.1 lists the algorithms we choose to implement and benchmark.

Table 4.1: List of step counting algorithms.

Algorithm ID	Step Counting Algorithm	Sensor Utilized	Source
1	Window Peak Detection (WPD)	Accelerometer	[17]
2	Hidden Markov Model (HMM)	Gyroscope | Accelerometer	[17, 63]
3	UPTIME	Accelerometer	[2]
4	Zero Crossing (ZC-gyro)	Gyroscope	[48]
5	AMPD	Accelerometer	[83]
6	ZC-acce (variant of 4)	Accelerometer	[48]

The Window Peak detection (WPD) algorithm [17] uses a centered moving average window on the smoothed, de-trended acceleration magnitude to find peaks associated with a heel strike; the hidden Markov model (HMM) proposed in [17, 63] is trained to discern the different phases during a gait period. A model with two hidden states is used to locate and discern between the different gait movement phases (heel strike, heel off) captured by the observations of the de-trended acceleration magnitude data (the original algorithm used the gyroscope data aligned with the X-axis. However, this data did not produce good results); The UPTIME algorithm [2] uses the de-trended acceleration magnitude in a finite state machine (FSM) with six states to identify the negative peaks associated with heel strikes; the Automatic multiscale-based peak detection (AMPD) technique of Scholkmann et al. [83] is a generic algorithm for peak detection in a signal. For step counting, it processes the magnitude of the acceleration, and finds the

peaks associated with heel strikes by detecting local maxima. The Savitzky-Golay filter [82] is used to smooth the acceleration magnitude before computing the local maxima; and the zero-crossing (ZC-gyro) technique proposed by Jayalath et al. [48] is based on using the gyroscopic data aligned in the medial-lateral direction and searching for zero crossings of the data. Assuming that the smartphone was kept approximately vertical and with its face parallel to the walker's body, this axis is approximately aligned with the smartphone's X-axis. A 6th order discrete Butterworth low pass filter is applied to smooth the signal before the zero crossing detector is applied. Each zero crossing is computed between positive and negative peaks larger than a threshold. A variant of the zero crossing technique that uses the de-trended acceleration magnitude (ZC-acce) instead of the gyroscope data was also evaluated to test the behavior of this technique when using different sensor data since most of the other techniques tend to use the de-trended acceleration magnitude.

4.2.4 Evaluation

The WeAllWalk dataset described in Ch. 3 was used to train and evaluate these step counting algorithms. This dataset enables benchmarking of new or existing algorithms since it contains carefully annotated ground truth data in the form of heel strike times. In particular, the algorithms are trained and tested on data acquired while walking straight segments, while any data that pertains to a turn or "feature" segment is not used since a step is not well defined in these segments.

In performing analysis on the algorithms listed in Tab. 4.1, we seek to find the difference in performance of these algorithms across the different "communities" of walkers: blind walkers using a cane (Blind:Cane); blind walkers using a guide dog

(Blind:Dog); and sighted walkers (Sighted). We also looked at differences between the considered algorithms using appropriate metrics, as well as at the potential influence on an algorithm's performance based of the phone location in one's garments. To train and test the different step counting algorithms, the WeAllWalk dataset was divided into four different training and testing sets. Tab. 4.2 shows the training and testing sets that were used for this analysis. In particular, we asked whether optimizing an algorithm's parameter on a certain community of walkers makes it less effective when used on another community. Set 1 aims to train the models using sighted data only, and test them using all blind user data. This approach is justified by the consideration that data from sighted walkers is, in general, more easily available than data from blind walkers (e.g., from other data sets (see Sec. 3.2.3)). Thus, one may ask how an algorithm trained on sight walkers would fare when used on data from blind walkers. Sets 3a-3c use blind or sighted user data divided by mobility tool aid to train and test the models. In particular, this set aims to independently test how the step counting algorithms perform when only trained on data for that particular modality. For this set we used the standard cross-validation technique (stratified leave one out), suitable for small size samples.

Table 4.2: Training and testing sets.

Set ID	Training	Testing
1	All sighted	All cane + guide dog
3a	Cane	Cane
3b	Guide dog	Guide dog
3c	Sighted	Sighted

4.2.4.1 Error Metrics

The quality of the step counting algorithms shown in Tab. 4.1 were evaluated using two different metrics. Both metrics compare the step detections (computed

on the data from the iPhones) with the ground truth data from the foot-mounted sensors.

Metric 1, SC-Error 1, seeks to find the total number of undercount or overcount events by looking at the time interval between two consecutive ground truth heel strike times. If the number of steps detected, n, within this interval is zero, an undercount event is declared. Otherwise, an overcount event equals to $n - 1$ is declared if $n > 1$. The cumulative number of overcount and undercount events is computed and normalized by the number of ground-truth steps. Figure 4.3 shows an example when no overcount or undercount events are detected by this metric. Every detected step falls within the interval between two consecutive ground truth heel strike times.

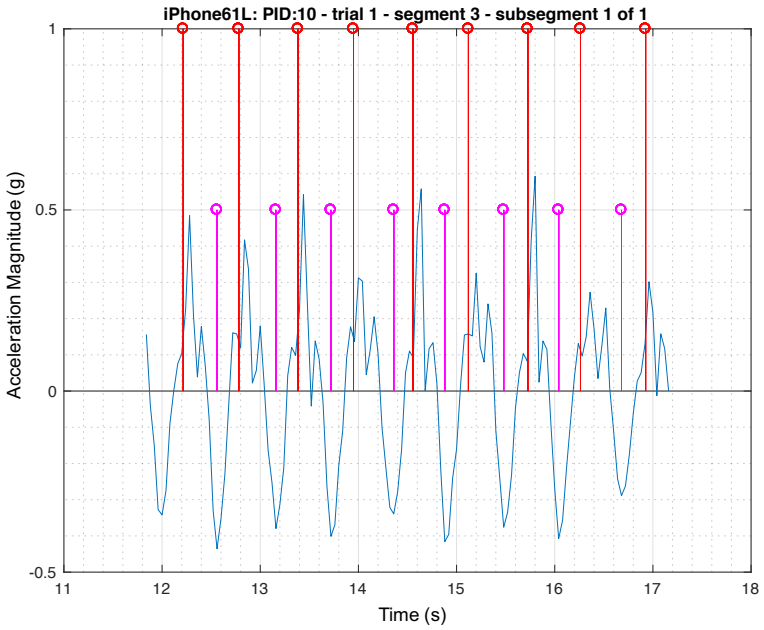


Figure 4.3: De-trended acceleration magnitude with steps detected (pink lines) by the UPTIME algorithm as compared to ground truth steps (red lines). In this particular case, no undercount or overcount events are observed.

The second metric, SC-Error 2, simply counts the total number of steps, N, in a segment and compares it with the total number of ground truth steps for that

segment. If the difference between the total number of steps detected and the ground truth is greater than zero, an overcount event is recorded. Otherwise, if the difference is less than zero, an undercount event is recorded. The total number of overcount and undercount events are summed together and normalized by the total number of ground truth events.

Note that the normalized undercount and overcount values obtained by the SC-Error 1 metric is always larger than or equal to the corresponding values computed by the SC-Error 2 metric, as missed counts or double counts between a time interval may even out over multiple steps. While the more lenient SC-Error 2 metric may be appropriate when measuring step counts over long distances, the more conservative SC-Error 1 metric may be useful when fine-grained tracking is desired.

4.2.4.2 Results

We processed the WeAllWalk data to benchmark the different step counting algorithms for the different communities represented in the data set. Figures 4.4 and 4.5 show the average SC-Error 1 and 2 for each community and for each considered algorithm under the Stratified Leave-One-Out training modality, respectively. All tests were conducted by taking the logarithm of the errors SC-Error1 or SC-Error2 as dependent variables.

We evaluated whether the two different training modalities considered (Train on Sighted and Stratified Leave-One-Out) are equivalent. Repeated ANOVA measures with blocking factors of Participant, Algorithm, and Phone placement found no significant difference in the mean value of either error SC-Error1 or SC-Error2.

We then considered the effect of an algorithm and the different communities for the Stratified Leave-One-Out: blind walkers using a cane, blind walkers using a

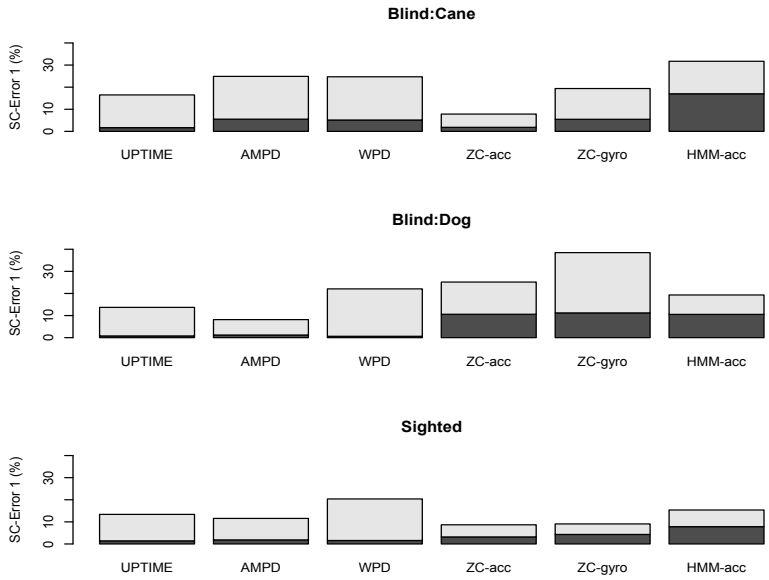


Figure 4.4: SC-Error 1 errors averaged over participants for the different communities and different algorithms considered for the Stratified Leave-One-Out training modality. Gray bars indicate undercount rates; black bars indicate overcount rates.

guide dog, and sighted walkers. This analysis addressed the effect of different step counting algorithms and different communities of users on the measured errors. The goal was to ascertain whether some algorithms are significantly better than others, and whether step counting produce significantly different errors for walkers in different communities. We tested the null hypothesis of equal error means for different levels of the factors Algorithm and Community, with blocking factors of Participant and Phone placement. Main effects and interactions were discovered using ANOVA at significance level $\alpha = 0.05$. Post-hoc multiple comparisons were conducted using Tukey's range test.

For error type SC-Error1, a significant effect of both Algorithm (p=4e-7) and Community (p=3e-3), as well as of their interaction (p=4e-5), was found. The mean error for walkers in the Sighted community (13.1%) was found to be significantly smaller than for both the Blind:Cane community (20.8%) and the

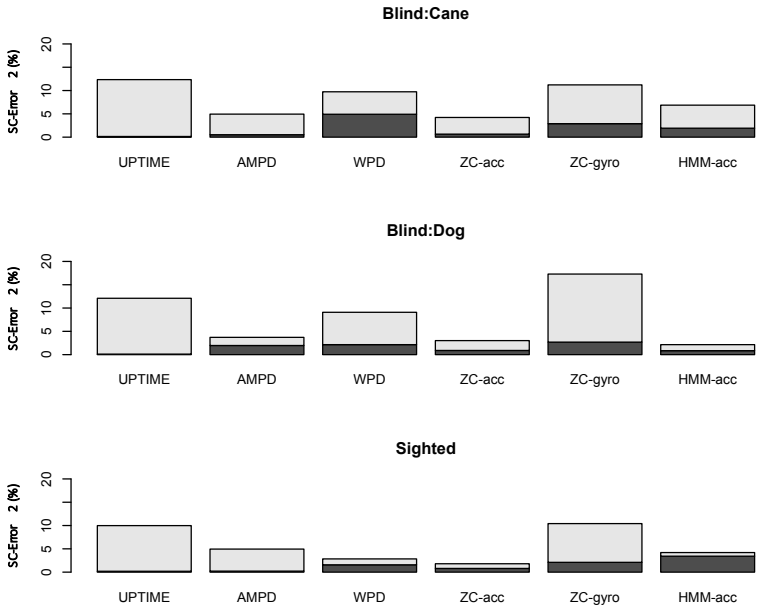


Figure 4.5: SC-Error 2 errors averaged over participants for the different communities and different algorithms considered for the Stratified Leave-One-Out training modality. Gray bars indicate undercount rates; black bars indicate overcount rates.

Blind:Dog community (21.1%). In terms of the algorithms, the mean error of WPD (22.7%) was found to be significantly larger than AMPD (17%), UPTIME (14.9%), and ZC-acc (11.6%, lowest). ZC-acc was also found to have mean error significantly smaller than HMM-acc (23.8%). Since significant interaction was found (p=4e-5), we studied the marginal effects of an algorithm at the different levels of communities. For the Blind:Cane community, the mean error for ZC-acc (7.8%, lowest) was significantly lower than for WPD (24.7%), AMPD (24.9%), and HMM-acc (31.7%). HMM-acc had a significantly higher mean error than UPTIME (16.5%); for the Blind:dog community, both AMPD (8.2%, lowest) and UPTIME (13.7%) had significantly lower mean error than ZC-gyro (38.5%). And for the Sighted community, WPD (20.4%) had significantly higher mean error than both ZC-acc (8.7%, lowest) and ZC-gyro (9.1%).

For error type SC-Error2, a significant effect of both Algorithm (p=5e-11) and Community (p=2e-2), but not their interaction, was found. A significant difference in mean error was found between Blind:Cane (8.7%) and Sighted (5.7%) communities. In terms of the algorithms, the mean error of ZC-acc (3.2%, lowest) was found to be significantly lower than for ZC-gyro (12.2%) and UPTIME (11.5%). UPTIME also had higher mean error than both AMPD (4.7%) and HMM-acc (5.0%). ZC-gyro was found to have higher mean error than WPD (7.3%) and AMPD (4.7%).

In order to study the effect of phone placement on the mean step counting error, we first clustered the various smartphone locations into four groups: pants front pockets (both left and right pockets), pants back pockets (both left and right pockets), jacket pockets (both left and right pockets), and placement of the smartphone at the chest level. Repeated measures ANOVA with blocking factors of Participant and Algorithm found no significant difference in the mean value of either SC-Error1 or SC-Error2 for the different smartphone locations.

