GesturePod: Gesture-based Interaction Cane for People with Visual Impairments

Shishir G. Patil Don K. Dennis Chirag Pabbaraju Nadeem Shaheer
Harsha Simhadri Vivek Seshadri Manik Varma Prateek Jain

Microsoft Research India

ABSTRACT

People using white canes for navigation face challenges concurrently accessing other devices, e.g., smartphones. Building on recent research on abandonment of specialized devices, we explore a new touch free mode of interaction, wherein a person with visual impairment performs gestures on their existing white cane to trigger tasks on their smartphone. We present an easy-to-integrate GesturePod, that clips on to any white cane and enables the detection of gestures performed with the cane. GesturePod, thereby, helps manage a smartphone without touch, or removing the phone from a pocket or bag. In this paper, we present design decisions and challenges in building the pod. We propose a novel, efficient machine learning pipeline to train and deploy the model. Our in-lab study shows that GesturePod achieves >92% gesture-recognition accuracy and can significantly reduce the time taken for common smartphone tasks. Our in-wild study suggests that GesturePod is a promising interaction tool for smartphone, especially in constrained outdoor scenarios.

KEYWORDS

Gesture recognition; Machine learning; Microcontrollers; Accessibility; White cane

1 INTRODUCTION

Smartphones have become an integral part of our lives. While new technologies and applications on the smartphone have improved the lives of all people, smartphones have significant potential to positively impact the lives of people with visual impairments (VI), particularly in the Global South, where mainstream apps such as ride-hailing and maps are beginning to combat the accessibility barriers. Furthermore, recent smartphone apps such as Seeing.AI [37], Soundscape [36], and Eye-D [49] allow people with VI to be more aware of and better navigate their surroundings. These apps provide a glimpse of the exciting possibilities that smartphones can offer people with VI.

CHI'19, May 2019, Glasgow, UK
2019. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM...$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

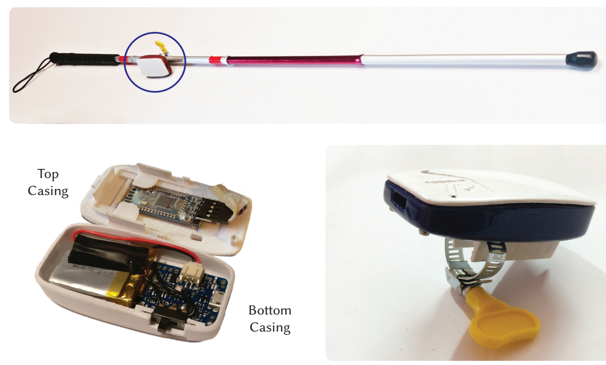
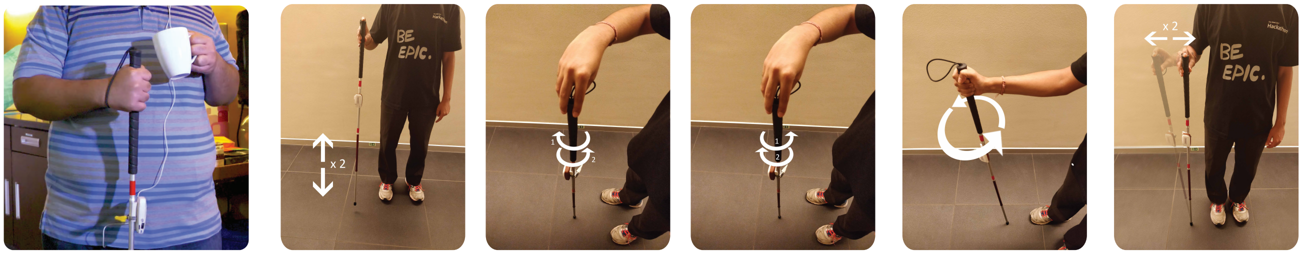


Figure 1: GesturePod. Top image shows the pod attached to a white cane. Bottom left image shows the interior of the pod.

Despite these benefits, smartphone *accessibility* remains a challenge. Studies (e.g., [26]) show that the user interfaces for mobile apps can be cumbersome to use for people with VI. Furthermore, for users with VI who use a cane for navigation, there are many situations where even accessing or locating the smartphone in a timely manner can be difficult. Figure 2a shows one such situation where both the hands of a person with VI are occupied with a cane and a coffee mug. In this paper, we focus on this problem of accessibility to a smartphone in constrained settings with particular emphasis on the Global South, where people living with disabilities often have lower incomes [42] and typically cannot afford expensive accessibility technologies.

One already available solution that improves access to the smartphone is voice commands. However, voice commands are not sufficient because 1) they may not work in all situations (e.g., noisy environments); 2) users may not always wish to use voice commands due to privacy reasons [55]; 3) a vast number of regional languages worldwide have no voice command support. For example, there is no support for voice commands for Tamil, the primary language for more than 70 million people around the world.

Since the white cane is the most common tool for blind people to navigate the real world [57], attempts have been made to augment the white cane with devices that allow a



(a) **(b) Double Tap** **(c) Right Twist** **(d) Left Twist** **(e) Twirl** **(f) Double Swipe**

Figure 2: (a) A person who is visually impaired in constrained environment, with both hands occupied. (b)-(f) Illustration of various gestures used in our study.

person with VI to interact with the environment. Such devices include buttons [47] and a touchpad [51] on the cane. On the other hand, there have also been work on using wearables, eg., rings [2] and dials [27] to access the phone. In this work, we explore a *novel*, *complementary* mode of converting the white cane into an interaction device—*performing gestures using the cane*. Since gestures are natural, easy to learn and perform, and do not require users to change their normal cane usage (for example, their grip on the cane), they provide a promising avenue for cane augmentation. However, when designing such a solution for users with VI, we must ensure that it does not interfere with their normal cane use. The solution should be adaptable to any cane and should not require modification of the cane. For user with VI in low-income settings (our target demographic), this translates to four main constraints: robust gesture recognition, low power consumption, low weight, and low cost. Section 3 describes these constraints in detail.

The main contribution of this work is designing a device, GesturePod, that can be clamped on to *any* white cane making it a gesture-based interaction device (Figure 1). To reduce cost and weight, GesturePod uses a low-cost, lightweight microcontroller, off-the-shelf sensors (accelerometer + gyroscope), a BLE module for communication, and a small battery. To reduce power consumption and robustly recognize a number of gestures, we carefully designed a machine learning (ML) pipeline that allows GesturePod to run the gesture recognition *fully* on the microcontroller. Overall, GesturePod costs less than 10 USD, weighs 49 g, continuously runs for 14+ hours on a single charge, and accurately recognizes five different gestures in the real world across a range of users and environments.

To enable deployment of our ML algorithm on the microcontroller, we designed novel features and exploit recently proposed ProtoNN algorithm [15]. Our ML pipeline consists of 1) a simple methodology to collect training data for gestures, 2) tools to automatically curate the collected data and train the ML model, and 3) an optimized code that runs the trained model on the microcontroller. This pipeline allows easy addition of new gestures or replacement of existing gestures. In fact, two high-school students added a sixth gesture to our GesturePod within a single day using our ML pipeline.

Based on exploratory interviews, we designed an Android app that maps gestures on the cane to common smartphone tasks. To understand the usefulness and adaptability of GesturePod based cane, we conducted two sets of user studies: in-lab and in-wild study. Our user studies indicate that 1) GesturePod is easy to learn and use, 2) it accurately detects gestures across multiple users and environments, and 3) it can significantly improve the time to perform common smartphone tasks, especially in constrained settings.

