Title

Ariadne’s Thread: Robust Turn Detection for Path Back-Tracing Using the iPhone

Authors

Flores, German
Manduchi, Roberto
Zenteno, Enrique D

Publication Date

2014-10-04

Peer reviewed

Ariadne's Thread: Robust Turn Detection for Path Back-Tracing Using the iPhone

German H. Flores
and Roberto Manduchi
University of California
Santa Cruz

Enrique D. Zenteno
CONABIO
Mexico City
Mexico

***Abstract*—Most systems for pedestrian localization and self-tracking aim to measure the precise position of the walker and match it against a map of the environment. In some cases, a simpler topological description of the path taken may suffice. This is the case for the system described in this paper, which is designed to help a blind person re-trace the route taken inside a building and to walk safely back to the starting point. We present two turn detection algorithms based on hidden Markov models (HMM), which process inertial data collected by an iPhone kept in the walker's front pocket, without the need for a map of the environment. Quantitative results show the robustness of the proposed turn detectors even in the case of drift in the measurements and noticeable body sway during gait.**

I. 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 to 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.

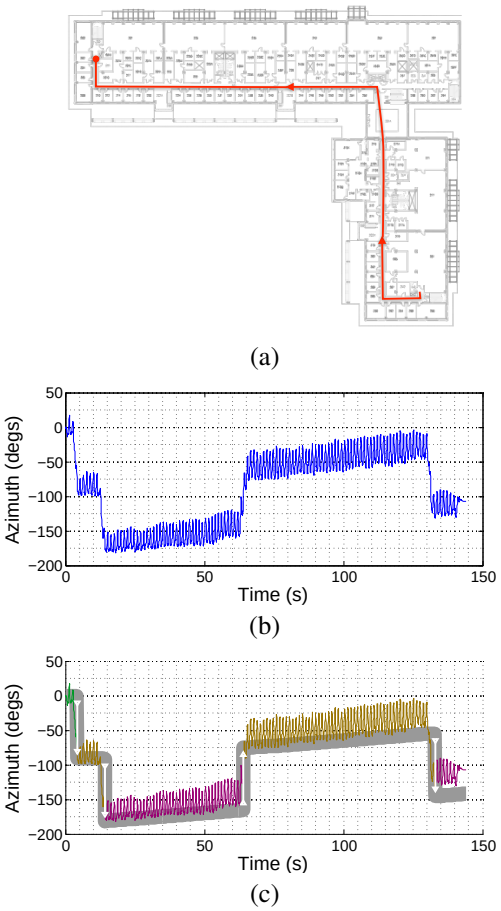

Fig. 1. (a): A floor plan with a path traversed by a walker. (b): The azimuth time series, collected by an iPhone 4 kept in the walker's front pocket. Drift and oscillation due to body sway are apparent. (c) Turns detected by our system. In this and subsequent figures, the white arrows represent turns (the direction and length indicating the turn angle), while the thick grey 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.

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 WiFi 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. 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 kept in one's pocket), and drift (error accumulation through time) is unavoidable over long hauls. Both these effects are visible in Fig. 1.

In this contribution 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 contribution is organized as follows. After the related work presented in the next section, we describe our two algorithms in Sec. III. The algorithms are then evaluated in Sec. IV, 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. V has the conclusions and indications for future research.

II. 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 [1], [2], [3]. 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) [4] and Wi-Fi positioning (e.g. the AXS system by EO Guidage[1]. A localization system for blind travelers developed by indoo.rs[2] and based on iBeacons (low power Bluetooth) is being tested at San Francisco Airport at the time of this writing. Promising results have also been obtained using a magnetic localization sensor [5].

Dead reckoning based on inertial data allows for positioning without reference to a WiFi 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 [6], [7]. 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 [8], or reset the system when a turn has been detected [9]. The use of inertial sensors to help a blind person walk straight (without veering) has also been proposed [10].

Finally, we should mention that a different HMM-based algorithm for localization was described in [11]. However, the state representation of this previous system includes spatial locations, whereas our model only represents orientations or turns.

III. 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 iOS' Core Motion Framework. `CMAttitude` provides the device's attitude with respect to a fixed reference system, using a 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 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,

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

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. 1, along with the path overimposed 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 oversmoothed 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.

A. HMM Turn Detection – Algorithm 1

In this case, $\mathcal{S}$ is formed by a fixed set of azimuth values: $\mathcal{S} = \{0^\circ \pm 90^\circ, 180^\circ\}$. If diagonal corridors are expected, then the set is augmented with the angles $\{\pm 45^\circ, \pm 135^\circ\}$. 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:

$$o^t = s^t + n^t \tag{1}$$

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.IV-C. 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. (1) should be changed into

$$o^t = s^t + n^t + \theta^0 \tag{2}$$

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. $[-45^\circ, 45^\circ]$). 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 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. (1) can be modified as follows to account for drift:

$$o^t = s^t + n^t + D^t \tag{3}$$

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.