4.2.5 Conclusions

Several interesting observations can be drawn from our investigation of step counting algorithms as applied to data in WeAllWalk. Our expectation that people in the different communities considered may have different walking characteristics was indirectly confirmed by the fact that the same step counting algorithms performs differently (in terms of mean error) across these communities. Our data found a significantly larger step counting error for blind walkers using a cane or (for SC-Error 1 only) a guide dog, than for sighted walkers. Among the algorithms considered, no clear winner was found, although our analysis does provide statistical evidence in favor of some techniques. For example, for blind walkers

using a long cane, the ZC-acc algorithm, with a low SC-Error 1 value (7.8%), compared significantly favorably against three other competitors. The same ZC-acc algorithm produced significantly lower mean SC-Error 2 values than two other techniques when tested across all communities of walkers. And our analysis did not find a significant dependence of step counting performance on the location where the iPhone is kept.

4.3 Turn Detection

4.3.1 Introduction

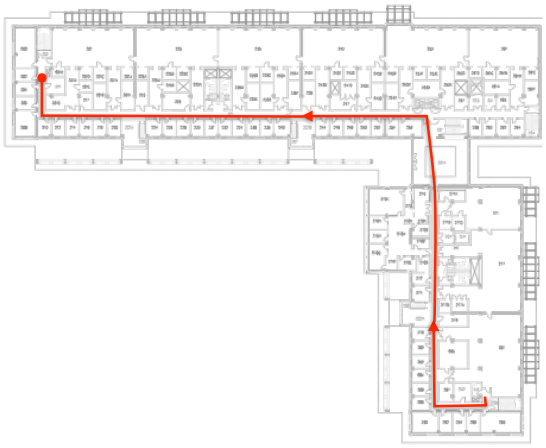
Pedestrian localization and self-tracking applications, both for indoors and outdoors, have become commonplace, enabled by the widespread diffusion of smartphones equipped with multiple sensors. Outdoor localization is normally obtained through GPS fixes, while indoor positioning typically combines beaconing from radio sources (Wi-Fi or low-power Bluetooth) with dead-reckoning from inertial sensors. While all pedestrians may benefit from these systems, they have tremendous potential to enable wayfinding and safe navigation by people with visual impairments or blindness. Indeed, accessible GPS apps specifically designed for blind persons have been on the market for years, and different types of indoor navigation systems are currently being evaluated.

These self-localization systems aim to measure the precise position of the walker and match it against a map of the environment. In some cases, however, a simpler topological description of the path taken may be sufficient to help a blind person re-trace the route taken inside a building and walk safely back to the starting point. A similar system could be useful in multiple situations. Consider for example the case of a blind individual going to a doctor's visit. A

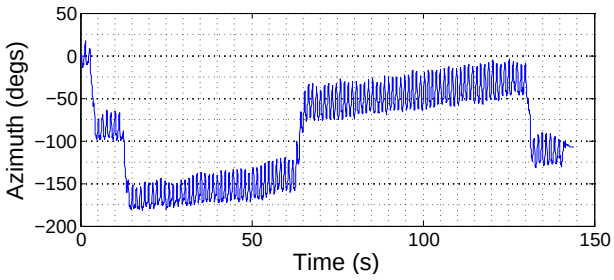
sighted receptionist accompanies her from the waiting room to the doctor's office, located two corridors down. The blind patient's smartphone, which she keeps in her pocket, runs an algorithm to detect and record the sequence of turns taken along the route to the doctor's office, together with the approximate number of steps between turns. After the visit, if no one is available to help, the blind patient consults her smartphone, which reads to her the list of turns taken in reverse order via synthetic speech, allowing her to safely trace her way back to the waiting room. A similar system could also be useful to a blind employee who just started working in an office building he is not familiar with. During orientation, the new employee could use this app to record the routes from his cubicle to key locations (such as the bathroom or the kitchenette), which he can use at a later time for reference until he familiarizes himself with the building.

The advantages of using such a simplified route representation (expressed in terms of turns and step counts) is that it can be obtained without any prior knowledge of the building's floor plan or Wi-Fi footprints, and without the need for wearable sensors (such as sole-mounted IMUs). Upon arrival at a building, a blind user can simply start the app, walk through a path while keeping the smartphone comfortably in his or her pocket, and stop the application at the end. Since the vast majority of buildings have corridors or pathways that intersect at 90° or 45°, only a small discrete set of turn angles needs to be considered.

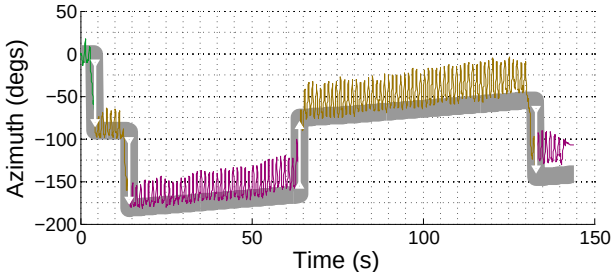
At first sight, it may seem that detecting turns should be quite easy to do, based on the azimuth data measured by the compass or the gyroscope embedded in the phone. But in fact, robust turn detection over extended paths requires careful processing of inertial data. Body sway during gait determines noticeable oscillations of the measured azimuth angle (especially when the smartphone is



(a)



(b)



(c)

Figure 4.6: (a): A floor plan with a path traversed by a walker. (b): The azimuth time series, collected by an iPhone 4 kept in the walker's front pocket. Drift and oscillation due to body sway are apparent. (c) Turns detected by our system. In this and subsequent figures, the white arrows represent turns (the direction and length indicating the turn angle), while the thick gray line represent the drift tracked by our system. The measured data is plotted in different colors, depending on the discrete heading direction as estimated by the system.

kept in one's pocket), and drift (error accumulation through time) is unavoidable over long hauls. Both these effects are visible in Fig. 4.6.

We introduce two algorithms for the robust detection of turns taken by a walker. Our system runs as an iPhone app, and uses attitude data that is produced by the Core Motion framework in iOS. Both algorithms are based on hidden Markov model (HMM) modeling of the measurements. The difference between the two algorithms is in the HMM state representation. In the first algorithm, states represent discrete azimuth angles; in this case, the algorithm could be seen as a robust denoising and quantization technique. This approach is quite straightforward and works well for short paths, but is liable to fail in the case of drift. The second algorithm defines "turns" (transitions between discrete azimuth orientations) as states. By doing so, it allows one to explicitly model drift, through the use of two "drift increment" auxiliary states.

This section is organized as follows. After the related work presented in the next section, we describe our two algorithms in Sec. 4.3.3. The algorithms are then evaluated in Sec. 4.3.4, both quantitatively over labeled data taken multiple indoor and outdoor paths, and qualitatively with data from two blind participants who tested the system indoors. Sec. 4.3.5 has the conclusions and indications for future research.

4.3.2 Related Work

Positioning systems for indoor human navigation have received increasing attention in recent years, and we refer the reader to the several survey articles available [57, 40, 28]. For what concerns blind pedestrians, a few accessible positioning systems are already available available on the market, including accessible GPS (e.g., the Seeing Eye GPS from Sendero Group) [66] and Wi-Fi positioning (e.g., the AXS system by EO Guidage[1]). A localization system for blind travelers

[1] http://eo-guidage.com/eng

developed by indoo.rs[2] and based on iBeacons (low power Bluetooth transmitters) is being tested at the San Francisco Airport. Promising results have also been obtained using a magnetic localization sensor [78].

Dead reckoning based on inertial data allows for positioning without reference to a Wi-Fi network, but is notoriously liable to drift. Good results have been obtained by placing an IMU system at a walker's shoe, and exploiting the so-called "zero-velocity updates", which rely on the fact that when the ambulatory motion of the leg switches from swing to stance, the tracked linear velocity can be safely set to zero, since the foot is normally considered to be static during the stance phase [36, 12]. However, carrying an IMU sensor in one's shoe is impractical (although things may change given the recent emphasis on wearable sensors; already mainstream brands such as Adidas and Nike sell shoes with simple embedded sensors). In order to reduce drift, one may rely on the user to signal when a specific landmark has been reached [6], or reset the system when a turn has been detected [34]. The use of inertial sensors to help a blind person walk straight (without veering) has also been proposed [75]. Finally, we should mention that a different HMM-based algorithm for localization was described in [58]. However, the state representation of this previous system includes spatial locations, whereas our model only represents orientations or turns.

4.3.3 Turn Detection Using HMM

Our algorithms use azimuth data produced by the phone's sensors. In our iPhone implementation, we use the *CMAttitude* object, a property of the *CMMotionManager* class defined in the iOS' Core Motion framework. *CMAttitude* provides the device's attitude with respect to a fixed reference system, using a

[2] http://indoo.rs

proprietary data fusion algorithm processing data from the accelerometer, the gyroscope, and the compass in the device. We don't require any particular placement of the phone (in all our experiments, the phone was kept inside a front pocket of the walker's pants) but assume that the location and orientation of the phone with respect to the walker's body does not change while walking. The phone's attitude at the time the system is started represents the frame of reference for all subsequent measurements. In our experiments, we started the app by pressing a switch in a earphone wire connected to the iPhone, which was already positioned in the users' pocket.

The azimuth angle with respect to a fixed orientation can be obtained by using a reference frame with one axis aligned with gravity (e.g., *CMAttitudeReferenceFrameXArbitraryZVertical* in the Core Motion framework). Another possibility (which can be used without access to the accelerometer) is to apply PCA to the time series of the Euler angles representing the attitude (suitably unwrapped); the largest principal component (characterized by the largest variance) normally corresponds to the azimuth angle. We have used both system in our implementation, with good success.

In very simple cases, turn detection could be implemented by simply thresholding the computed azimuth angle. For example, if the current measured orientation is 15°, and after a while it becomes 100°, we could conclude that a 90° turn took place (remember that we are constraining turns to be multiple of 90° or 45°). However, thresholding the measured azimuth would result in gross errors if the reference frame is not well aligned with the main corridor axes in the building (note that the reference frame normally depends on the orientation of the device when the app was started). Another simple turn detector could be based on the analysis of large azimuth variations, for example by computing and thresholding

the derivative of the attitude angle over a suitable time scale. Being differential in nature, this method is independent of the chosen reference frame.

An example of azimuth time series is shown in Fig. 4.6, along with the path over-imposed on a floor plan. Four 90° turns (right, right, left, right) were taken. The oscillatory character of the data, due to the walker's gait, is apparent. In addition, the data has a very noticeable drift. It should be clear that the naive methods discussed above (thresholding the azimuth data or its derivative) would likely fail here. Drift makes it difficult or impossible to select good thresholds on the azimuth, and thresholding the time derivative of the azimuth data may result in multiple spurious detections due to the gait-induced oscillations. This could be mitigated by smoothing the data with a low-pass filter before derivative computation. However, choosing the correct scale for the smoother is challenging: residual oscillations may cause undesired detections, while over-smoothed transitions may be missed.

Our proposed turn detector models the azimuth time series as a hidden Markov model (HMM). We will actually consider two different HMMs representations. The first one is simpler, but cannot deal with large drift. The second one is designed to also model drifts, but requires a modification of the classic Viterbi algorithm.

We will use the following notation for both algorithms. The measured azimuth value at time t will be denoted by o^t, and the sequence of measurements from time 0 to t will be denoted by $o^{0:t}$. We assume that time periods t take on integer values between 0 and T. Each time instant is endowed with a state s^t, a random variable that takes values in a set $\mathcal{S}$. The event "s^t is equal to the n-th element in $\mathcal{S}$" will be denoted by s_i^t. In our discussion, we will always assume that azimuth data has been unwrapped using any standard method.

4.3.3.1 HMM Turn Detection – Algorithm 1

In this case, $\mathcal{S}$ is formed by a fixed set of azimuth values: $\mathcal{S} = \{0°, \pm 90°, 180°\}$. If diagonal corridors are expected, then the set is augmented with the angles $\{\pm 45°, \pm 135°\}$. Thus, a state represent a quantized version of the heading direction, under the assumption that the reference frame is aligned with the main corridor directions. In our HMM modeling, the time series of states s^t is assumed to form a Markov chain, while the observations o^t are simply noisy measurements of the states:

$$o^t = s^t + n^t \tag{4.1}$$

where n^t is white Gaussian noise. The Markov chain models the correlation between consecutive samples of heading angle, while the noise models short-term azimuth variations due to body sway. Estimation of the HMM parameters (transition probabilities $P(s^t|s^{t-1})$ and noise variance) is discussed later in Sec.4.3.4.3. The Viterbi algorithm computes the sequence of states $s^{0:T}$ (Viterbi path) that maximizes the posterior probability $P(s^{0:T}|o^{0:T})$. The complexity of the algorithm is $T \cdot \|\mathcal{S}\|^2$, which, given the limited number of states in our system, is quite manageable. "Turns" are defined as switches from one state to another state. It's worth emphasizing that our approach is very different from simple quantization of the observations into values of $\mathcal{S}$, possibly after smoothing the data: this simple procedure would take decisions based on *local* observations, while the Viterbi path of states is computed from *global* analysis of the data.

There are two main problems with this HMM representation. The first problem is that the reference frame is not, in general, aligned with the principal corridor directions. In practice, this means that Eq. (4.1) should be changed into

$$o^t = s^t + n^t + \theta^0 \tag{4.2}$$

where θ^0 is a constant but unknown azimuth offset. In order to estimate the offset θ^0, we proceed as follows. Given the sequence of measurements $o^{0:T}$, we run the Viterbi algorithm multiple times on the "biased" measurements $o^{0:T} + \theta_n^0$, where θ_n^0 is a set of fixed bias samples. For example, one could sample values of θ^0 uniformly within a certain interval (e.g., $[-45°, 45°]$). Then, the bias θ_n^0 that results in the highest likelihood $P(o^{0:T}|s^{0:T})$ for the Viterbi path $s^{0:T}$ is chosen, and the associated Viterbi path is produced in the output.