We make the following contributions in the paper.

- We design an easy-to-use GesturePod that can be clamped on to any white cane and recognize a range of gestures performed on the cane. We address several technical challenges to ensure that GesturePod is robust, power-efficient, lightweight, and inexpensive.
- Our in-lab user study shows that GesturePod accurately recognizes gestures across a range of users and settings, and significantly reduces the time to complete common smartphone tasks.
- We qualitatively evaluate the adaptability of GesturePod using an in-wild study. Feedback from users indicate that our approach of using gestures on the white cane is a promising method for interacting with smartphone.

2 RELATED WORK

White cane: The white cane is an important navigation device for people with VI [58] and naturally has been subject of several recent studies [47, 52]. In particular, [53] studied various navigational challenges for people with VI and outlined several solutions. [21] highlighted key issues with electronics-based canes such as battery life, reaction time, floor-level etc., and provided several design guidelines.

Most existing solutions augment the cane with obstacle detection capabilities by using ultrasound sensors [46], camera [19], LIDAR [43], RFID tags [9, 13, 29] etc. Our solution is complementary and can potentially be combined with such solutions to provide easier and safer access to devices. Existing studies that augment cane with IMU sensors [5, 12, 24], typically focus on understanding cane's usage pattern rather than using it as an accessibility device.

A notable exception is the work by Batterman et al. [6] which proposes a button-based solution for the cane that adds a "home" button and four arrow buttons to the cane. Although, such solutions are complementary to our gesture based solution, our informal discussions with potential users indicated that touch-free interaction mechanisms like gestures might be more preferable (see Section 3.1).

Gesture Recognition: Gesture recognition is an extensively-studied problem, especially in the context of touch-free interaction systems [4, 28, 54]. But many solutions in this domain focus on camera-related sensors [14, 17, 39, 41, 44], that require large computing resources for image processing.

Several existing applications have also used relatively cheaper IMU sensors for detecting gestures [3, 10, 16, 23, 25, 30, 33, 35]. For example, [32] used the accelerometer readings from a watch, transmitted them to an Android tablet and ran a Naïve Bayes classifier on the data for recognizing hand gestures. [31] employed a k-Nearest Neighbor (kNN) on electromyography and accelerometer signals received via Bluetooth on a Nokia phone to remotely control mobile devices. [18] designed a Magic Ring, which can detect static predefined finger gestures like Finger Up, Finger Down, etc. using hand-tuned rules.

While gesture recognition has been studied extensively, most of the existing solutions either: a) rely on hand-tuned rules to detect activities like running, sleeping [11] that do not require accurate and real-time recognition, or b) apply simple ML algorithms that can handle simple gestures in *restricted* settings [31], or c) use a powerful device to detect gestures via computationally-expensive ML algorithms [30].

Our setting involves simple-to-perform gestures that are nuanced enough that rule-based systems and simple ML models cannot robustly detect these gestures across a variety of users, grips, and different environments—e.g., floor type, handedness. While more powerful ML models [34, 56] can robustly recognize our gestures, they cannot be run on our resource-constrained microcontrollers—a requirement that results from our design constraints. To the best of our knowledge, our GesturePod is the first gesture detection system that can detect complex-to-recognize gestures accurately and robustly on a tiny microcontroller.

3 DESIGN PRINCIPLES AND CHALLENGES

As mentioned in the introduction, we explore gestures on a white cane as a mode of interacting with the smartphone. In this section, we describe the rationale behind our approach and the challenge in designing such a system.

3.1 Why Gestures on the Cane?

Why Gestures? Gestures offer many advantages compared to other modes of cane based interaction (e.g., buttons, touchpad). First, unlike gestures, mounting buttons or touchpad on a cane may require the user to change their grip or normal usage. For example, one of the user in our study mentioned that "If I want to use the button at least I should stand or something like that but if I use a gesture by walking only I can use the gestures easily." Second, gestures can be intuitive and require a short learning curve. In fact, in all our studies, users learned the gestures with just 10 minutes of training. Third, unlike buttons, we can seamlessly add new gestures without modifying the cane. As mentioned before, two high-school students were able to add an additional gesture after playing with the device for a day.

Why on the Cane? Prior works proposed wearable technologies (e.g., band [11], smart watch [1], ring [2]) that offer touch-based or gesture-based interaction. Typically, these devices are expensive and therefore do not cater to our target demographic. Furthermore, intuitively, touch-based interaction are difficult to use when both the user's hands are occupied with the cane and some other object, a common scenario for cane users. In fact, our GesturePod can be potentially used to design other wearables like gesture-band or gesture-ring. However, given that the user might already have a cane for navigation, additional devices might see significant drop in usage as observed by various studies on technology abandonment [22, 45].

3.2 Design Constraints and Implications

Our goal is to design a simple, easy-to-use gesture-recognition device that can be clamped on to any, existing, white cane without any modifications to the cane. There are four main design constraints that make the design of such a device technically challenging.

First, for the device to be practical, it must robustly detect a number of gestures. These gestures must not interfere with the normal use of the cane and should be recognizable under a range of environment and users. Second, as the device has to be battery-operated, it must consume small amount of power to ensure that it lasts at least an entire day without recharging. This limits us to using low-power microcontrollers and simple off-the-shelf sensors (accelerometer and gyroscope) on the device. Third, as the users will be carrying the cane in their hand, the device must not add significant weight to the white cane. This constraint eliminates the possibility of directly mounting a phone or a large battery on to the cane. Finally, our focus is on persons with VI in the Global South. Therefore, our device should be of low cost. This is one of the reasons why devices like smart-watches that cost over 100 USD would not suit this audience. In fact, while people with VI in the US typically use iPhones (with better accessibility features) [38], a recent study found that people with VI in India use cheaper Android devices [20].

3.3 Technical Challenge

A naive approach for gesture recognition in our system is to use the microcontroller on the device to *only* collect the data from the sensor and transmit the data to the smartphone. Smartphone will then apply gesture recognition algorithm to the sensor data. However, for accurate gesture recognition, the device must sample the sensor data at a reasonably high rate and continuously transmit the data to the smartphone. Our experiments show that such method results in high battery consumption, and require the device to be charged almost every hour. Hence, the only option is to perform the gesture recognition on the microcontroller itself.

This task is quite challenging as the device must recognize multiple gestures (listed in Section 4.2) in a variety of different conditions (e.g., flooring), for a range of users (e.g., left/right handed, varying forces) and variations in grip. Such a sophisticated setting eliminates simple rule-based solutions. In fact, even after a month of effort, we could not create a usable rule-based solution with low false positives *and* false negatives. On the other hand, typical ML solutions require large amount of RAM and compute; for example, to obtain similar accuracy as our system, traditional algorithms [40] that apply standard classifiers like Gradient Boosted Decision Tree [7] with conventional FFT features [8] would require more than 1MB of RAM which is prohibitively large for a microcontroller.

So, the key technical challenge that we address is:

> Can we design a low-cost, low-power device that accurately and robustly recognizes a number of natural gestures in real-time using just a low-end microcontroller and off-the-shelf sensors?

In the following section, we describe the design and implementation of our GesturePod that successfully addresses the above challenge.

4 GESTUREPOD: DESIGN AND IMPLEMENTATION

We have developed a plug-and-play GesturePod that can be clipped onto any white cane. The pod is used for both labeled data collection, necessary for training the ML model and for deploying the trained model to infer gestures on the cane. We now describe design and implementation of GesturePod.

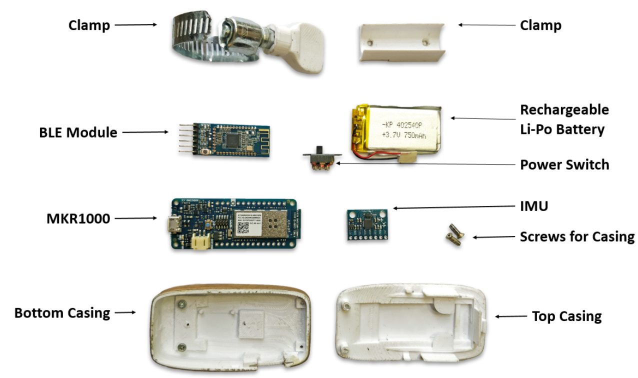


Figure 3: Components of GesturePod

4.1 Components of GesturePod

Our GesturePod is made up of five key components (Figure 3): 1) an off-the-shelf Inertial Measurement Unit (IMU), 2) an Arduino MKR1000 development board, 3) a Bluetooth Low-Energy (BLE) module, 4) a rechargeable battery, and 5) an on-off switch. The IMU contains an accelerometer, a gyroscope, and an internal 1 KB buffer that captures sensor data at a frequency of 200 Hz. The MKR1000 board consists of an ARM Cortex-M0+ microcontroller with 32 KB working memory and 256 KB of read-only flash. The microcontroller runs the entire gesture recognition pipeline (described in Section 4.4). GesturePod uses a BLE module to communicate recognized gestures to a connected smartphone. The rechargeable battery has 750 mAh capacity and powers all the above components. All these components are housed in a 76 mm×38 mm×25 mm casing with a clamp that allows us to mount GesturePod on most standard white canes.

4.2 Proposed Gestures on the Cane

We designed our gestures such that they can be mapped to common navigation buttons typically found in Android devices. Specifically, we use the *double swipe* gesture for the *select* action, *right twist* for *next*, *left twist* for *previous*, and *double tap* for *back/exit*. For added functionality, we included a fifth gesture: *twirl*. The following list describes how each of these gestures are performed (pictorially shown in Figure 2).

1. **Double tap** (D-T): Tap the cane on the floor twice
2. **Right twist** (R-T): Twist the cane to the right
3. **Left twist** (L-T): Twist the cane to the left
4. **Double swipe** (D-S): Tilt the cane to the right twice
5. **Twirl** (Tw): Make a circle with the cane's grip

We ensured these gestures are not triggered during natural cane use. For instance, we originally considered a single tap to map to the *select* action. However, we found single tap to generate too many false positives. While in this work, we focus on the above mentioned 5 gestures, adding a new gesture and replacing an existing gesture only requires collecting data points for the new gesture (Section 4.3) and training a new model using our ML pipeline (Section 4.4).

4.3 Data Collection for Model Training

To train our ML model, we need: 1) positive examples for each of the five gestures, and 2) negative examples where no gesture is performed. We used two different methods to collect, curate, and augment training data for each kind.

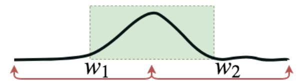
Positive Examples for Gestures. Seven sighted volunteers from our organization helped us collect training data for the five gestures. For each gesture, the volunteer performs the gesture using the cane, whilst an observer roughly marks the boundaries of the gesture in a program running on the computer. The program collects the sensor data using serial communication, labels it with the corresponding gesture, and stores it in a database. To ensure robust gesture recognition, our training data included variations in flooring, grip, orientation of the cane, and handedness. In total, we collected data for 102 double taps, 55 right twists, 54 left twists, 353 twirls, and 83 double swipes.

As the observer only approximately marked the boundaries of each gesture, we further curated the training data to ensure same duration for each example. We found that all the gestures could be performed within 1.5 seconds. So, we manually trimmed our training examples such that 1) each example had sensor data for exactly 2 seconds, and 2) the gesture is roughly centered within the 2-second window. We increase the training set size by ten-folds by adding additional examples where the region containing the gesture is shifted on either side by up to 25 ms at 5 ms steps.

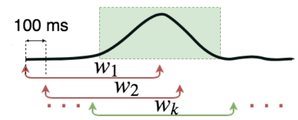
Negative Examples. For the GesturePod to be usable, we must ensure that it does not trigger any of our gestures during the normal use of the cane. While one can easily design a rule-based system to distinguish between our five gestures, engineering a rule-based system to avoid false positives in the numerous scenarios occurring in regular cane use is impractical. In fact, to significantly reduce the false positive rate in our ML-based model, we had to add three kinds of negative examples to our training dataset.

First, we added negative examples from regular cane use without any gestures. For this purpose, we connected GesturePod to an SD-card to record *all* sensor data from the IMU. We attached the pod to a cane and asked our volunteers to walk around with the cane without performing any of the five gestures. We collected 8 minutes of such data and clipped it to generate 2-second segments of negative examples.

Our initial model trained with just these negative examples resulted in high false positive rate. One of the major source of false positives was partial gestures, e.g., a single tap, that were recognized by the GesturePod as a full gesture. We



(a) Disjoint windows



(b) Overlapping windows

Figure 4: Striding Windows— w_1 and w_2 are disjoint windows in (a), while they are overlapping in (b)

found that these partial gestures occur frequently during natural cane use. For instance, the "three-point" technique [48] to climb stairs involves a single tap on each step. To avoid these false positives, we retrained our model using negative examples augmented with data from partial gestures, e.g., single tap, half a twist etc.

For the last round of data collection, we tagged any additional false positives generated by our model. For each instance, we extracted the corresponding sensor data and included it in our dataset as a negative example.

4.4 Machine Learning Pipeline

Our goal is to train a machine learning (ML) model that can take continuous stream of data from GesturePod's IMU sensors and detect whenever there is a gesture. However, similar to training examples, we cannot simply segment the sensor data stream into disjoint 2-second windows and run the model on each window. This is because a gesture performed by the user may cut across two consecutive windows as shown in Figure 4a. For this purpose, we generate overlapping 2-second windows by sliding windows by 100 ms–for example, in Figure 4b, window W_k encompasses the signature for the gesture.

Now, at the beginning of every sliding window (i.e. every 100 ms), the microcontroller must 1) fetch the IMU sensor data for the past 100 ms, 2) run ML prediction on the latest 2-second window, and 3) communicate any recognized gesture to the smartphone. Steps (1) and (3) consume roughly 20 ms. Therefore, ML prediction must complete within 80 ms.

ML prediction typically consists of two parts: 1) data featurization that converts raw sensor data into features that are suitable for the ML model, and 2) a classification algorithm that classifies the featurized data into one of the gestures (including no gesture).

For our classification algorithm, we use the multi-class formulation of the recently-proposed ProtoNN [15] algorithm which is specifically designed to generate models small enough to run on microcontrollers. However, when we employed ProtoNN with standard ML features (e.g., FFT features [8], clustering-based features [50]), it exceeded the time budget due to data featurization cost. Therefore, we had to design a set of features that 1) are easy enough to compute,

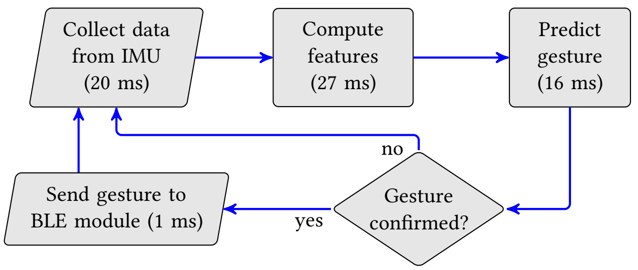


Figure 5: Prediction pipeline loop that runs on the single-threaded microcontroller (must finish in 100 ms).