The second problem with this approach is that drift is not modeled, as states represent fixed azimuth values. A possible solution could be to directly include drift in the state. Let D^t be the drift at time t. Eq. (4.1) can be modified as follows to account for drift:

$$o^t = s^t + n^t + D^t \tag{4.3}$$

One could sample the set of possible drift values, and create new states that include drift. More specifically, if $\{s_i\} = \mathcal{S}$ are the original states, and $\{D_n\}$ is the set of sampled drifts, one could consider an expanded states set $\bar{\mathcal{S}} = \{s_i + D_n\}$ for all states s_i and drifts D_n. (Care should be taken when defining the transition probabilities on the Markov chains on the new states set $\bar{\mathcal{S}}$. Considering that drift is slowly changing, the transition probability $P(s_i + D_n|s_j + D_m)$ (which involves a change in drift value) should be smaller than the original transition probability $P(s_i|s_j)$.) The main problem with this approach is that it increases the number of states by a factor equal to the number N_D of drift samples considered. Consequently, the complexity of the Viterbi algorithm increases by a factor ofN_D^2. For this reason, we haven't implemented this variant, and instead devised a new model as described next.

4.3.3.2 HMM Turn Detection – Algorithm 2

In order to explicitly account for drift, we devised a different HMM model for the observed data. In this model, states represent not the azimuth angle, but rather the switch between two discrete azimuth angles. Similarly to the previous case, the state set $\mathcal{S}$ contains 0° (indicated by s_0), ±90°, and 180°. If diagonal corridors are expected, it also contains ±45° and ±135°. However, rather than the actual azimuth, these states represent the difference between the discrete orientation at time t and at time $t-1$. In addition, $\mathcal{S}$ contains two *drift increment* states, d and $-d$. These states represent differences between the azimuth at time t and $t-1$ due solely to drift. A sequence of "differential" states $s^{0:t}$ represents the azimuth angle θ^t as follows:

$$\theta^t = \theta^0 + \sum_{\tau=0}^{t} s^\tau \tag{4.4}$$

The measurement o^t is thus modeled as a noisy version of this "discrete " azimuth angle θ^t:

$$o^t = \theta^t + n^t \tag{4.5}$$

Since two consecutive turns are not physically possible given the fast measurement rate, the transition probability $P(s_i^t|s_j^{t-1})$ between two "differential" states is set to 0 unless j=0. In other words, after a turn, it is assumed that the discrete azimuth θ remains the same for at least one sample more. We make the same assumption for the drift increment state: there cannot be two consecutive drift increments. This also means that $P(s_0^t|s_i^{t-1}) = 1$ for $i \neq 0$. This particular form of the transition probabilities translates into a specific form of the state graphs through time, as shown in Fig. 4.7. This oriented graph represents the possible state paths that can be generated by the model. While this graph would be

fully connected with standard HMM, this is not the case under the constraints described above.

This HMM model, however, has an intrinsic problem: the emission probabilities $P(o^t|s^t)$ are uninformative. In other words, knowledge of a state at a particular time tells very little (or nothing) about the measured azimuth. Indeed, the measurement o^n is a noisy version of the azimuth θ^t (Eq. (4.5)), the latter being a function of *all* previous states $s^{0:t}$, not just of s^t (Eq. (4.4)).

To overcome this problem, one may reason as follows. Recall from basic theory [77] that the Viterbi algorithm computes the following:

$$\arg\max_{s^{0:t}} P(s^{0:T}|o^{0:T}) = \arg\max_{s^{0:T}} P(o^{0:T}|s^{0:T})P(s^{0:T}) \tag{4.6}$$

$$= \prod_{t=1}^{T} P(o^t|s^t)P(s^t|s^{t-1})P(o^0|s^0)P(s^0)$$

under standard assumptions. The Viterbi algorithm computes this maximization in a recursive form, by defining two functions:

$$f^t(s^t) = \max_{s^{t-1}} P(o^{t-1}|s^{t-1})P(s^t|s^{t-1})f^{t-1}(s^{t-1}) \tag{4.7}$$

$$g^t(s^t) = \arg\max_{s^{t-1}} P(o^{t-1}|s^{t-1})P(s^t|s^{t-1})f^{t-1}(s^{t-1})$$

where $f^0(s^0) = P(s^0)$. In practice, $f^t(s^t)$ represents the "score" of the optimal path arriving at s^t, while $g^t(s^t)$ is the state, at time $t-1$, that precedes the state s^t in the optimal path through s^t.

Our proposal is to substitute the uninformative emission probability $P(o^t|s_n^t)$ with the following:

$$P(o^t|s^t, R^t(s^t)) \tag{4.8}$$

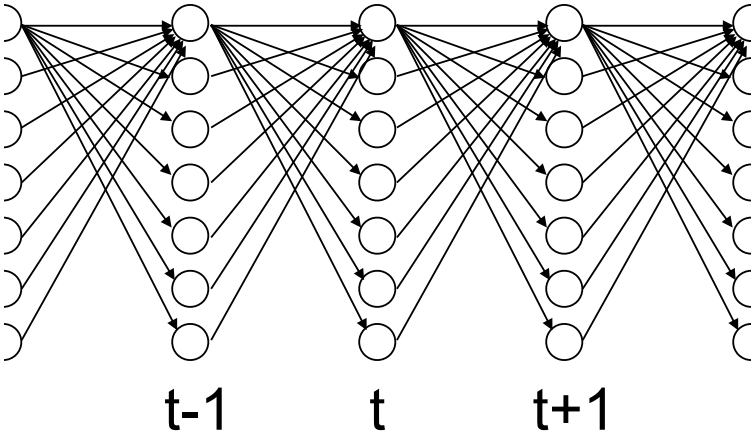


Figure 4.7: The graph of the states $\{s_n^t\}$ used in Algorithm 2. The algorithm produces a directed path from this graph. The first node in each column represents the state s_0.

where

$$R^t(s^t) = \{g^t(s^t), g^{t-1}(g^t(s^t)), \ldots, g^0(g^1(\ldots g^t(s^t)))\} \tag{4.9}$$

Basically, we condition the observation o^t not just on the state s^t, but on the optimal path to s^t, computed based on the measurements $o^{0:t-1}$. To compute the value in (4.8), we can sum together the values of the states in $R^t(s^t)$ and of s^t, obtaining an azimuth value $\hat{\theta}^t$, and then compute $P(o^t|\hat{\theta}^t)$ as by (4.5). In practice, one needs to store at each node s_n^t the sum of the values in the sequence $R^t(s_n^t)$, which can be done recursively.

It is easy to see that the complexity of this algorithm, owing to the special structure of the state graph shown in Fig. 4.7, is linear in the number of states, rather than quadratical.

4.3.3.2.1 Turn Clustering

When implementing our Algorithm 2 using a state set $\mathcal{S}$ that includes $\pm 45°$ and $\pm 135°$, we noticed that turns by $90°$ were often detected as a close sequence of turns by $\pm 45°$, especially when the walker took the turn slowly. We then decided

to introduce a post-processing algorithm that "clusters" together multiple turns detected within an interval of 2 seconds. (This is reasonable since a walker could hardly take two actual turns in such a short time span.) A single turn is produced in lieu of the cluster, with a turn angle equal to the sum of the turns in the cluster, timestamped with the time of occurrence of the first turn in the cluster.

4.3.4 Experiments

4.3.4.1 Data Sets

We took a series of measurements with an experimenter walking on a number of different paths, both indoors (in three different buildings in our campus) and outdoors. Data for 72 paths with multiple turns was collected, 31 of which were used for training (to compute the transition probabilities as well as the value of the drift increments d and of the associated conditional probabilities). In addition, data was collected for 14 straight paths, in order to estimate the variance of the noise n^t due to body sway. The 41 "test" paths (on which the algorithms were assessed) contained an average number of 5 left and 5 right turns each. 18 such path also contained 180° turns, while 4 of them contained 45° turns. The phone was kept in the right front pocket of the experimenter's pants. Azimuth data was collected (along with timestamps) at a rate of 25 readings per second.

This data was labeled with the time stamp and the angle amount of each turn taken. Each time the experimenter took a turn while collecting the data, he or she pressed a switch on a earphone set connected to the iPhone, thus recording the turn. The actual amount of turn angle was recorded after completion of the path, by consulting the map on which the path was traced. Fig. 4.8 shows two examples of paths taken in two different buildings.

Figure 4.8: Examples of indoor paths from our collection.

4.3.4.2 Assessment Metric

To assess the quality of the proposed algorithms, we need to define a metric that compares the output of the turn detector with the labeled data. We chose an approach that exploits the metric properties of the (unwrapped) azimuth angle. Given the sequence of turns (from the output of the algorithm or from the labeled

data), we create an "integral" azimuth time series by accumulating in time the information from the turns. Formally, let (t^n, h^n) be the n-th turn, where t^n represents the time at which the turn took place, and h^n is the turn angle (e.g., $-90°$). Then, the integrated azimuth sequence is given by:

$$\theta^t = \sum_{n:t^n \leq t} h^n \tag{4.10}$$

Based on the integral azimuth sequences built from the labeled data ($\{\theta^t_{\texttt{ld}}\}$) and from the algorithm output ($\{\theta^t_{\texttt{out}}\}$), we define a squared error as follows:

$$E^2 = \sum_{t=0}^{T} (\theta^t_{\texttt{ld}} - \theta^t_{\texttt{out}})^2/(T+1) \tag{4.11}$$

It is clear that if the two sequences of turns (ground truth and estimated) coincide perfectly, then the error is 0. Small differences in the time localization of the same turn will result in small errors. However, if a turn is completely missed, or a spurious turn is detected, the error may become quite large.

4.3.4.3 HMM Parameter Selection

The transition probabilities $P(s^t_i|s^{t-1}_j)$ for Algorithm 1 were assigned based on the corresponding relative frequencies computed in the 31 "training" sequences. In the case of Algorithm 2, the relative frequencies are used for the non-zero transition probabilities, but not for the transitions to the drift increments state d (since no ground truth is available for this). The variance of the noise n was assigned based on the sample variance measured on the 14 straight paths. In order to determine appropriate values for the drift increment d and its associated transition probability $P(d^t|s^{t-1}_0)$, we adopted the following strategy. We first created two sets of values ($\{d_n\}$, $\{p_n\}$) by uniformly sampling the interval of possible drift

increments $[0°, 0.57°]$ with step of $0.0057°$, and the interval of probability values $[0, 1]$ with step of 0.01. For each pair (d_n, p_m), we first assigned $P(d^t|s_0^{t-1}) = P(-d^t|s_0^{t-1}) = p_m$, normalized the set of non-zero transition probabilities to 1, and computed the likelihood of the measurements $P(o^{0:T}|s^{0:T})$ for the Viterbi path. Note that this last step includes maximum likelihood computation of the bias azimuth O described in Sec. 4.3.3.1. The pair (d_n, p_m) that maximized $P(o^{0:T}|s^{0:T})$ was then used in our evaluations with the "test" sequences.

4.3.4.4 Results

Fig. 4.9 shows the cumulative distribution of the errors E, defined in Eq. (4.11), over the test paths for Algorithm 1 and Algorithm 2. For both algorithms, we tested the case with turns taking values multiple of 45° or of 90° only. Additionally, for Algorithm 2, we considered the clustering strategy described in Sec. 4.3.3.2.1 for the 45° turn case. The best performing system on our data is Algorithm 2 when only multiples of 90° are considered, followed by Algorithm 2 with turns multiples of 45° and clustering. This result may be a consequence of the paucity of 45° turns in our collected data. Algorithm 1 is shown to perform generally worse than Algorithm 2. For comparison, we also show the error distribution using a turn detector that simply quantizes (by rounding) the measured azimuth to the closest multiple of 45° or 90°. Quantization to multiples of 45° produces large errors; quantization to multiples of 90° gives results comparable to Algorithm 1 but much worse than Algorithm 2 (when only multiples of 90° are considered).

Fig. 4.10 shows an example with a path processed by various variants of Algorithm 2. Without the clustering mechanism described in Sec. 4.3.3.2.1, 90° turns are almost always detected as pairs of 45° (Fig. 4.10 (b)). This situation is

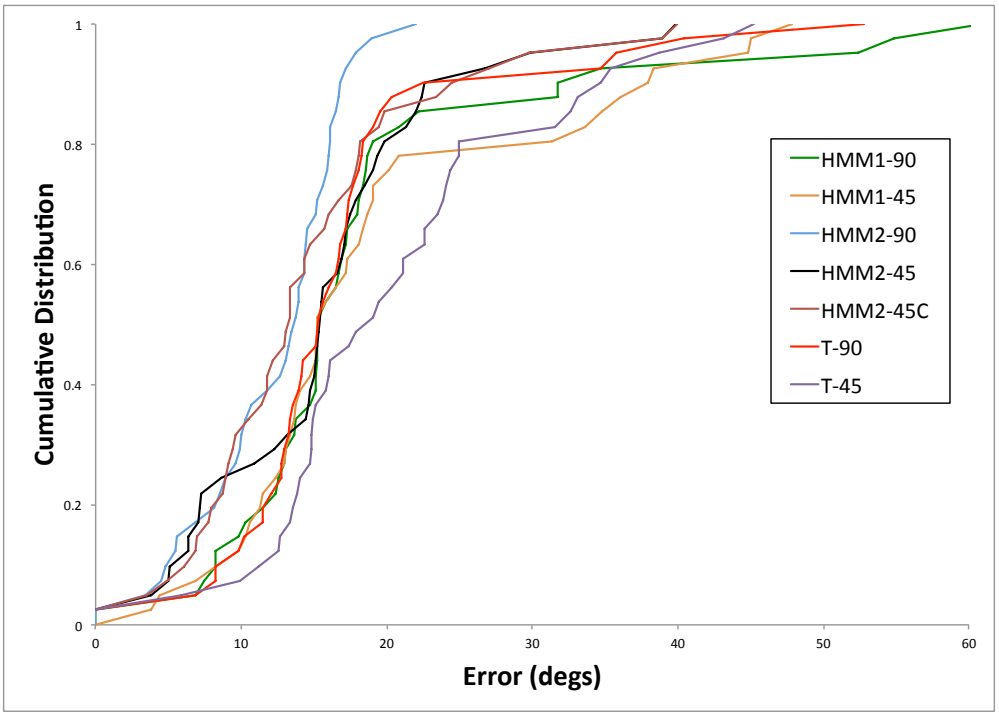
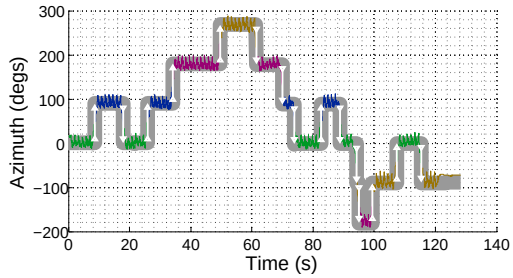


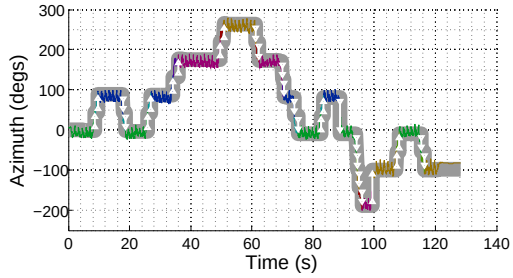
Figure 4.9: Cumulative distribution of the error E (in degrees) defined in Eq. (4.11) over test paths for Algorithm 1 (HMM1) and Algorithm 2 (HMM2). HMM1-90, HMM2-90: only turns that are multiple of 90° considered. HMM1-45, HMM2-90: turns that are multiple of 45° considered. HMM2-45C: clustering postprocessing included (Sec. 4.3.3.2.1). For comparison, we also show results obtained by simple quantization (rounding) the azimuth angle into multiple of 90° (T-90) or 45° (T-45). The cumulative error distribution at a certain error value E_0 represents the proportion of measurements in our dataset for which the error in Eq. (4.11) is less than E_0.

corrected for the most part by the clustering algorithm (Fig. 4.10 (c)). The importance of the correct choice of "bias" θ^0 via the method discussed in Sec. 4.3.3.1 is highlighted by Fig. 4.10 (d), obtained without bias computation ($\theta^0 = 0°$). Note how, without bias correction, multiple turns are incorrectly measured.