2) consumes small amount of memory, and 3) are still robust enough to be able to discern practical gestures.

Our Proposed Features. As mentioned before, the featurization step converts raw sensor data into a set of features. The raw sensor data consists of six dimensions, three each from the accelerometer and the gyroscope. For 2-seconds, when sampled at 200 Hz, the raw data for each prediction instance consist of 400 values in each of the six dimensions.

We design two kinds of features. First, for each of the 6 dimensions, we group the 400 values into 20 equally-spaced bins in their range and count the number of values in each bin. Such equally-spaced bin counts are particularly efficient to compute in our strided-window setting. Second, bin count features discard phase in the gyroscope values (clockwise vs. anti-clockwise). To capture this phase information, we add four additional features: the index and length of the longest positive and negative sequence of values from each dimension of the gyroscope. However, typically, the only axis of rotation is along the vertical axis of the cane. Therefore, we only needed phase information along the this axis of the gyroscope. In total, we compute 124 features for each training sample.

Model Training. With these new set of features, we trained the ProtoNN model on our dataset using a commodity PC. We randomly split the collected training dataset (discussed earlier) into 80% training samples and 20% testing samples. We tuned the ProtoNN hyper-parameters simultaneously to achieve high accuracy and low model size. Our final model was just 6 KB in size and achieved an accuracy of 99.9% on the test data. In the real world, our model achieves 92% accuracy (discussed in Section 5).

Prediction Pipeline. Our gesture prediction pipeline runs a continuous cycle of (a) data collection from IMU, (b) feature computation, (c) ProtoNN inference algorithm on the model generated by the ProtoNN training algorithm and (d) BLE communication for relaying the gestures detected to the phone. With our new feature set and our 6 KB ProtoNN model, the microcontroller can complete data featurization and the ML classification in 27 ms and 16 ms, respectively.

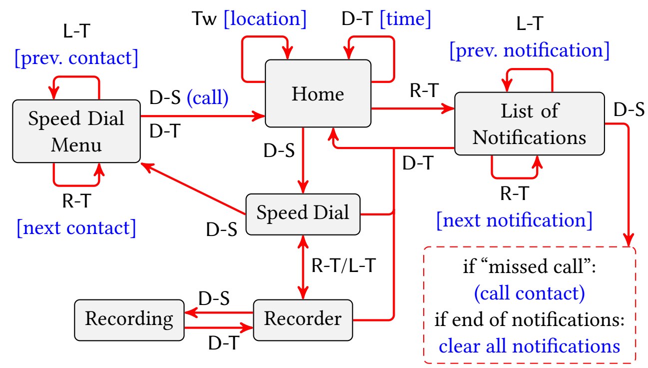


Figure 6: State machine for the app on smartphone. During state transitions, the app explicitly reads out the text corresponding to the item within square brackets.

To reduce false positives, we use a secondary filter wherein our algorithm keeps track of the latest n predictions. If the majority of the these predictions point to a particular gesture, then our algorithm *confirms* the presence of that gesture and communicates the gesture to the smartphone. Due to 100 ms stride in our windows, typically 4 to 6 consecutive windows fully contains a gesture. Therefore, we set $n = 6$. Including this step, microcontroller can execute the entire prediction pipeline well within the time budget of 100 ms (see Figure 5).

4.5 Android App

Our exploratory interviews (see supplementary material) with 15 volunteers with visual impairments—who did not take part in the user studies—indicated that the core smartphone tasks that they wanted to be more accessible are: reading out time and location, reading notifications, answering and declining calls, and callbacks to missed calls.

Based on this initial feedback, we designed an Android app with the aim of simplifying access to the above mentioned tasks via gestures. For instance, left twists and right twists are used to scroll forward and backward through the app menu or notifications list, and the user can double tap at any time to return to the home state, much like a home button on android phones. Gestures performed on the cane results in the corresponding transitions in the app's state machine (Figure 6). For example, a double tap at the 'home' state of our app triggers the phone to read out the current time. The Android app is designed to give the user voice feedback over the phone's speakers or earphones. As users tend to always have one of their earphones in their ear, they preferred voice feedback over earphones. Similarly a twirl in the home state reads out the current location. We should note that the users in our exploratory interviews also wanted deeper integration with messaging (e.g. WhatsApp), taxi (e.g. Uber) and other apps, but for the initial study, we avoided working with the third-party apps.

5 IN-LAB STUDY

We conducted an in-lab user study to *quantitatively* measure the robustness and effectiveness of GesturePod. Specifically, we ran two experiments that measure 1) the accuracy with which GesturePod could recognize the gestures performed by our users, and 2) the improvement in the time taken by our users to complete specific tasks on their smartphone using the cane mounted with our GesturePod (henceforth referred to as I-Cane). In addition to these measurements, we collected feedback on the usability of the cane on a Likert scale (see supplementary material).

For this study, we recruited 12 users with VI from three Non-Government Organizations (NGOs) and 6 sighted users from neighboring establishments not related to our institute. All users with VI had received mobility training to use the white cane. Among our 12 users with VI, we had diversity in terms of qualifications, gender, age, and handedness. Eight had owned a smartphone for at least three months. Based on consultation with the NGOs and our Institutional Review Board, we compensated each user 7 USD, which is roughly a day's minimum wage for a skilled worker.

5.1 Accuracy of GesturePod

All 18 users took part in this experiment. We first trained users on how to perform the five gestures (details below), and then measured the accuracy of GesturePod in recognizing the gestures performed by our users. For this experiment, we designed an Android app that speaks out a random gesture and waits for the user to perform the gesture with I-Cane for a period of 10 seconds. We used this app for both training and the subsequent accuracy measurement.

User Training. We first described the project to all our users. For users with VI, we held their hands and demonstrated how to perform each gesture once. Sighted users observed how we perform each gesture once. After this initial demonstration, we used our app to train the users where the app requests the user to perform each gesture 5 times. For each attempt, the app notifies the user if GesturePod recognized the gesture. The training lasted 10 minutes for each user.

Accuracy Measurement and Metrics. After the training phase, we use our app to measure the gesture recognition accuracy of GesturePod for each user. The app reads out 25 gestures (5 instances of the 5 gestures) in a *random* order for each user, and tracks if GesturePod correctly recognized the gesture. We measure accuracy for each gesture using two metrics: 1) *recall*, the fraction of times the performed gesture was detected correctly, and 2) *precision*, the fraction of times GesturePod's detected gesture was correct. A high recall is an indicator of low false negatives, and high precision is an indicator of low false positives.

Performed Gesture	**Recognized Gesture**						
	D-T	R-T	L-T	Tw	D-S	N-G	Recall
D-T	95	0	0	0	0	0	1
R-T	0	86	8	0	0	1	0.91
L-T	0	9	81	0	0	5	0.85
Tw	0	0	0	82	11	2	0.86
D-S	0	0	0	1	93	1	0.98
Precision	1	0.91	0.91	0.99	0.89	-	

Table 1: Confusion matrix for gesture detection performed across all visually impaired users. N-G corresponds to the case when no gesture was recognized.

Results and Discussion. Table 1 presents the results of our experiment for users with VI. The ij^{th} value in the table is the number of times a gesture in the i^{th} row is detected as a gesture in the j^{th} column. We draw three conclusions. First, GesturePod recognizes double tap (D-T) and double swipe (D-S) with high recall. Second, even though the model occasionally mispredicts a right twist (R-T) as a left twist (L-T) and vice versa, it still achieves over 86% recall for both these gestures. Third, as a twirl (Tw) performed incorrectly has a signature similar to double swipe (D-S), the model sometimes mispredicts a twirl as a double swipe.

Overall, across all gestures, GesturePod achieves a precision of 92%±3% (with 95% confidence). The results for sighted users are comparable— 94% ± 3.8% (with 95% confidence). We present results for sighted users in the supplementary material due to lack of space. Note that our ML model was not trained with data from these users, thus these results highlight the robustness and generalizability of our model.

5.2 Impact on Smartphone Access Time

In this experiment, we measured the impact of the I-Cane on time taken to complete certain activities on the smartphone. We did this experiment with the 8 users with VI who owned a smartphone for at least three months. We studied the following five activities (also described in Section 4.5).

(1) Answer a phone call from a test phone.
(2) Call back the last caller from a missed call notification.
(3) Start an audio recording and stop.
(4) Know the current geographic location.
(5) Check for notifications and read out the time.

Based on the feedback from our exploratory interviews, we identified two experimental settings: an *unconstrained setting* in which one of the user's hands is free and the other hand is holding the cane, and a *constrained setting* in which both of the user's hands are occupied–one hand holding the cane and the other hand holding an item (e.g., a bag). Apart from whether one of the users' hand was free, there were no differences between the two settings.

User Training. At the beginning of this experiment, we read out the state machine (Figure 6) that maps gestures on the I-Cane to activities on the phone to each user. We present the exact text used in the supplementary material. The users then practiced the activities using the I-Cane once.

Evaluation. We measure the time taken by each user to complete the five activities in four different scenarios: *constrained* or *unconstrained*, and smartphone-alone or I-Cane. To eliminate "competition effect", we did not a priori inform the participants that we will be timing them. Instead, we video record the experiment and replay the video to measure the time after the study. To eliminate "practice effect", across users, we randomize 1) the order of these scenarios, and 2) the order of the activities within each scenario. For the smartphone-alone scenarios, participants used their usual mode of interacting with the smartphone, such as touch UI, voice commands, or accessibility mode.

Results and Discussion. Some users were unable to complete some of the activities using the smartphone-alone. Specifically, three users were unable to complete the recording activity in both unconstrained and constrained settings; and one user was unable to identify the location and read out notifications in the constrained setting. In contrast, when using the I-Cane, all the users were able to complete all the activities in both the unconstrained and constrained settings.

Figure 7 plots the average speed-up in task completion times using I-Cane compared to the smartphone-alone for each of the five activities across all the users, along with the standard deviation. While computing speedup, we exclude data points where a user was unable to complete the activity on the smartphone-alone. We draw two conclusions from our results. First, in both unconstrained and constrained settings, users were able to complete tasks 2×–9× faster using I-Cane than using their smartphone-alone. Second, as expected, the speedups are slightly higher in the constrained setting, especially for the location task. For the location task, most users had to navigate multiple screens to identify the location using the smartphone-alone, whereas, with the I-Cane, they could complete the task with a single gesture.

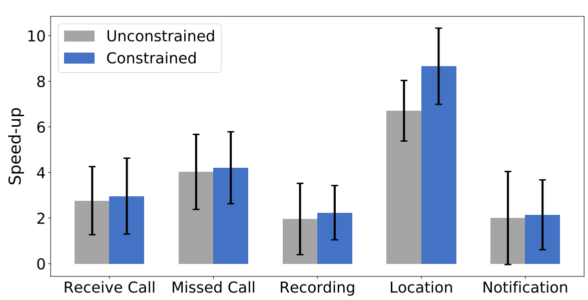


Figure 7: Speedup in task completion times using I-Cane over using smartphone alone.

While one user successfully used Google voice commands to complete some tasks, other users stated that they generally do not prefer voice commands for various reasons:

(1) "I reside in a hostel with other blind people. If all of us start talking to use the phone, there would be a mess"
(2) "It does not recognize vernacular accents"
(3) "On the streets, with the honking and noise, it does not recognize our voices"

6 IN-WILD STUDY

To understand the usefulness and adaptability of GesturePod in the real-world, we recruited three users with visual impairment from two NGOs. All three users had received mobility training and use a smartphone. In fact, one of the users had participated in our in-lab user study. Similar to that study, we briefly described our project to each user and trained them about our gestures and the app. Additionally, we trained each user on the operation of GesturePod, switching it on/off, charging it, and connecting it to their phone. We attached our GesturePod to each user's cane and connected the pod to their phone. Following this training, we informed the user that, for the remainder of the study, they could interact with their phone either using their normal method or through gestures on the cane, whichever they deemed fit. The study lasted 15 days for each user. We did not contact the user until the end of the study.

Evaluation Metrics. During the study, our app logs all the gestures performed by the user. However, to prevent "competition effect," we did not mention this logging to our users. At the end of the study, we collected feedback from each user in three ways: 1) System Usability Scale (SUS) to measure usability of the system, 2) Likert scale responses on the design of the system, and 3) semi-structured interviews. We recorded, transcribed, and analyzed the interviews to identify emergent themes. To prevent response biases, an independent individual not part of our project collected the SUS and Likert scale responses.

6.1 Quantitative Results

Figure 8 plots the number of gestures we detected for each of the three users (P1—P3). As there was a high variance in the number of gestures across users (possible due to differing phone usage), the figure plots the number of gestures in log scale. In addition, during our interviews, we learned that the users did not use their cane on days they were with their family members due to social stigma (see next section). Therefore, we plot the number of gestures for each user only for days on which we detected at least one gesture. Our results show that each user used the cane on at least 11 days

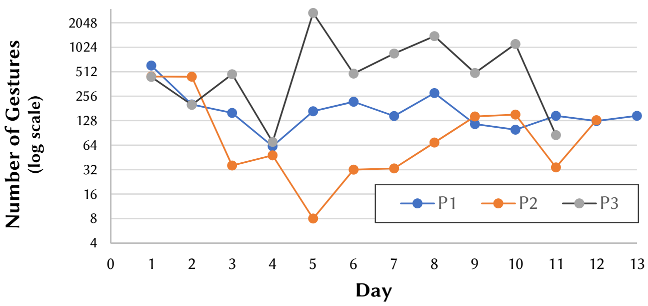


Figure 8: Number of gestures performed by users on days we detected a gesture

Question	P1	P2	P3
Was performing the gestures comfortable?	3	5	5
Were the gestures intuitive to perform?	5	5	5
Was the cane able to detect your gestures accurately?	3	4	4
How easy did you find it to remember the gestures?	5	5	5
How much effort was required to get used to the way the gestures are to be performed?	5	4	5
How would you rate ease to use and operate the app?	4	5	4
How would you rate the physical design of the pod?	5	5	5
How would you rate the overall product experience?	3	4	5

Table 2: User ratings from in-wild study on different aspects of GesturePod. Rating: 1 - lowest, 5 - highest.

out of 15 days of study, and each user's median number of gestures performed was more than 58. We noticed that P3 performed a large number of gestures on day 5. During our interviews, he told us that he had demonstrated the cane to his students on that day.

As our app cannot distinguish between a real gesture and a false positive, we asked our users if they experienced any false positive during their cane use. Two of the users mentioned occurrences of few false positives across all 15 days: P1 (4 false double taps) and P2 (20 false double swipes). These numbers are negligible in comparison to the total number of gestures performed by our users, and the users did not report significant inconvenience due to false positives.

Users also gave positive feedback on both the System Usability Scale and the Likert scale. We provide the exact questions for SUS and the anchors for the Likert scale in the supplementary material. In summary, the scores for the SUS scale were encouraging: P1–90, P2–75, and P3–95. Table 2 shows the responses from each user for the Likert scale. Users gave a high rating for almost all aspects of the solution.