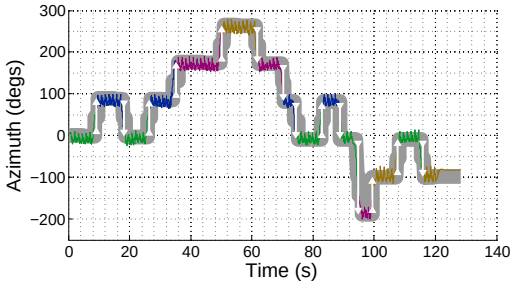
Fig. 4.11 shows three examples with outdoor paths (Algorithm 2, turns by multiple of 45°, clustering). The long stretch with very noticeable drift shown in Fig. 4.11 (a) is well tracked by our system (note how the azimuth θ defined in (4.4), marked by the thick grey line, precisely models the accumulated drift).



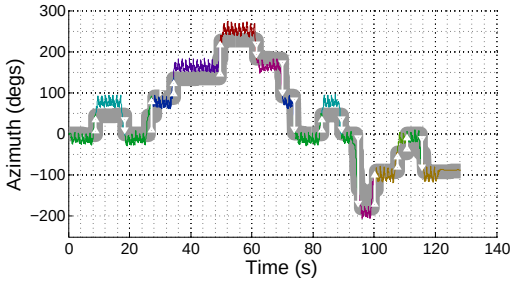
(a)



(b)



(c)



(d)

Figure 4.10: Results on a path using Algorithm 2. (a) Only multiple of 90° turn angles considered. (b) Multiple of 45° turn angles considered. (c) Same as (b), with clustering postprocessing. (d) Same as (c), but without bias computation ($\theta^0 = 0°$).

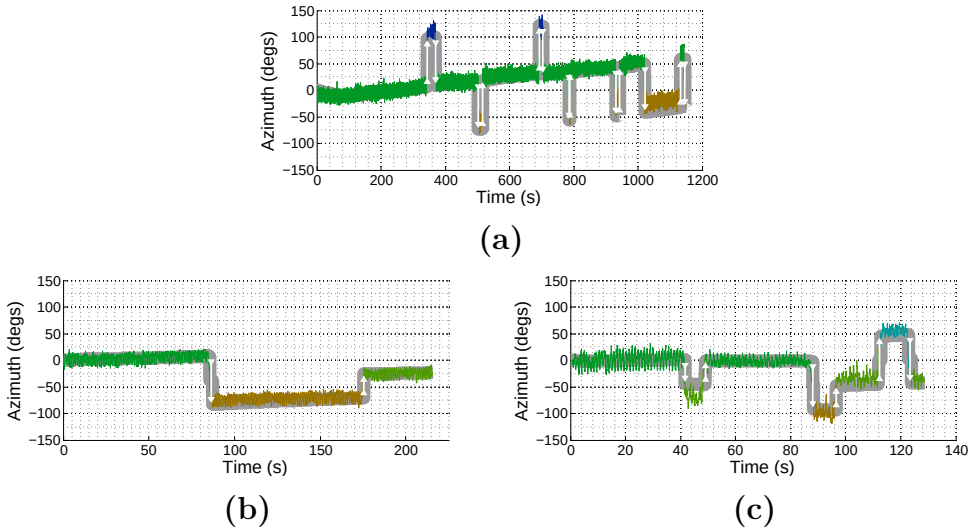
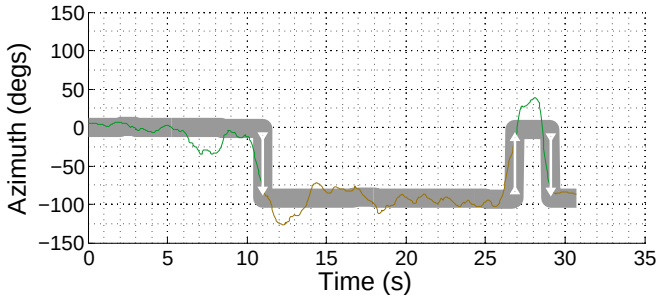


Figure 4.11: Examples of outdoor paths, processed by Algorithm 2 (multiple of 45° turn angles considered, clustering postprocessing).

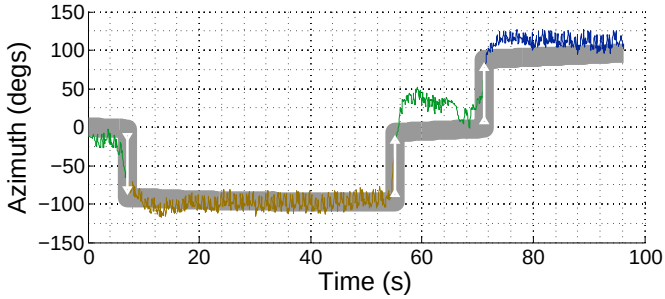
Fig. 4.11 (b) and (c) show example of turns by 45°, which are correctly detected by the algorithm.

Fig. 4.12 shows an example of results with data collected from two blind participants, who tested our system. Since our algorithm is intended to support navigation without sight, it is important that blind walkers be included in the experiments. Both participants used a white cane during data collection. As expected, the gait of blind walkers is different from that of sighted walkers: they tend to be more hesitant while moving forward (especially in areas they are unfamiliar with), often veer in their path [75], and take turns more slowly. Some of these characteristics are visible in the plots of Fig. 4.12.

In addition to the experiments described in Sec. 4.3.4, we trained and tested the HMM1-90 and HMM2-90 models with the WeAllWalk dataset and performed similar statistical analysis as the one performed for the step counter algorithms (see section 4.2.4). The error metric in this case consisted of counting the number of undercount and overcount number of turns by computing the text edit distance



(a)



(b)

Figure 4.12: Examples with data collected by two blind walkers.

between the detected sequence of turns and the ground truth. The overall error was equal to the normalized sum of overcount and undercount rates. In general, HMM1-90 produced a larger rate of overcount turns. HMM2 produced a more balanced rate of undercount and overcount number of turns. The average HMM1-90 error was 54.8%, whereas the average error for HMM2-90 was 42.6%. The error between the three different communities (Blind:Dog (52.5%), Blind:Cane(52.0%), and Sighted (41.7%)) did show a significant error.

4.3.5 Conclusions

We have presented two different algorithms for detecting turns taken by a pedestrian from data collected by an iPhone, which can be conveniently kept in the user's pocket. The phone doesn't need to be at a particular orientation, as

long as its orientation remains constant with respect to the user's body. One limitation of both algorithms is the assumption that the person only takes turns at discrete angles, and thus this approach is appropriate for building with with "standard" floor plans, containing corridors that intersect at 90° or 45°. The best performing algorithm is Algorithm 2 when only multiples of 90° (HMM2-90) are considered.

4.4 Safe Return System

4.4.1 Introduction

Navigating through not well-known or new environments can be challenging and potentially unsafe for persons who are blind. Blind travelers cannot recognize visual landmarks at a distance, cannot preview visible portions of a route, and cannot access visual maps. Although many blind persons are able to move independently through orientation and mobility (O&M) training, traversing through a building for the very first time can lead to getting lost or injured. In order to learn the spatial layout of a place while traversing a route, blind travelers rely heavily on path integration, a mechanism that is known to produce systematic errors [61]. While some blind individuals are able to build fairly precise spatial representations from direct locomotion experience [88], others can develop only a limited understanding of the environment during route traversal [38]. In addition, when visiting a building for the first time, self-orientation without sight may be very challenging, and blind travelers normally rely on a sighted human guide [47, 60]. Developing a technological solution that can support safe blind wayfinding may be very attractive for increased mobility and independence.

In this section, we describe and evaluate a safe return system that can provide some level of assistance when a blind traveler is lost or unsure about how to walk back to a specific location and may not be able to get human assistance. The ability to be able to backtrack a route to its starting point (e.g., entrance to a building or main lobby) is critical for safety reasons, and allows a blind person to move independently in a building, with very minimal assistance or none at all. Furthermore, this work is inspired by the observation that many blind people are able to move independently in a building once they familiarize themselves with the building's layout. This system does not rely on existing maps and is simple to use, allowing the traveler to use the system without interfering with any mobility aid (cane or guide dog) that he or she may be using.

The safe return system runs in an iPhone 6 that is controlled by an Apple Watch. The sensors from the iPhone are used to count steps, detected turns, produce guidance directions in speech form, and communicate with the Apple Watch. In particular, the inertial sensor data provided by the iOS frameworks is used to count steps and detect $\pm 90^\circ$ turns (only $\pm 90^\circ$ turns are measured since most buildings have corridors that intersect at right angles). While a walker traverses a path from an initial location to a destination, the system builds and records a simple representation of the path as a sequence of left and right turns, as well as step counts between turns (see Figure 4.13). Then when the user walks back to the starting point, the system tracks the user's location by counting steps and detecting turns as the user walks, and produces verbal directions in terms of the remaining turns and steps to take to reach the starting location.

Figure 4.13: An example of an indoor path. This path can be described as a sequence of seven straight segments separated by six left or right turns. The way-in path from the East to the West side of the building can be represented by a series of steps and turns (left or right) as follows: 7-L-3-R-35-R-7-L-52-L-5-R-10, where each number represents the length of a segment in steps, and L or R indicate left or right turn. The return path has similar representation, but in reverse order: 10-L-5-R-52-R-7-L-35-L-3-R-7

4.4.2 Related Work

There is increasing interest in technology that can support independent wayfinding in indoor, GPS-denied environments for people who are blind. Some of these technologies involve some sort of infrastructure modifications, such as the placement of infrared beacons [16], RFID tags [23], or Bluetooth Low Energy beacons (iBeacons [1]). Unfortunately, these methods invariably require additional costs for installation, calibration (e.g., RSSI fingerprinting for iBeacons [54]) and/or maintenance, which may hamper their wide diffusion. Technologies that require no environment modifications include mobile computer vision [37, 62, 87] and inertial sensing [22, 26, 27, 76, 79, 34]. Mobile computer vision for wayfinding require use of a camera, either from a smartphone held by hand or attached to the user's garment, or as part of a wearable device (e.g., Google Glass). These systems

may recognize features such as doors or signs, be trained to recognize a specific environment, or be used (possibly in conjunction with a depth sensor) to estimate and track the location of the user using simultaneous localization and mapping (SLAM). Compared to computer vision systems, inertial navigation [50, 97] relies on sensors that, embedded in a smartphone or other devices, can be worn discreetly, require no aiming, and are unaffected by adversarial visual conditions (bad lighting, motion blur, or occluders). Positioning by inertial sensors requires time integration of sensory data (dead reckoning), an operation that invariably results in drift increasing over time. Hence, when used for pedestrian navigation, dead reckoning requires some sort of periodic "reset" such as zero-velocity updates [11] (a technique that is only possible for shoe-mounted sensors), or user-initiated updates when a known landmark is reached [27]. A popular alternative is to use the inertial sensors only for step counting [95] and orientation estimation [67, 33], then integrate this information to estimate the user's position. In benign situations (users walking with consistent stride length, corridors intersecting at right angles), this simple technique might be quite effective. Note that step counting does not seem to be greatly affected by the exact location of the sensor on the walker's body [4]. Virtually all wayfinding systems require prior knowledge of the map of the building to be visited, with the exception of SLAM-based systems that build a map during traversal.

4.4.3 Path Matching

As mentioned in section 4.4.1, the system uses the iPhone sensors to count the number of steps and detect the turns taken as the user walks from an initial location to a destination (e.g., from the main lobby to the doctor's office), and then back to the initial location (e.g., from the doctor's office to the main lobby).

Let the path from the initial location to a destination be the "way-in" or "tracking" phase, and the path from the destination to the initial location be the "return" or "guidance" phase. In both phases, the system determines and records the turns and steps taken along the paths. During way-in phase, the user may be helped by a human guide via verbal directions; may have consulted a tactile map; may remember a prior traversal of the same path; or may simply explore the place; However, during the return phase, the system not only determines and records the number of steps and turns taken, but also determines the current location of the user and provides appropriate guidance instructions. In this section, we describe the technique developed to determine the user's current location during the return phase. It is important to note that although there may be more than one return path, the system only helps the user re-trace the original path.