These results indicate that our approach of using gestures on the cane to interact with the smartphone is robust, promising, and potentially useful in the real world.

6.2 Qualitative Feedback

We conducted semi-structured interviews with our users at the end of the study. In general, the users found I-Cane to be most useful in constrained scenarios. In contrast, it was least useful when the users were with their family members, due to limited requirement of independent navigation as well as due to social stigma. All the users mentioned that they did not feel any strain while performing the gestures, and the pod (and it's weight) did not effect their normal cane usage.

Constrained Scenarios. All users reported I-Cane to be helpful in constrained scenarios, i.e., when both their hands were occupied. One such scenario is during the use of public transit such as buses or trains, in which seating is often unavailable. A common challenge in such situations is having to balance oneself by holding on to a railing. The exact moment of a cellphone vibrating can be jarring and distracting, and trying to remove it from one's pocket or bag and operate it can cause tripping and injury.

> *When I was traveling by metro, holding the cane in one hand and the railing (support) in the other hand. I am getting calls, before it was not possible (to answer them). Now I can talk using the cane.* —P2

Another common situation that people report is needing to know the current time or location, especially when they are travelling. Here too, one needs both hands, if touching a tactile watch, or managing multiple swipes or strokes on a touchscreen to get the time or location.

> *When I am traveling in bus, we are standing and it is easy to know the time and location using the cane.* —P3

Users also found it reasonably easy to perform gestures while moving, especially, double tap and twist gestures. This can potentially help them attend to time-critical tasks.

> *Earlier I had to take my phone from pocket and stand somewhere to pick up a call, but now I can answer it while walking.* —P3

One disadvantage with not having a visual interface to a smartphone is the time taken to react to a call due to the inability to glance at a ringing device. Moreover, when the phones need to be kept in a secure location, such as locked in a purse or in a secure pocket area, the reaction time in removing the device can be long enough to delay a timely interaction. All users noted the benefit of I-Cane in being able to complete tasks without needing to physically handle a phone and deal with the touchscreen.

> *Many times I would miss calls due to delay in accepting them. Now that does not exist.* —P2

Users found the cane to be useful even when one of their hands were free.

> *In footpaths, even without any luggage in our hand, we have to stop and then remove the phone from our pockets, and then note the time. This makes people behind us to*

stop. Also, this is very long. With cane, we can know the time immediately. —P1

Situational Awareness. Locational awareness is a critical part of independent life. Our app logs indicate that all users frequently used the cane to query for location. In fact, all users explicitly mentioned liking the ability to query their current location with a single gesture.

In unknown locations, sometimes people won't tell us correctly where we are, sometimes it's hard to find people. With the cane, we can now know the location on our own. Before taking the phone out would take time. —P1

Locational awareness may require reassurance or repeated checking within very short intervals. This is common for sighted people using mapping technologies (e.g., Google maps) in which one may glance at a screen several times as they move. Such repeated queries for location can be particularly tedious for people with VI just using their smartphone UI. With I-Cane, the users were able to locate themselves quickly, and repeatedly.

If I want to go from office to Parngipalla, sometime I will get confuse in the crosses, if I check the location it was saying the cross 27th main 18th cross. So I was able to easily find out I am in this particular cross and I can navigate well. —P3

Cultural Conditioning. We found that the notion of people with VI navigating on their own was still alien. This was true for our users whose family members would often accompany them to various locations rather than allow them to travel on their own. While this was internalized as care on part of family members to ensure that they were not hurt, the net outcome was limited experience with outdoor spaces.

Users also referred to the use of the cane as stigmatizing because they identified one as being disabled. Consequently, there were entire days during which some of our users did not use the cane. As two users put it,

I am a totally blind person and I cannot live without the cane ... yet when I am at home my parents do not let me use the cane. —P3

Stigma was related not just to the individual, but also to those around them by extension, since disability was sometimes seen as something embarrassing and therefore to be hidden from public view.

My sister and mother do not allow me to use the cane when they are around me, as I was getting engaged. —P2

Such issues of stigma are relevant to design decisions, since on one hand, there is a need for greater social awareness of disability as diversity, but there is also a need to consider the possibility that people with VI may find themselves forced to use devices or aids that do not reveal their disability to those around them.

7 DISCUSSION

In this section, we discuss insights we gained during our study. We hope that these insights will guide future research that aims to design solution for a similar context.

Safety of Gestures. Among the five gestures, we felt that the double-tap gesture may create noise that may either disturb people in the vicinity or draw attention to the user. However, our users mentioned that this was not a concern. One user mentioned that a "bigger" twirl could potentially hit someone or something in the immediate vicinity of the user. Given that these two gestures were the most popular among our users, it seems like users naturally adapted to performing these gestures in safe and polite manner.

Better Integration with Android. One of the user wished that he could "operate all of the mobile" using gestures. This is possibly by combining our gestures with the native android accessibility app, i.e., TalkBack. But we believe that easier and faster access to certain frequent and time-critical tasks on smartphone might be a better application of GesturePod; we will study both these approaches in future work.

Additional Functionality. Android's Google assistant requires an always-on microphone, which can drain the battery rapidly on low-end phones. In fact, one user mentioned that he would like one gesture to be directly wake up the Google assistant. Similarly, another user wanted a mechanism to know the battery level of the GesturePod, so that they can charge the pod at an appropriate time. These functionality can be added to our app's state machine. Similarly, some users requested gestures to be mapped to an action of their choice (e.g., call a specific person). These requests suggest that our Android app's state machine should be customizable by each user.

8 CONCLUSION

We introduced GesturePod, an easy-to-use gesture recognition device that can be clamped on any white cane so that gestures of the cane can be used to access smartphone. GesturePod is a touch free, low-cost (less than 10 USD), has short learning curve, and does not require carrying around additional devices other than the cane. GesturePod's real-time and power-efficient gesture recognition is enabled by a carefully engineered ML pipeline that runs on a tiny microcontroller housed inside the pod. Our in-lab and in-wild user studies indicate that 1) GesturePod is robust in recognizing gestures across a wide range of users and environments, and 2) GesturePod significantly speeds up access to smartphone for specific tasks. Both our results and qualitative feedback from the users suggest that, for persons with VI, performing gestures on the cane is a promising mode of interaction with their smartphone, especially in workplace or outdoors. office or are on the move.

REFERENCES

[1] Apple. 2015. Apple Watch. https://www.apple.com/com/watch/.

[2] Daniel Ashbrook, Patrick Baudisch, and Sean White. 2011. Nenya: Subtle and Eyes-free Mobile Input with a Magnetically-tracked Finger Ring. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 2043–2046. https://doi.org/10.1145/1978942.1979238

[3] Daniel Ashbrook and Thad Starner. 2010. MAGIC: A Motion Gesture Design Tool. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*. ACM, New York, NY, USA, 2159–2168. https://doi.org/10.1145/1753326.1753653

[4] Daniel Ashbrook, Carlos Tejada, Dhwanit Mehta, Anthony Jiminez, Goudam Muralitharam, Sangeeta Gajendra, and Ross Tallents. 2016. Bitey: An Exploration of Tooth Click Gestures for Hands-free User Interface Control. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '16)*. ACM, New York, NY, USA, 158–169. https://doi.org/10.1145/2935334.2935389

[5] Lawrence K. Au, Winston H. Wu, Maxim A. Batalin, Thanos Stathopoulos, and William J. Kaiser. 2008. Demonstration of Active Guidance with SmartCane. In *Proceedings of the 7th International Conference on Information Processing in Sensor Networks (IPSN '08)*. IEEE Computer Society, Washington, DC, USA, 537–538. https://doi.org/10.1109/IPSN.2008.52

[6] Jared M. Batterman, Vincent F. Martin, Derek Yeung, and Bruce N. Walker. 2018. Connected cane: Tactile button input for controlling gestures of iOS voiceover embedded in a white cane. *Assistive Technology* 30, 2 (2018), 91–99. https://doi.org/10.1080/10400435.2016.1265024 arXiv:https://doi.org/10.1080/10400435.2016.1265024 PMID: 28140766.

[7] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. ACM, 785–794.

[8] Gabe Cohn, Daniel Morris, Shwetak Patel, and Desney Tan. 2012. Humantenna: using the body as an antenna for real-time whole-body interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1901–1910.

[9] J. Faria, S. Lopes, H. Fernandes, P. Martins, and J. Barroso. 2010. Electronic white cane for blind people navigation assistance. In *2010 World Automation Congress*. 1–7.

[10] Davide Figo, Pedro C. Diniz, Diogo R. Ferreira, and João M. Cardoso. 2010. Preprocessing Techniques for Context Recognition from Accelerometer Data. *Personal Ubiquitous Comput.* 14, 7 (Oct. 2010), 645–662. https://doi.org/10.1007/s00779-010-0293-9

[11] Fitbit. 2007. Fitbit. https://www.fitbit.com/com/home.

[12] German H. Flores and Roberto Manduchi. 2016. 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 (ASSETS '16)*. ACM, New York, NY, USA, 141–150. https://doi.org/10.1145/2982142.2982179

[13] A. J. Fukasawa and K. Magatani. 2012. A navigation system for the visually impaired an intelligent white cane. In *2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. 4760–4763. https://doi.org/10.1109/EMBC.2012.6347031

[14] Maribeth Gandy, Thad Starner, Jake Auxier, and Daniel Ashbrook. 2000. The Gesture Pendant: A Self-illuminating, Wearable, Infrared Computer Vision System for Home Automation Control and Medical Monitoring. In *Proceedings of the 4th IEEE International Symposium on Wearable Computers (ISWC '00)*. IEEE Computer Society, Washington, DC, USA, 87–. http://dl.acm.org/citation.cfm?id=851037.856538

[15] Chirag Gupta, Arun Sai Suggala, Ankit Goyal, Harsha Vardhan Simhadri, Bhargavi Paranjape, Ashish Kumar, Saurabh Goyal, Raghavendra Udupa, Manik Varma, and Prateek Jain. 2017. ProtoNN: Compressed and Accurate kNN for Resource-scarce Devices. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Doina Precup and Yee Whye Teh (Eds.), Vol. 70. PMLR, International Convention Centre, Sydney, Australia, 1331–1340. http://proceedings.mlr.press/v70/gupta17a.html

[16] Björn Hartmann, Leith Abdulla, Manas Mittal, and Scott R. Klemmer. 2007. Authoring Sensor-based Interactions by Demonstration with Direct Manipulation and Pattern Recognition. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '07)*. ACM, New York, NY, USA, 145–154. https://doi.org/10.1145/1240624.1240646

[17] Yongsik Jin, Jonghong Kim, Bumhwi Kim, Rammohan Mallipeddi, and Minho Lee. 2015. Smart Cane: Face Recognition System for Blind. In *Proceedings of the 3rd International Conference on Human-Agent Interaction (HAI '15)*. ACM, New York, NY, USA, 145–148. https://doi.org/10.1145/2814940.2814952

[18] Lei Jing, Yinghui Zhou, Zixue Cheng, and Tongjun Huang. 2012. Magic Ring: A Finger-Worn Device for Multiple Appliances Control Using Static Finger Gestures. *Sensors* 12, 5 (2012), 5775–5790. https://doi.org/10.3390/s120505775

[19] Jin Sun Ju, Eunjeong Ko, and Eun Yi Kim. 2009. EYECane: Navigating with Camera Embedded White Cane for Visually Impaired Person. In *Proceedings of the 11th International ACM SIGACCESS Conference on Computers and Accessibility (Assets '09)*. ACM, New York, NY, USA, 237–238. https://doi.org/10.1145/1639642.1639693

[20] Vaishnav Kameswaran, Jatin Gupta, Joyojeet Pal, Sile O'Modhrain, Tiffany C. Veinot, Robin N. Brewer, Aakanksha Parameshwar, Vidhya, Y., and Jacki O'Neill. 2018. 'We can go anywhere': Understanding independence through a case study of ride-hailing use by people with visual impairments in metropolitan India. In *To appear in Proceedings of the 21st ACM Conference on Computer-Supported Cooperative Work and Social Computing (CSCW '18)*. ACM, New York, NY, USA.

[21] Sung Yeon Kim and Kwangsu Cho. 2007. Usability and design guidelines of smart canes for users with visual impairments. *International Journal of Design* 7, 1 (2007), 99–110. http://www.ijdesign.org/index.php/IJDesign/article/view/1209/559

[22] Marion Koelle, Susanne Boll, Thomas Olsson, Julie Williamson, Halley Profita, Shaun Kane, and Robb Mitchell. 2018. (Un)Acceptable!?!: Re-thinking the Social Acceptability of Emerging Technologies. In *Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems (CHI EA '18)*. ACM, New York, NY, USA, Article W03, 8 pages. https://doi.org/10.1145/3170427.3170620

[23] Louis Kratz, Daniel Morris, and T. Scott Saponas. 2012. Making Gestural Input from Arm-worn Inertial Sensors More Practical. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 1747–1750. https://doi.org/10.1145/2207676.2208304

[24] Sven Kratz and Maribeth Back. 2015. Towards Accurate Automatic Segmentation of IMU-Tracked Motion Gestures. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '15)*. ACM, New York, NY, USA, 1337–1342. https://doi.org/10.1145/2702613.2732922

[25] Sven Kratz, Michael Rohs, and Georg Essl. 2013. Combining Acceleration and Gyroscope Data for Motion Gesture Recognition Using Classifiers with Dimensionality Constraints. In *Proceedings of the 2013 International Conference on Intelligent User Interfaces (IUI '13)*. ACM, New York, NY, USA, 173–178. https://doi.org/10.1145/2449396.2449419

[26] Ravi Kuber, Amanda Hastings, Matthew Tretter, and Dónal Fitzpatrick. 2012. Determining the accessibility of mobile screen readers for blind users. (2012).

[27] Fingertips Lab. 2017. O6. https://www.kickstarter.com/projects/55699542/o6-free-your-eyes.