During the way-in phase, let $TA^{in}[k]$ be the k-th turn amount in degrees ($90°$ or $-90°$), and $S^{in}[k]$ the number of steps between the $(k-1)-th$ and the $k-th$ turn ($S^{in}[1]$ is the number of steps from the starting point). If there are N turns before arriving at a destination, $S^{in}[N+1]$ represents the number of steps between the last turn and the arrival point. The whole path is thus represented as the sequence $Path^{in} = [S^{in}[1], TA^{in}[1], S^{in}[2], TA^{in}[2], ..., S^{in}[N], TA^{in}[N], S^{in}[N+1]]$ (e.g., $Path^{in} = [7 - L - 3 - R - 35 - R - 7 - L - 52 - L - 5 - R - 10]$ for the way-in path in Fig. 4.13).

The return phase begins with the user located at the end point of the way-in path, facing the opposite direction he or she came from, and having the way-in path be in reverse order. This return phase is then represented as: $Path^{ret} = [S^{ret}[1], TA^{ret}[1], S^{ret}[2], TA^{ret}[2], ..., S^{ret}[N], TA^{ret}[N], S^{ret}[N+1]]$, where $S^{ret}[i] = S^{in}[N - i + 2]$ and $TA^{ret}[i] = TA^{in}[N - i + 1]$. During this phase, the system not only counts steps and detects turns, but also tries to match the current location

of the user within the way-in reverse path. Specifically, upon detecting a turn, the path matcher attempts to identify the index k of that turn in $Path^{ret}$. Based on this information, the user can be notified about where he or she stands in the path, and how many waypoints exist until the destination.

If the user were to walk the return path carefully (e.g., taking the same sequence of turns in reverse, and the same number of steps between corresponding turns), path matching would be trivial. However, the path matching module must be able to manage adversarial situations. Let $TA^{det}[i]$ and $S^{det}[i]$ denote the i-th detected turn and step count, respectively, between the (i-1)-th and the i-th detected turns during the return path. Some of the adversarial situations may include measuring a different number of steps between corresponding turns (e.g., $S^{det}[i] \neq S^{ret}[i]$) due to, for example, differences in walking pattern between the way-in and the return phase, due to step counting inaccuracies, or different sequences of turns (e.g., $TA^{det}[i] \neq TA^{ret}[i]$), due to, for example, the user turning too soon and having to correct his or her route (see example in Fig. 4.14). In order to deal with these situations, we initially implemented a path matching algorithm inspired by the longest common subsequence (LCS) problem [13]. Specifically, given the current detected sequence $Path^{det} = [S^{det}[1], TA^{det}[1], ..., S^{det}[K-1], TA^{det}[K]]$ of measured step counts and turns, the algorithm determines a subsequence of turns that matches an initial contiguous subsequence of turns in $Path^{ret}$ while minimizing an overall cost that consists of step count differences and deletions. Given a sorted array I^{det} of indices between 1 and K such that $TA^{det}(I^{det}[i]) = TA^{ret}[i]$, the cost assigned to the subsequence of $Path^{det}$ defined by the indices in I^{det} is equal to cost shown in

Equation 4.12,

$$C(I^{det}) = \sum_{i=1}^{||I^{det}||} \left| \left(\sum_{I^{det}[i-1]+1}^{I^{det}[i]} S^{det}(I^{det}[i]) \right) - S^{ret}(i) \right| + C_{skip}(K - ||I^{det}||) \quad (4.12)$$

where $||Idet||$ represents the length of I^{det}, and by convention $I^{det}[0] = I^{det}[1] - 1$. C_{skip} is the cost assigned to each deletion of a turn in $Path^{det}$. Given $Path^{det}$, the optimal subsequence I^{det} can be found via dynamic programming. The last element of $I^{det}(I^{det}[||I^{det}||])$ represents the estimated index of the last matched turn in $Path^{ret}$. Thus, the system estimates that the K-th detected turn matches the $I^{det}[||I^{det}||]$-th turn of $Path^{ret}$, and can provide further guidance accordingly (e.g., by announcing the number of steps until the next turn in $Path^{ret}$).

This path matching algorithm worked flawlessly when tested with sighted testers. Unfortunately, trials with our first three blind participants revealed a number of design flaws, necessitating a thorough revision of the algorithm. Firstly, in some situations, when the turn sequences and step counts were greatly mismatched between $Path^{ret}$ and $Path^{det}$, it may be necessary to also delete some turns from $Path^{ret}$, and not just from $Path^{det}$ to obtain a correct final match. Secondly, deleting an isolated turn from $Path^{det}$ may result in an inconsistent situation, with the system assuming the user is oriented in the wrong direction and thus providing incorrect guidance. To avoid this situation, the algorithm was revised to compute and record the user's orientation $O^{ret}[i]$ right after the i-th turn, by integrating the turn amounts until that time: $O^{ret}[i] = \sum_{1 \leq i} TA^{ret}[1]$. Likewise, the user's orientation is computed at run time based on the detected turns: $O^{det}[j] = \sum_{1 \leq j} TA^{det}[1]$. The algorithm then computes two matching subsequences of turns, one from $Path^{ret}$ and one from $Path^{det}$, with the constraint that matching turns result in the same orientation. In other words, if I^{ret} and I^{det} represent sorted arrays of indices between 1 and N and 1 and K, respectively,

we impose that $O^{ret}[I^{ret}[k]] = O^{det}[I^{det}[k]]$. Even after imposing equal orientation for after matching turns, though, the system could at times end up finding an optimal subsequence with a final matched turn resulting in an orientation that is different from what is currently measured ($O^{det}[K]$). To avoid this inconsistent situation, we impose that the selected matching subsequences be such that $O^{det}[I^{det}[||I^{det}||]] = O^{det}[K]$. Among all possible matching sequences satisfying these requirements, we select one minimizing the cost in Equation 4.13, which again penalizes discrepancies between step counts, as well as deletions in $Path^{det}$ and deletions in $Path^{ret}$.

$$C(I^{det}, I^{ret}) = \sum_{i=1}^{||I^{det}||} \left| \left(\sum_{I^{det}[i-1]+1}^{I^{det}[i]} S^{det}(I^{det}[i]) \right) - \left(\sum_{I^{ret}[i-1]+1}^{I^{ret}[i]} S^{ret}(I^{ret}[i]) \right) \right| + C_{skip}(K + I^{ret}(||I^{ret}||) - 2||I^{det}||) \quad (4.13)$$

A (not necessarily unique) optimal matching subsequence can be easily computed using dynamic programming, which must be run for each new detected turn. Note that, given the typically small number of turns, computational time is negligible. A simple example of matching subsequence computation is shown in Fig. 4.14 using an oriented graph representation. The state graph construction and path cost determination are shown on the left images, whereas the right images show the return path $Path^{ret}$ (equal to the way-in path $Path^{in}$ in reverse) (shown by thick gray lines), and the currently detected path $Path^{det}$ (shown by a black line with red circles indicating turns (current location is on the dashed line)). The user's orientation $O^{ret}[i], O^{det}[j]$ are shown together with the step counts between turns $S^{ret}[i], S^{det}[j]$ on the vertical axis (return) and horizontal axis (detected) of the state graph. Compatible nodes (i, j) with $O^{ret}[i] = O^{det}[j]$ are shown by squares, with the associated optimal sub-path cost inscribed. Nodes associated

with newly detected turns are shown with dashed contour. Edges are marked with their associated edge cost. Optimal paths terminating at the dummy end node (shown with gray contour) are marked in black. The last node in the optimal path is filled in gray.

In the example of Fig. 4.14, the user took a right turn before its time, ending at a wall. Then, the user turned left before taking the correct right turn. The system incorrectly matched the first turn detected with the first right turn in $Path^{ret}$, and produced an incorrect direction accordingly ("10 steps till a left turn"). However, as soon as the user turned left to correct the route, the system correctly deleted the first right and left detected turns. The direction produced at this time ("3 steps till a right turn") refers to the first segment in $Path^{ret}$, and accounts for the 12 steps already taken by the user. Finally, as the user makes a right turn onto the second segment in the path, the system matches this with the second turn in $Path^{ret}$, producing the correct direction ("10 steps till a left turn").

Each node in the graph represents a potential match between a turn in $Path^{ret}$ and a turn in $Path^{det}$, with a subsequence of turns represented by a path in the graph. Each node is assigned an index (i, j), where i is the index of a turn in $Path^{ret}$ and j is the index of the matching turn in $Path^{det}$. Edge costs combine both penalties in step count discrepancies and deletion costs. Two dummy nodes are included: an initial node with orientation of 0, and a final node. All paths must go from the initial to the end dummy node. The edge from a node (i,j) to the final dummy node has cost proportional to the number of terminal turns deleted from $Path^{det}$ when the orientation at this node is consistent with the current orientation ($O^{ret}[i] = O^{det}[K]$), or infinite cost otherwise (thus inhibiting sequences with inconsistent final orientation).

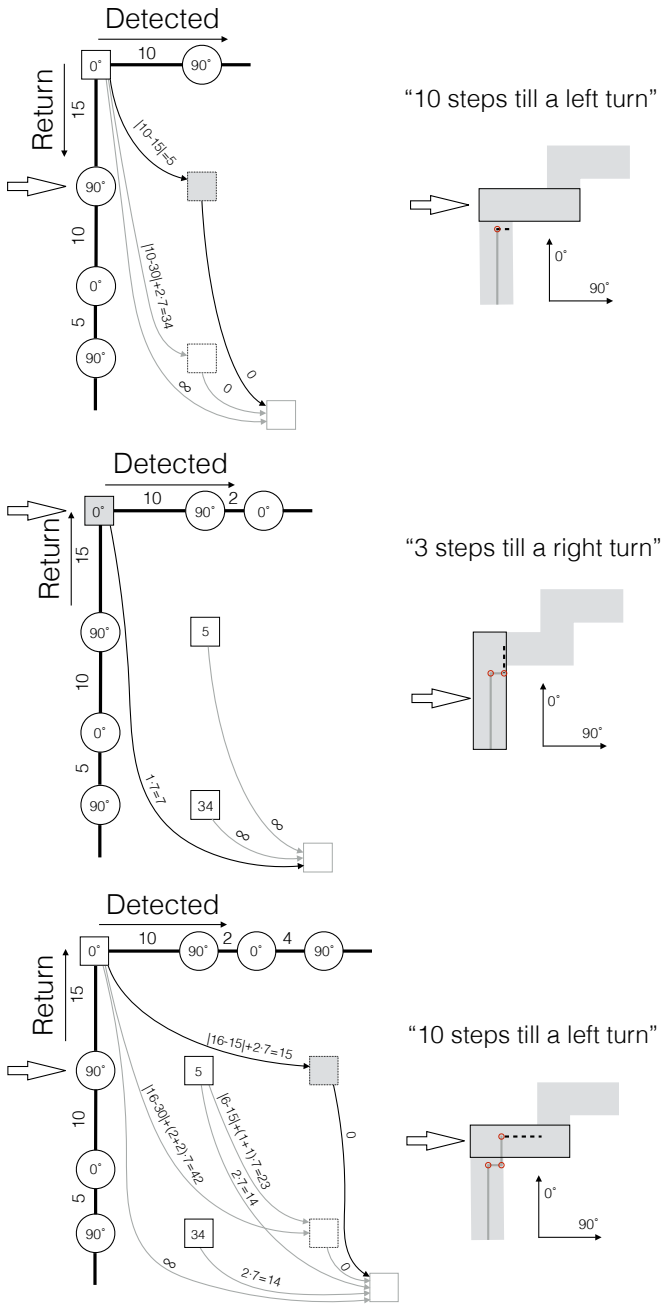


Figure 4.14: Three snapshots during a return path. Left: State graph construction and path cost determination. Right: The return path and the currently detected path. In this example and in the experiments, $C_{skip} = 7$.

4.4.4 System Architecture

4.4.4.1 System Architecture Overview

The safe return system was implemented as an app using the Swift programming language. The iPhone app was responsible for running one of the step counter algorithms described in Sec. 4.2, the turn detection algorithm described in Sec. 4.3, and the path matching algorithm described in Sec. 4.4.3. As mentioned in section 4.4.3, the iPhone ran in two different modes: tracking mode or guidance mode. During the tracking mode, the system recorded the iPhone's sensor data (user acceleration and attitude), and ran a step counter and a turn detection algorithm to determine the turns and the number of steps taken. The number of steps and turns taken were stored in memory and used by the guidance phase. During the guidance phase, the system ran the same step counter and turn detection algorithms to determine, in real-time, the turns and steps taken, and ran the path matching algorithm described in section 4.4.3 to determine the user's current location. Based on the path matching results, guidance messages were synthesized and spoken by the iPhone or Apple Watch. In order to allow the traveler use of the system without interfering with the mobility aid he or she may be using, an app was also developed for an Apple Watch. The Apple Watch performs two main tasks: (1) control the system (e.g., specifying when to start and end tracking, and when to start and end guidance), and (2) provide accessibility information (e.g., query the system to get updated directions). Figure 4.15 shows the safe return system architecture we implemented. The Apple Watch communicated with the iPhone via Bluetooth, and the user was able to control the system by using the Apple Watch Digital Crown and touch screen.

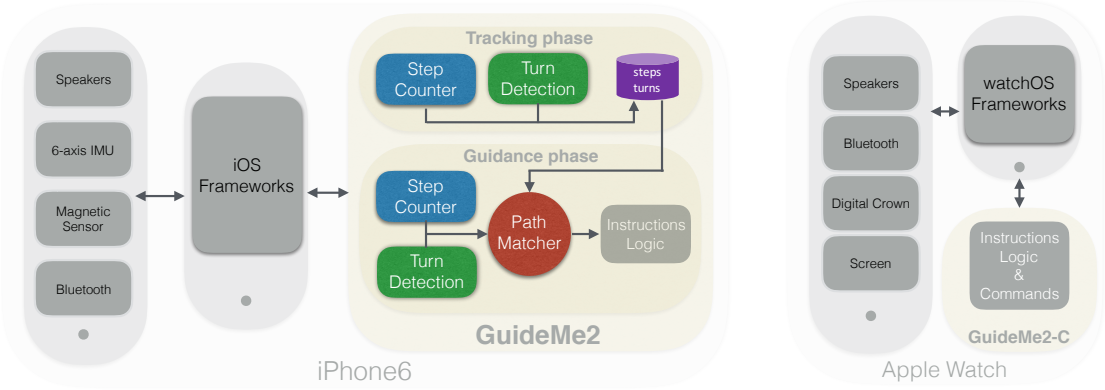


Figure 4.15: Safe return system architecture.

4.4.4.2 User Interface

Users of the system controlled the system via the Apple Watch interface. Users could scroll a menu of options by rotating the Digital Crown; menu options were presented sequentially in speech form, and could be selected by double-tapping the watch's screen. The following options were available: "Track me as I walk" to be selected at the beginning of the way-in phase; "Stop tracking me" to indicate arrival at the way-in path destination; "Guide me back" to initiate the guidance phase; "Stop guiding me" to end the guidance phase; "Restart system" to reinitialize the iPhone 6 and Apple Watch for the next tracking phase; and "Get last instruction" to listen to remaining step counts and turns. The last option was also actionable by a "swipe up" gesture on the Watch's screen, allowing for faster access to the last instruction.

Speech output is generated by the system in five different situations: (1) at the beginning of the guidance phase; (2) right after each detected turn; (3) when the user was estimated to be at 7 steps (or less) away from the next turn or destination; (4) upon estimated arrival to a destination; and (5) whenever the user "swiped up" the Watch's screen. In cases (1)-(3), a sentence is generated informing the user of the number of steps till the next waypoint (left or right turn) or, in the

last segment of the path that leads to the destination. Note that for case (3), the number of steps in the synthesized sentence is actually reduced by two; this is because this sentence is normally produced while the user is walking, and by the time the sentence is completed, the user has already taken approximately two more steps. In case (5), the system speaks the remaining list of step counts and turns till the destination, in addition to the remaining number of steps till the next waypoint.

Audio output from the system was produced by the iPhone and by the Apple Watch in different contexts. Due to the relatively low sound volume generated by the Watch, system-prompted notifications were produced by the iPhone kept in the user's pocket. For users with a hearing impairment, a wired earphone plugged in the iPhone was used. In a more realistic environment, with substantial background noise, earphones or bonephones would be better to use. For user-prompted notifications (case (5)), as well as during browsing of the menu, speech was produced by the Watch itself (users had to moved their wrist close to their faces to wake the Watch up and be able to hear the audio from the Watch.)

4.4.5 Experiments

The safe return system was tested with five participants (two females, 3 males) in different phases of the system. All participants were blind, except for some remaining light perception. Their ages ranged between 25 and 67 (median: 59). One participant (P1) used a guide dog, while the remaining ones used a white long cane. All participants except for P2 were expert travelers. Participant P2 was a recently blind woman who was still perfecting her mobility skills with her long cane. All participants were (in different measures) technology savvy; all were

iPhone users. None of them owned an Apple Watch, one of the reasons being that, at the time of the test, Apple Watch did not support Bluetooth for VoiceOver.

The step counting and turn detection algorithms used in the safe return system were the Automatic Multiscale-based Peak Detection (AMPD) or the variant of the zero crossing technique that uses the de-trended acceleration magnitude (ZC-acce) described in 4.2.3, and the HMM2-90 model described in 4.3.3.2. Note that we used the ZC-acce step counting algorithm on the system used by the last participant, P5, since we noticed that the AMPD algorithm tended to undercount or overcount steps when it was used by the first four participants, P1-P4. These algorithms were trained using the WeAllWalk dataset described in Ch. 3.

Before starting the experiment, and after signing the IRB-approved consent form, each participant was first explained the general concept of the experiment and the functioning of the system. He or she was then handed the iPhone and the Watch, and shown the correct way to interact with the Watch. Then, the participants were asked to rehearse using the system for at least two times on a simple path with two turns until they felt comfortable with its interface and with the tracking and guidance mechanism. At this point, participants were walked to the basement of one of the buildings in our campus (see floor plan in Fig. 4.13), which was the area selected for the experiments. The walkable areas included both wide and narrow corridors containing several turns. All participants were tested with the same sequence of eight paths, shown in Fig. 4.16. Four paths contained two turns (Path 2a-2d); two paths contained three turns (Paths 3a, 3b); and two paths contained four turns (Paths 4a, 4b). The participants used our safe return system only in two of the 2-turn paths, as well as in each of the 3-turn and 4-turn paths (these paths were selected at random for each participant). In the remaining paths, participants were asked to back-track the path without

the system's assistance. Note that we didn't do a within subject experiment since doing so would have reduced the number of trajectories for the same number of trials, thus penalizing diversity. Furthermore, participants may have become exhausted or could have memorized the path or layout, which would have skewed the results. The paths used in this experiment varied in the number of turns, length, and difficulty (some corridors were narrow or contained obstacles).

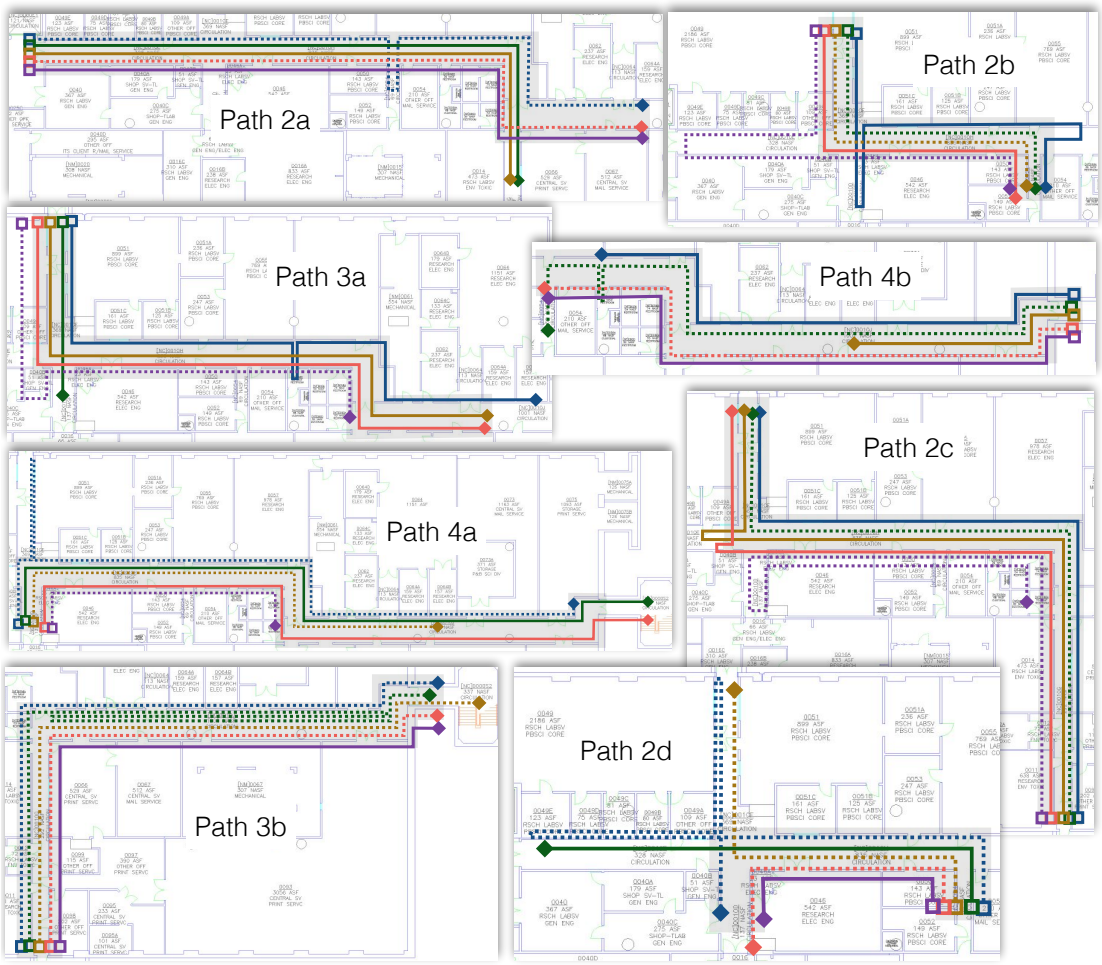


Figure 4.16: Individual return paths taken by the participants. The way-in path is shown by a thick gray line. Each return path started at the location marked by a hollow square, and ended at the location marked by a diamond. Return paths were traversed with (solid line) or without (dashed line) assistance from our system. P1: blue. P2: green. P3: ocher. P4: red. P5: purple.

Figure 4.17: Two of the participants controlling the system via the Apple Watch.

At the beginning of each trial, participants switched the system to the tracking mode by selecting the appropriate entry in the Apple Watch menu, and began walking along the path. During the tracking mode phase, they received verbal directions from the experimenter about when to take a turn and, if necessary, they were warned of potential obstacles. Upon arrival at the destination, participants stopped the tracking mode and turned around. If the path was selected for assisted back-tracing, participants switched the system to guidance mode and began to receive directions from the system. Otherwise, they simply attempted to backtrack the path on their own. At the beginning of the experiment, participants were encouraged to interrogate the system (by swiping up on the Watch's screen) as often as they wanted. Fig. 4.17 shows two participants controlling and interrogating the system via the Apple Watch. Only participant P3 used this option on a regular basis. Fig. 4.16 shows all return paths taken by our participants,

as guided by the system (solid line) or autonomously (dashed line). Note that participants P1-P3 used our earlier version of the path matcher, while P4 and P5 used the improved version. In addition, for each of the paths during the tracking mode phase, participant P5 was asked a random question in order to keep the participant from memorizing the path traveled. This was only done for P5 and not the other participants because we initially felt that it would be too distracting for the participants. However, after participant P4, we felt that it would be good to add this event to ensure the participants were not just memorizing the path. As shown in Fig. 4.16, flagrant cases of system malfunctioning can be noted in Path 2a (Participants P2 and P3), Path 3a (P2), and Path 4b (P3). Deviations in Path 2 and 3a by P1 are due to step undercounting during the return phase, combined with the influence of the guide dog, who sometimes pulled P1 in the wrong direction (see also Path 2a and 4a). Although our participants were for the most part able to back-track the paths autonomously, they did get confused at times, most notably in Path 2d (P1, P3), Path 3b (P2), Path 2c (P5), and Path 2b (P5). At the end of the trials, participants were asked to answer a short questionnaire, with five Likert scale survey questions as shown in Tab. 4.3.

Table 4.3: Exit survey - Likert scale questions (1: I don't agree. - 5: Completely agree). Note that P2 and P4 also expressed a comment in their answer to the third question: *"If it works well", ⋆"I already knew where to go, so it didn't help me".

Question	P1	P2	P3	P4	P5
I think that the system gave me correct information	2	2	2	3	4
I think that the system gave me useful information	3	2	2	3	3
I think that the system made it easier for me to find my way back	2	3	4*	1⋆	3
I think that the system was simple to use	5	5	5	5	4
I enjoyed using the Apple Watch to control the system	5	5	4	5	5

The open-ended questions and a summary of the answers were as follows:

1. Do you think that a safe return system that works as well or better than the one you tested today would be useful in your daily life, or in the daily life of a blind person you know?

    - P1: Absolutely. Especially if it works well, I would start to use it a lot and rely on it, especially for right/left directions.
    - P2: Yes. Working with other people with low vision, they often don't have good orientation. For example, when visiting the doctor's office. I would use it.
    - P3: Yes. Very useful. No need to memorize. There are situations in private places (e.g., inside a restroom) where I am on my own and have no help.
    - P4: I do think so. In certain situations (shopping centers, malls, open spaces, when there are lots of twists/turns). Lots of opportunities.
    - P5: I think so. I don't know if I would use it every day, probably only occasionally, but some people I know would use when the visit buildings or go to class.

2. What functionalities would you like to be added to this system?

    - P1: I would love to save a path and name it and share it. I'd like to access other paths that other people have been tracked on. However, I am not sure I would trust others - maybe I would trust my path more. I'd like to have a "Where am I" function. At least know that I am a couple of turns away.
    - P2: Can you save a location and name it? Would it be an app? Affordable? (We don't need more devices). Only on iPhones? Would like to be able to bookmark a path.

- P3: When it's time to make a turn, I want it to tell me "Make a turn here". Notifications need to be clear (e.g., when I get to destination), maybe with a musical jingle. If I take a wrong turn, I want it to re-route me.
- P4: I would like the ability to pause/resume tracking or guidance (e.g., if I stop to talk with someone.) It should also be robust for situations such as bumping into someone. Also, I would like the ability to calibrate the device for the particular stride of the user.
- P5: In the finished version, I'd like to have a setting where every 15 seconds or so the system would give me the next instruction (like an auto-repeat). Also, I'd like it to identify 45° turns.

3. Any additional comments or feedback?

- P3: When I walk, I must integrate all five senses, not just counting steps. Counting steps is just a part of it, and I don't always do it. When I hear "12 steps till a turn", I take it seriously. It is difficult not to trust the system. It needs to be precise. If the error in step counting is 100%, then it is completely useless.
- P4: Good and easy to use, especially when Bluetooth will be supported with Voiceover on the Watch.
- P5: Good start. Seems to not be very precise in the number of steps it computes (overcompensates). It's good that you don't need cellular data, as sometimes there is no connectivity. Does it need the Watch? Because not everyone has one. The finished version should have Braille support for people who are deaf/blind.

4.4.6 Discussion and Conclusions

As discussed in the previous sections, the development of the safe return system followed a typical design cycle scheme. The system worked perfectly with sighted users, but there were several design flaws when tested with blind users (as noted by the modest Likert ratings given by P1-P3 to the first two questions in Table 4.3, which addressed the technical aspects of the system). A redesign of the system was necessary to address each of the issues noted in the first test. The final version of the system worked very well when tested by participants P4 and P5, even though it sometimes undercounted steps during the return phase, resulting in turn directions given sometimes too late (this may explain the average ratings given by P4 and P5 to the first two questions).