[28] Walter S. Lasecki, Young Chol Song, Henry Kautz, and Jeffrey P. Bigham. 2013. Real-time Crowd Labeling for Deployable Activity Recognition. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work (CSCW '13)*. ACM, New York, NY, USA, 1203–1212. https://doi.org/10.1145/2441776.2441912

[29] Je Seok Lee, Heeryung Choi, and Joonhwan Lee. 2015. TalkingCane: Designing Interactive White Cane for Visually Impaired People's Bus Usage. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct (MobileHCI '15)*. ACM, New York, NY, USA, 668–673. https://doi.org/10.1145/2786567.2793686

[30] J. Liu, Z. Wang, L. Zhong, J. Wickramasuriya, and V. Vasudevan. 2009. uWave: Accelerometer-based personalized gesture recognition and its applications. In *2009 IEEE International Conference on Pervasive Computing and Communications*. 1–9. https://doi.org/10.1109/PERCOM.2009.4912759

[31] Zhiyuan Lu, Xiang Chen, Zhangyan Zhao, and Kongqiao Wang. 2011. A Prototype of Gesture-based Interface. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services (MobileHCI '11)*. ACM, New York, NY, USA, 33–36. https://doi.org/10.1145/2037373.2037380

[32] David Mace, Wei Gao, and Ayse Coskun. 2013. Accelerometer-based Hand Gesture Recognition Using Feature Weighted Naive Bayesian Classifiers and Dynamic Time Warping. In *Proceedings of the Companion Publication of the 2013 International Conference on Intelligent User Interfaces Companion (IUI '13 Companion)*. ACM, New York, NY, USA, 83–84. https://doi.org/10.1145/2451176.2451211

[33] Joseph Malloch, Carla F. Griggio, Joanna McGrenere, and Wendy E. Mackay. 2017. Fieldward and Pathward: Dynamic Guides for Defining Your Own Gestures. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 4266–4277. https://doi.org/10.1145/3025453.3025764

[34] Jani Mäntyjärvi, Juha Kela, Panu Korpipää, and Sanna Kallio. 2004. Enabling Fast and Effortless Customisation in Accelerometer Based Gesture Interaction. In *Proceedings of the 3rd International Conference on Mobile and Ubiquitous Multimedia (MUM '04)*. ACM, New York, NY, USA, 25–31. https://doi.org/10.1145/1052380.1052385

[35] David A. Mellis, Ben Zhang, Audrey Leung, and Björn Hartmann. 2017. Machine Learning for Makers: Interactive Sensor Data Classification Based on Augmented Code Examples. In *Proceedings of the 2017 Conference on Designing Interactive Systems (DIS '17)*. ACM, New York, NY, USA, 1213–1225. https://doi.org/10.1145/3064663.3064735

[36] Microsoft. 2017. Microsoft Soundscape. https://www.microsoft.com/en-us/research/product/soundscape/.

[37] Microsoft. 2017. Seeing AI. https://www.microsoft.com/en-us/seeing-ai/.

[38] John Morris and James Mueller. 2014. Blind and deaf consumer preferences for android and iOS smartphones. In *Inclusive designing*. Springer, 69–79.

[39] Annika Muehlbradt, Varsha Koushik, and Shaun K. Kane. 2017. Goby: A Wearable Swimming Aid for Blind Athletes. In *Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '17)*. ACM, New York, NY, USA, 377–378. https://doi.org/10.1145/3132525.3134822

[40] Fionn Murtagh and Pierre Legendre. 2011. Ward's hierarchical clustering method: clustering criterion and agglomerative algorithm. *arXiv preprint arXiv:1111.6285* (2011).

[41] Uran Oh, Lee Stearns, Alisha Pradhan, Jon E. Froehlich, and Leah Findlater. 2017. Investigating Microinteractions for People with Visual Impairments and the Potential Role of On-Body Interaction. In *Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '17)*. ACM, New York, NY, USA, 22–31. https://doi.org/10.1145/3132525.3132536

[42] Philip O'Keefe. 2007. People with disabilities in India: from commitments to outcomes (English). *Washington, DC: World Bank* 1, 41585 (2007), 1–185. http://documents.worldbank.org/curated/en/358151468268839622/People-with-disabilities-in-India-from-commitments-to-outcomes

[43] Tomàs Pallejà, Marcel Tresanchez, Mercè Teixidá, and Jordi Palacin. 2010. Bioinspired Electronic White Cane Implementation Based on a LIDAR, a Tri-Axial Accelerometer and a Tactile Belt. *Sensors* 10, 12 (2010), 11322–11339. https://doi.org/10.3390/s101211322

[44] Sumita Sharma, Saurabh Srivastava, Krishnaveni Achary, Blessin Varkey, Tomi Heimonen, Jaakko Hakulinen, Markku Turunen, and Nitendra Rajput. 2016. Gesture-based Interaction for Individuals with Developmental Disabilities in India. In *Proceedings of the 18th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '16)*. ACM, New York, NY, USA, 61–70. https://doi.org/10.1145/2982142.2982166

[45] Kristen Shinohara and Jacob O. Wobbrock. 2011. In the Shadow of Misperception: Assistive Technology Use and Social Interactions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 705–714. https://doi.org/10.1145/1978942.1979044

[46] Vaibhav Singh, Rohan Paul, Dheeraj Mehra, Anurag Gupta, Vasu Dev Sharma, Saumya Jain, Chinmay Agarwal, Ankush Garg, Sandeep Singh Gujral, M. Balakrishnan, Kolin Paul, P.V.M. Rao, and Dipendra Manocha. 2010. Smart cane for the visually impaired: Design and controlled field testing of an affordable obstacle detection system. In *12th International Conference on Mobility and Transport for Elderly and Disabled Persons (TRANSED '10)*.

[47] Dring Alert System. 2017. The Connected Cane. http://dring.io/en/the-connected-cane/.

[48] APH Tech. 2017. Long Cane Techniques. https://tech.aph.org/sbs/04_sbs_lc_study.html.

[49] GingerMind Technologies. 2017. Eye-d. https://www.eye-d.in/.

[50] T Warren Liao. 2005. Clustering of time series data-a survey. *Pattern Recognition* 38, 11 (2005), 1857–1874.

[51] WeWALK. 2018. WeWALK SMART CANE. https://get.wewalk.io/.

[52] WhiteCaneDay 2017. White Cane Day. http://www.whitecaneday.org/canes/.

[53] Michele A. Williams, Caroline Galbraith, Shaun K. Kane, and Amy Hurst. 2014. "Just Let the Cane Hit It": How the Blind and Sighted See Navigation Differently. In *Proceedings of the 16th International ACM SIGACCESS Conference on Computers & Accessibility (ASSETS '14)*. ACM, New York, NY, USA, 217–224. https://doi.org/10.1145/2661334.2661380

[54] Koji Yatani and Khai N. Truong. 2012. BodyScope: A Wearable Acoustic Sensor for Activity Recognition. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing (UbiComp '12)*. ACM, New York, NY, USA, 341–350. https://doi.org/10.1145/2370216.2370269

[55] Hanlu Ye, Meethu Malu, Uran Oh, and Leah Findlater. 2014. Current and Future Mobile and Wearable Device Use by People with Visual Impairments. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 3123–3132. https://doi.org/10.1145/2556288.2557085

[56] Ying Yin and Randall Davis. 2013. Gesture Spotting and Recognition Using Salience Detection and Concatenated Hidden Markov Models. In *Proceedings of the 15th ACM on International Conference on Multimodal Interaction (ICMI '13)*. ACM, New York, NY, USA, 489–494. https://doi.org/10.1145/2522848.2532588

[57] Yuhang Zhao, Cynthia L. Bennett, Hrvoje Benko, Edward Cutrell, Christian Holz, Meredith Ringel Morris, and Mike Sinclair. 2018. Demonstration of Enabling People with Visual Impairments to Navigate Virtual Reality with a Haptic and Auditory Cane Simulation. In *Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems (CHI EA '18)*. ACM, New York, NY, USA, Article D409, 4 pages. https://doi.org/10.1145/3170427.3186485

[58] Yuhang Zhao, Cynthia L. Bennett, Hrvoje Benko, Edward Cutrell, Christian Holz, Meredith Ringel Morris, and Mike Sinclair. 2018. Enabling People with Visual Impairments to Navigate Virtual Reality with a Haptic and Auditory Cane Simulation. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 116, 14 pages. https://doi.org/10.1145/3173574.3173690