One important observation from these early experiments is that even the best of systems will produce some errors, such as undercounting or overcounting steps, resulting in occasional incorrect directions (see an example in Fig. 4.14). For the system to be useful, the user needs to understand these limitations, and find an acceptable way to use the information from the system while ultimately being in charge of taking a decision. This is by no means specific to our safe return system. Each one of the participants quickly found a way to use the (possibly incorrect) information from the system with their own (possibly inaccurate) sense of orientation. However, as noted by P3, it is not always easy to decide when or when not to trust the system. The system did not provide the exact time when a turn should be taken due to the inaccuracy of the step counting component. Instead, the system simply notifies the user when a turn is approaching, a few steps in advance. Users are then in charge of finding exactly when it is possible to turn. Nonetheless, it is critical that the system recovers from momentary errors

(e.g., taking a wrong turn). The path matching algorithm was designed to provide this level of robustness.

As it can be seen in Figure 4.16, most participants were able to correctly backtrack most of the paths even without assistance from the system. In particular, participant P4, an expert traveler, was able to perfectly re-trace all of his paths. He admitted that, during the trials with the safe return system active, he was for the most part, monitoring how the system worked, rather than relying on it for assistance. We are very encouraged by the responses given by our participants, who unanimously declared that, in their opinion, this system, if well functioning, could be very useful (see answers to the first open-ended question). As shown in Table 4.3, the score given for correctness and usefulness of information provided by the system was between 2 and 3. In terms of the system making it easier to find his or her way back, the Likert score range between 1-4. This is in part due to the system evolving during the tests, and as pointed out previously, the system tended to undercount or overcount steps, thus resulting in occasional incorrect directions. Lastly, the participants were very happy with the system's ease of use and with the Watch-based interface (see answers to the last two questions in Table 4.3). The Apple Watch made it easy to control the system without having to stop and take out the iPhone.

The safe return system described in this chapter works well in indoor environments where hallways intersect at 90°, and where paths are mostly rectilinear and void of many obstacles and crowded places (e.g., hospitals, office buildings, grocery stores). However, in places where large crowds or non-static obstacles (e.g, suit cases, shopping bags) might be present at all times (e.g., airport terminals, shopping malls during busy hours), the system might not perform well due to the need to take arbitrary trajectories in order to avoid running into people or

obstacles. In these cases, a blind person might need to stop multiple times, take multiple turns that are not 90° in order to go around obstacles, and possibly take multiple short steps within the same location. All these events would be make the system over or under count the number of steps and turns, which could make the safe return system unreliable. In recent years, wireless technologies (e.g., Wi-Fi, ZigBee, Bluetooth and Bluetooth Low Energy (BLE)) have become widely available in different hardware products (e.g., iBeacons, routers) that can be found in some commercial and residential areas. Applications, services, and products that make use of these hardware devices have proven to be fairly accurate at pinpointing the location of a person or object. However, these systems require the hardware devices to be working properly and be present in all areas of interest, which might not always be the case due to recurring costs of keeping the devices working properly. In the cases in which an infrastructure might already be in place in a building (e.g., BLE beacons), the safe return system presented in this chapter might not be needed. However, it is important to note that not all public or private places are equipped with wireless devices, and in some cases, these devices might not work properly (e.g., due to battery decay) or might be unreliable (due to signal interference and radio propagation effects). In these cases, the system presented in this chapter can aid these systems that use wireless devices or other similar technologies when they become unreliable or when no hardware infrastructure is in place.

This chapter described all the components that were implemented to build a system that can assist a blind person who attempts to back-track a path taken in a building. The safe return system was implemented as an iPhone app controlled by an Apple Watch. The system is easy to use, and users don't need to interact with the iPhone while walking (which could be problematic when handling a long

cane or a guide dog). All the participants found great potential in the safe return concept, and were enthusiastic about the user interface designed around the Apple Watch.

Chapter 5

Conclusion

This dissertation describes the research completed to develop robust and safe guidance systems that provides real-time and reliable information to blind and visually impaired travelers directly from their smart devices. The research began with a novel system developed to convey travel-related information to blind passengers when using public transportation. The user study conducted to evaluate this system shed light on a number of accessibility issues that needed to be addressed when building a personal navigation system. In general, personal navigation systems must be able to provide accurate spatial and directional information, while still being able to deal with adverse situations that may be encountered throughout a path (e.g., running into obstacles, taking the wrong route). Inspired by this research, our focus went into developing an indoor guidance system for the blind that was robust and that required no external information or specialized hardware (e.g., building map, Wi-Fi, or beacons). The goal was to build a system that would help overcome many of the challenges faced by the visually impaired community when they travel to unfamiliar indoor environments. Some of the challenges include (1) the need to be able to safely re-trace a route taken inside a building to go back to a starting location, and (2) the system must be able to

take into consideration the differences in gait and upper body motion, as well as be able to handle other adverse situations. In order to deal with different gait and body motion patterns, we built a data set of inertial time series measurements (WeAllWalk), and used it to train and evaluate the turn detector and step counter components of the guidance system. The turn detector robustly detects turns, a step counter counts the steps taken along a path, and a path matching algorithm determines the user's current location. We devised two turn detection algorithms based on HMMs to robustly detect turns even in the presence of drift in the inertial sensor measurements and noticeable body sway during gait. We evaluated several step accounting algorithms, and based on their accuracy we selected the best algorithms to use in the robust path back-tracing guidance system. And we devised a path matching algorithm to determine the user's current location. The guidance system was implemented as an app and tested it on an iPhone6 that was controlled by an Apple Watch. Based on the user study we conducted with blind participants, the system was able to recover from momentary errors, and we received positive feedback with regards to the usability and functionality of the system. We hope that the research presented in this dissertation inspires the next generation of blind wayfinding systems to help blind people safely navigate indoor and outdoor environments.

Bibliography

[1] Dragan Ahmetovic, Cole Gleason, Kris M Kitani, Hironobu Takagi, and Chieko Asakawa. Navcog: turn-by-turn smartphone navigation assistant for people with visual impairments or blindness. In *Proceedings of the 13th Web for All Conference*, page 9. ACM, 2016.

[2] Moustafa Alzantot and Moustafa Youssef. Uptime: Ubiquitous pedestrian tracking using mobile phones. In *Wireless Communications and Networking Conference (WCNC), 2012 IEEE*, pages 3204–3209. IEEE, 2012.

[3] Michael Angermann, Patrick Robertson, Thomas Kemptner, and Mohammed Khider. A high precision reference data set for pedestrian navigation using foot-mounted inertial sensors. In *Indoor Positioning and Indoor Navigation (IPIN), 2010 International Conference on*, pages 1–6. IEEE, 2010.

[4] Ilias Apostolopoulos, Daniel S Coming, and Eelke Folmer. Accuracy of pedometry on a head-mounted display. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 2153–2156. ACM, 2015.

[5] Ilias Apostolopoulos, Daniel S. Coming, and Eelke Folmer. Accuracy of pedometry on a head-mounted display. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 2153–2156, New York, NY, USA, 2015. ACM.

[6] Ilias Apostolopoulos, Navid Fallah, Eelke Folmer, and Kostas E Bekris. Integrated online localization and navigation for people with visual impairments using smart phones. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 3(4):21, 2014.

[7] Mohamed Attia, Adel Moussa, and Naser El-Sheimy. Map aided pedestrian dead reckoning using buildings information for indoor navigation applications. *Positioning*, 2013, 2013.

[8] Shiri Azenkot and Emily Fortuna. Improving public transit usability for blind and deaf-blind people by connecting a braille display to a smartphone.

In *Proceedings of the 12th international ACM SIGACCESS conference on Computers and accessibility*, pages 317–318. ACM, 2010.

[9] Michel Banâtre, Paul Couderc, Julien Pauty, and Mathieu Becus. Ubibus: Ubiquitous computing to help blind people in public transport. In *International Conference on Mobile Human-Computer Interaction*, pages 310–314. Springer, 2004.

[10] S Barbeau, J Labrador, PL Winters, H Perez, and NL Georggi. The travel assistant device: Utilizing gps-enabled mobile phones to aid transit rides with special needs. In *Proc. 15th World Congress on Intelligent Transportation Systems, NY*, 2008.

[11] Stéphane Beauregard. Omnidirectional pedestrian navigation for first responders. In *Positioning, Navigation and Communication, 2007. WPNC'07. 4th Workshop on*, pages 33–36. IEEE, 2007.

[12] Stephane Beauregard and Harald Haas. Pedestrian dead reckoning: A basis for personal positioning. In *Proceedings of the 3rd Workshop on Positioning, Navigation and Communication*, pages 27–35, 2006.

[13] Lasse Bergroth, Harri Hakonen, and Timo Raita. A survey of longest common subsequence algorithms. In *String Processing and Information Retrieval, 2000. SPIRE 2000. Proceedings. Seventh International Symposium on*, pages 39–48. IEEE, 2000.

[14] BB Blasch, SJ LaGrow, and WR De l'Aune. Three aspects of coverage provided by the long cane: Object, surface, and foot-placement preview. *Journal of Visual Impairment and Blindness*, 90:295–301, 1996.

[15] S Bohonos, A Lee, A Malik, C Thai, and Roberto Manduchi. Universal real-time navigational assistance (urna): an urban bluetooth beacon for the blind. In *Proceedings of the 1st ACM SIGMOBILE international workshop on Systems and networking support for healthcare and assisted living environments*, pages 83–88. ACM, 2007.

[16] John Brabyn, William Crandall, and William Gerrey. Talking signs: a remote signage, solution for the blind, visually impaired and reading disabled. In *Engineering in Medicine and Biology Society, 1993. Proceedings of the 15th Annual International Conference of the IEEE*, pages 1309–1310. IEEE, 1993.

[17] Agata Brajdic and Robert Harle. Walk detection and step counting on unconstrained smartphones. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pages 225–234. ACM, 2013.

[18] Alasdair Cain. Design elements of effective transit information materials. In *2005 Bus & Paratransit Conference*, 2005.

[19] Giuseppe Caso, Luca De Nardis, Andrea Ferrante, and Maria Gabriella Di Benedetto. 2013 international conference on indoor positioning and indoor navigation, ipin 2013. In *IEEE Computer Society*, 2013.

[20] Vivek Chandel, Anirban Dutta Choudhury, Avik Ghose, and Chirabrata Bhaumik. Actrak-unobtrusive activity detection and step counting using smartphones. In *International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services*, pages 447–459. Springer, 2013.

[21] Chao Chen, Steve Anton, and Abdelsalam Helal. A brief survey of physical activity monitoring devices. *University of Florida Tech Report MPCL-08-09*, 2008.

[22] Diansheng Chen, Wei Feng, Qiteng Zhao, Muhua Hu, and Tianmiao Wang. An infrastructure-free indoor navigation system for blind people. *Intelligent Robotics and Applications*, pages 552–561, 2012.

[23] Sakmongkon Chumkamon, Peranitti Tuvaphanthaphiphat, and Phongsak Keeratiwintakorn. A blind navigation system using rfid for indoor environments. In *Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, 2008. ECTI-CON 2008. 5th International Conference on*, volume 2, pages 765–768. IEEE, 2008.

[24] Gastone Ciuti, Leonardo Ricotti, Arianna Menciassi, and Paolo Dario. Mems sensor technologies for human centred applications in healthcare, physical activities, safety and environmental sensing: a review on research activities in italy. *Sensors*, 15(3):6441–6468, 2015.

[25] James Coughlan, Roberto Manduchi, and Huiying Shen. Cell phone-based wayfinding for the visually impaired. In *1st International Workshop on Mobile Vision*, 2006.

[26] M Bernardine Dias, Aaron Steinfeld, and M Beatrice Dias. *Future directions in indoor navigation technology for blind travelers*. Boca Raton, FL: CRC Press, 2015.

[27] Navid Fallah, Ilias Apostolopoulos, Kostas Bekris, and Eelke Folmer. The user as a sensor: navigating users with visual impairments in indoor spaces using tactile landmarks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 425–432. ACM, 2012.

[28] Navid Fallah, Ilias Apostolopoulos, Kostas Bekris, and Eelke Folmer. Indoor human navigation systems: A survey. *Interacting with Computers*, 25(1):21–33, 2013.

[29] Brian Ferris, Kari Watkins, and Alan Borning. Location-aware tools for improving public transit usability. *IEEE Pervasive Computing*, 9(1):13–19, 2010.

[30] German Flores, Benjamin Cizdziel, Roberto Manduchi, Katia Obraczka, Julie Do, Tyler Esser, and Sri Kurniawan. Transit information access for persons with visual or cognitive impairments. In *International Conference on Computers for Handicapped Persons*, pages 403–410. Springer, 2014.

[31] German Flores and Roberto Manduchi. Experiments with a public transit assistant for blind passengers. In *International Conference on Computers Helping People with Special Needs*, pages 43–50. Springer, 2016.

[32] German H Flores and Roberto Manduchi. Weallwalk: An annotated data set of inertial sensor time series from blind walkers. In *Proceedings of the 18th International ACM SIGACCESS Conference on Computers and Accessibility*, pages 141–150. ACM, 2016.

[33] Germán H Flores, Roberto Manduchi, and Enrique D Zenteno. Ariadne's thread: Robust turn detection for path back-tracing using the iphone. In *Ubiquitous Positioning Indoor Navigation and Location Based Service (UPINLBS), 2014*, pages 133–140. IEEE, 2014.

[34] Jesus Zegarra Flores and René Farcy. Indoor navigation system for the visually impaired using one inertial measurement unit (imu) and barometer to guide in the subway stations and commercial centers. In *Computers Helping People with Special Needs*, pages 411–418. Springer, 2014.

[35] Randal C Foster, Lorraine M Lanningham-Foster, Chinmay Manohar, Shelly K McCrady, Lana J Nysse, Kenton R Kaufman, Denny J Padgett, and James A Levine. Precision and accuracy of an ankle-worn accelerometer-based pedometer in step counting and energy expenditure. *Preventive medicine*, 41(3):778–783, 2005.

[36] Eric Foxlin. Pedestrian tracking with shoe-mounted inertial sensors. *Computer Graphics and Applications, IEEE*, 25(6):38–46, 2005.

[37] Cole Gleason, Anhong Guo, Gierad Laput, Kris Kitani, and Jeffrey P Bigham. Vizmap: Accessible visual information through crowdsourced map reconstruction. In *Proceedings of the 18th International ACM SIGACCESS Conference on Computers and Accessibility*, pages 273–274. ACM, 2016.

[38] Reginald G Golledge. Human wayfinding and cognitive maps. *Wayfinding behavior: Cognitive mapping and other spatial processes*, pages 5–45, 1999.

[39] Susan Vincent Graser, Robert P Pangrazi, and William J Vincent. Effects of placement, attachment, and weight classification on pedometer accuracy. *Journal of Physical Activity and Health*, 4(4):359–369, 2007.

[40] Yanying Gu, Anthony Lo, and Ignas Niemegeers. A survey of indoor positioning systems for wireless personal networks. *Communications Surveys & Tutorials, IEEE*, 11(1):13–32, 2009.

[41] Markus Guentert. Improving public transit accessibility for blind riders: a train station navigation assistant. In *The proceedings of the 13th international ACM SIGACCESS conference on Computers and accessibility*, pages 317–318. ACM, 2011.

[42] Kotaro Hara, Shiri Azenkot, Megan Campbell, Cynthia L Bennett, Vicki Le, Sean Pannella, Robert Moore, Kelly Minckler, Rochelle H Ng, and Jon E Froehlich. Improving public transit accessibility for blind riders by crowdsourcing bus stop landmark locations with google street view: An extended analysis. *ACM Transactions on Accessible Computing (TACCESS)*, 6(2):5, 2015.

[43] JC Aguilar Herrera, Paul-Gerhard Plöger, André Hinkenjann, Jens Maiero, M Flores, and A Ramos. Pedestrian indoor positioning using smartphone multi-sensing, radio beacons, user positions probability map and indoorosm floor plan representation. In *Indoor Positioning and Indoor Navigation (IPIN), 2014 International Conference on*, pages 636–645. IEEE, 2014.

[44] Michael Horvat, Christopher Ray, Vincent K Ramsey, Tanya Miszko, Roger Keeney, and Bruce B Blasch. Compensatory analysis and strategies for balance in individuals with visual impairments. *Journal of Visual Impairment and Blindness*, 97(11):695–703, 2003.

[45] Andreas Hub. Precise indoor and outdoor navigation for the blind and visually impaired using augmented maps and the tania system. In *Proceedings of the 9th International Conference on Low Vision*, pages 2–5, 2008.

[46] R Jackson. In what pocket do you keep your phone?, 2013.

[47] William H Jacobson. Orientation and mobility. In *Assistive Technology for Blindness and Low Vision*, pages 29–58. CRC Press, 2012.

[48] Sampath Jayalath, Nimsiri Abhayasinghe, and Iain Murray. A gyroscope based accurate pedometer algorithm. In *International Conference on Indoor Positioning and Indoor Navigation*, volume 28, page 31st, 2013.

[49] Amy A Kalia, Gordon E Legge, Rudrava Roy, and Advait Ogale. Assessment of indoor route-finding technology for people with visual impairment. *Journal of visual impairment & blindness*, 104(3):135, 2010.

[50] Wonho Kang and Youngnam Han. Smartpdr: Smartphone-based pedestrian dead reckoning for indoor localization. *IEEE Sensors journal*, 15(5):2906–2916, 2015.

[51] Seung Young Kim and Gu-In Kwon. Interrupt-based step-counting to extend battery life in an activity monitor. *Journal of Sensors*, 2016, 2015.

[52] Lukas Köping, Marcin Grzegorzek, and Frank Deinzer. Probabilistic step and turn detection in indoor localization. 2014.

[53] Jana Kosecka, Liang Zhou, Philip Barber, and Zoran Duric. Qualitative image based localization in indoors environments. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–II. IEEE, 2003.

[54] Pavel Kriz, Filip Maly, and Tomas Kozel. Improving indoor localization using bluetooth low energy beacons. *Mobile Information Systems*, 2016, 2016.

[55] Quentin Ladetto and Bertrand Merminod. In step with ins: Navigation for the blind, tracking emergency crews. *Gps World*, 13(10), 2002.

[56] R Lanzerotti. Blind and low vision priority project. *Report from the Light-House for the Blind and San Francisco Mayors Office on Disability*, 2007.

[57] Hui Liu, Houshang Darabi, Pat Banerjee, and Jing Liu. Survey of wireless indoor positioning techniques and systems. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 37(6):1067–1080, 2007.

[58] Jingbin Liu, Ruizhi Chen, Ling Pei, Robert Guinness, and Heidi Kuusniemi. A hybrid smartphone indoor positioning solution for mobile lbs. *Sensors*, 12(12):17208–17233, 2012.

[59] Sharon A Livingstone-Lee, Ronald W Skelton, and Nigel Livingston. Transit apps for people with brain injury and other cognitive disabilities: the state of the art. *Assistive Technology*, 26(4):209–218, 2014.

[60] Richard G Long and EW Hill. Establishing and maintaining orientation for mobility. *Foundations of orientation and mobility*, 1, 1997.

[61] Jack M Loomis, Roberta L Klatzky, Reginald G Golledge, Joseph G Cicinelli, James W Pellegrino, and Phyllis A Fry. Nonvisual navigation by blind and sighted: assessment of path integration ability. *Journal of Experimental Psychology: General*, 122(1):73, 1993.

[62] Roberto Manduchi, Sri Kurniawan, and Homayoun Bagherinia. Blind guidance using mobile computer vision: A usability study. In *Proceedings of the 12th international ACM SIGACCESS conference on Computers and accessibility*, pages 241–242. ACM, 2010.

[63] Andrea Mannini and Angelo Maria Sabatini. A hidden markov model-based technique for gait segmentation using a foot-mounted gyroscope. In *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE*, pages 4369–4373. IEEE, 2011.

[64] Michael Marschollek, Mehmet Goevercin, Klaus-Hendrik Wolf, Bianying Song, Matthias Gietzelt, Reinhold Haux, and Elisabeth Steinhagen-Thiessen. A performance comparison of accelerometry-based step detection algorithms on a large, non-laboratory sample of healthy and mobility-impaired persons. In *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*, pages 1319–1322. IEEE, 2008.

[65] Sarah J Mason, Gordon E Legge, and Christopher S Kallie. Variability in the length and frequency of steps of sighted and visually impaired walkers. *Journal of visual impairment & blindness*, 99(12):741, 2005.

[66] Michael May and Kim Casey. *Accessible Global Positioning Systems (GPS).* In Assistive technology for blindness and low vision, Manduchi, R. and Kurniawan, S., eds., CRC Press, 2012.

[67] Nesma Mohssen, Rana Momtaz, Heba Aly, and Moustafa Youssef. It's the human that matters: accurate user orientation estimation for mobile computing applications. In *Proceedings of the 11th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, pages 70–79. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2014.

[68] S Murphy Kelly. San francisco airport tests beacon sensors to guide blind travelers, 2014.

[69] Kazuhiro Nakamura, Yoshiyuki Aono, and Yoshiaki Tadokoro. A walking navigation system for the blind. *Systems and computers in Japan*, 28(13):36–45, 1997.

[70] Najme Zehra Naqvib, Ashwani Kumar, Aanchal Chauhan, and Kritka Sahni. Step counting using smartphone-based accelerometer. *International Journal on Computer Science and Engineering*, 4(5):675, 2012.

[71] Thanh Trung Ngo, Yasushi Makihara, Hajime Nagahara, Yasuhiro Mukaigawa, and Yasushi Yagi. The largest inertial sensor-based gait database and performance evaluation of gait-based personal authentication. *Pattern Recognition*, 47(1):228–237, 2014.

[72] World Health Organization et al. Draft action plan for the prevention of avoidable blindness and visual impairment 2014–2019. towards universal eye health: a global action plan 2014–2019. *Proceedings of the 66th World Health Assembly*, 28, 2013.

[73] Thomas Olutoyin Oshin and Stefan Poslad. Ersp: An energy-efficient real-time smartphone pedometer. In *Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on*, pages 2067–2072. IEEE, 2013.

[74] Meng-Shiuan Pan and Hsueh-Wei Lin. A step counting algorithm for smartphone users: Design and implementation. *IEEE Sensors Journal*, 15(4):2296–2305, 2015.

[75] Sabrina A Panëels, Dylan Varenne, Jeffrey R Blum, and Jeremy R Cooperstock. The walking straight mobile application: Helping the visually impaired avoid veering. In *Int. Conf. on Auditory Display (ICAD 2013)*. Georgia Institute of Technology, 2013.

[76] Bettina Pressl and Manfred Wieser. A computer-based navigation system tailored to the needs of blind people. *Computers Helping People with Special Needs*, pages 1280–1286, 2006.

[77] Lawrence Rabiner and Biing-Hwang Juang. An introduction to hidden markov models. *ASSP Magazine, IEEE*, 3(1):4–16, 1986.

[78] Timothy H Riehle, Shane M Anderson, Patrick A Lichter, NA Giudice, SI Sheikh, RJ Knuesel, DT Kollmann, and DS Hedin. Indoor magnetic navigation for the blind. In *Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE*, pages 1972–1975. IEEE, 2012.

[79] Timothy H Riehle, Shane M Anderson, Patrick A Lichter, William E Whalen, and Nicholas A Giudice. Indoor inertial waypoint navigation for the blind. In *Engineering in Medicine and Biology Society (EMBC), 2013 35th Annual International Conference of the IEEE*, pages 5187–5190. IEEE, 2013.

[80] UkJae Ryu, Kyungho Ahn, Entae Kim, Munhae Kim, Boyeon Kim, Sunghun Woo, and Yunseok Chang. Adaptive step detection algorithm for wireless smart step counter. In *Information Science and Applications (ICISA), 2013 International Conference on*, pages 1–4. IEEE, 2013.

[81] Angelo M Sabatini, Chiara Martelloni, Sergio Scapellato, and Filippo Cavallo. Assessment of walking features from foot inertial sensing. *IEEE Transactions on biomedical engineering*, 52(3):486–494, 2005.

[82] Ronald W Schafer. What is a savitzky-golay filter?[lecture notes]. *IEEE Signal processing magazine*, 28(4):111–117, 2011.

[83] Felix Scholkmann, Jens Boss, and Martin Wolf. An efficient algorithm for automatic peak detection in noisy periodic and quasi-periodic signals. *Algorithms*, 5(4):588–603, 2012.

[84] Jungryul Seo, Yutsai Chiang, Teemu H Laine, and Adil M Khan. Step counting on smartphones using advanced zero-crossing and linear regression. In *Proceedings of the 9th International Conference on Ubiquitous Information Management and Communication*, page 106. ACM, 2015.

[85] J Silva, C Silva, L Marcelino, R Ferreira, and A Pereira. Assistive mobile software for public transportation. *Proc. UBICOMM*, 2011.

[86] Jerry Tai Fook Lim, Goh Han Leong, and Tan Kok Kiong. Accessible bus system: a bluetooth application. *Assistive Technology for Visually Impaired and Blind People*, pages 363–384, 2008.

[87] YingLi Tian, Xiaodong Yang, Chucai Yi, and Aries Arditi. Toward a computer vision-based wayfinding aid for blind persons to access unfamiliar indoor environments. *Machine vision and applications*, 24(3):521–535, 2013.

[88] Carla Tinti, Mauro Adenzato, Marco Tamietto, and Cesare Cornoldi. Visual experience is not necessary for efficient survey spatial cognition: evidence from blindness. *The Quarterly Journal of Experimental Psychology*, 59(7):1306–1328, 2006.

[89] Michal Tomlein, Pavol Bielik, Peter Krátky, Stefan Mitrík, Michal Barla, and Mária Bieliková. Advanced pedometer for smartphone-based activity tracking. In *HEALTHINF*, pages 401–404, 2012.

[90] Kinh Tran, Tu Le, and Tien Dinh. A high-accuracy step counting algorithm for iphones using accelerometer. In *Signal Processing and Information Technology (ISSPIT), 2012 IEEE International Symposium on*, pages 000213–000217. IEEE, 2012.

[91] Pepijn Viaene, Ann Vanclooster, Kristien Ooms, and Philippe De Maeyer. 2014 ubiquitous positioning indoor navigation and location based service (upinlbs).

[92] Philip L Winters, Sean Barbeau, and Nevine Labib Georggi. Travel assistance device (tad) to help transit riders. *Final report for Transit IDEA Project*, 52, 2010.

[93] Oliver J. Woodman. An introduction to inertial navigation. Technical Report UCAM-CL-TR-696, University of Cambridge, Computer Laboratory, August 2007.

[94] Ling-Mei Wu, Jia-Shing Sheu, Wei-Cian Jheng, and Ying-Tung Hsiao. Pedometer development utilizing an accelerometer sensor. *World Academy of Science, Engineering and Technology, International Journal of Computer, Electrical, Automation, Control and Information Engineering*, 7(7):864–869, 2013.

[95] Che-Chang Yang and Yeh-Liang Hsu. A review of accelerometry-based wearable motion detectors for physical activity monitoring. *Sensors*, 10(8):7772–7788, 2010.

[96] Zheng Yang, Chenshu Wu, and Yunhao Liu. Locating in fingerprint space: wireless indoor localization with little human intervention. In *Proceedings of the 18th annual international conference on Mobile computing and networking*, pages 269–280. ACM, 2012.

[97] Zheng Yang, Chenshu Wu, Zimu Zhou, Xinglin Zhang, Xu Wang, and Yunhao Liu. Mobility increases localizability: A survey on wireless indoor localization using inertial sensors. *ACM Computing Surveys (CSUR)*, 47(3):54, 2015.

[98] Mitsuru Yoneyama, Yosuke Kurihara, Kajiro Watanabe, and Hiroshi Mitoma. Accelerometry-based gait analysis and its application to parkinson's disease assessment-part 1: Detection of stride event. *IEEE Transactions on neural systems and rehabilitation engineering*, 22(3):613–622, 2014.