B. 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), $\pm 90^\circ$, and 180°. If diagonal corridors are expected, it also contains $\pm 45^\circ$ and $\pm 135^\circ$. 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:

$$\theta^t = \theta^0 + \sum_{\tau=0}^{t} s^\tau \tag{4}$$

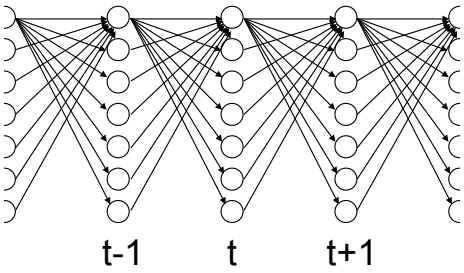

Fig. 2. The graph of the states $\{s_n^t\}$ used in Algorithm 2. The algorithm produces a directed path from this graph. The first node in each column represents the state s_0.

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

$$o^t = \theta^t + n^t \tag{5}$$

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. 2. 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. (5)), the latter being a function of *all* previous states $s^{0:t}$, not just of s^t (Eq. (4)).

To overcome this problem, one may reason as follows. Recall from basic theory [12] that the Viterbi algorithm computes the following:

$$\arg\max_{s^{0:t}} P(s^{0:T}|o^{0:T}) = \arg\max_{s^{0:T}} P(o^{0:T}|s^{0:T})P(s^{0:T}) \tag{6}$$

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

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

$$f^t(s^t) = \max_{s^{t-1}} P(o^{t-1}|s^{t-1})P(s^t|s^{t-1})f^{t-1}(s^{t-1}) \tag{7}$$

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

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:

$$P(o^t|s^t, R^t(s^t)) \tag{8}$$

where

$$R^t(s^t) = \{g^t(s^t), g^{t-1}(g^t(s^t)), \ldots, g^0(g^1(\ldots g^t(s^t)))\} \tag{9}$$

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 (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 (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. 2, is linear in the number of states, rather than quadratical.

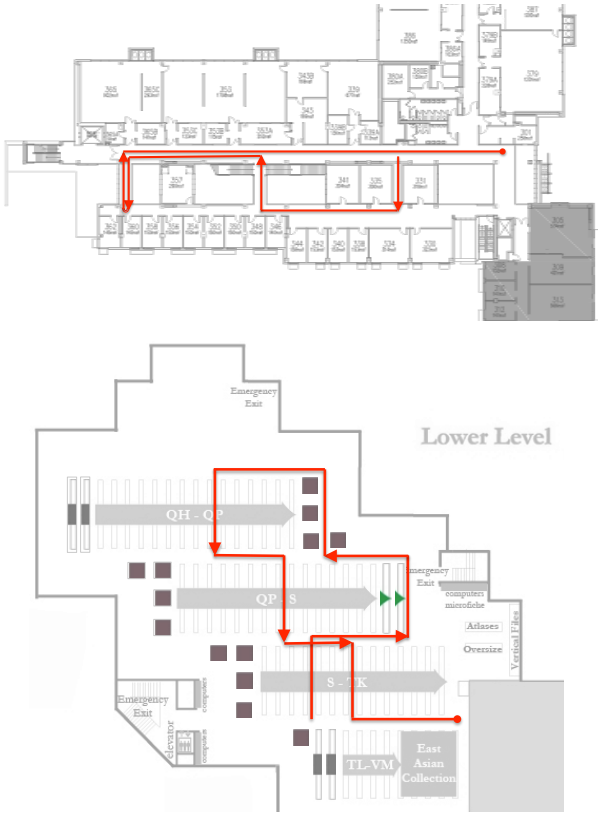

Fig. 3. Examples of indoor paths from our collection.

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.

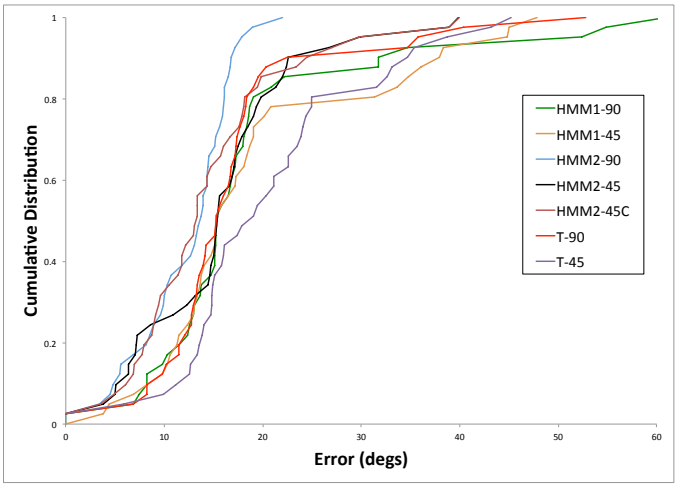

Fig. 4. Cumulative distribution of the error E (in degrees) defined in Eq. (11) over test paths for Algorithm 1 (HMM1) and Algorithm 2 (HMM2). HMM1-90, HMM2-90: only turns that are multiple of 90° considered. HMM1-45, HMM2-45: turns that are multiple of 45° considered. HMM2-45C: clustering postprocessing included (Sec. III-B1). 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. (11) is less than E_0.

IV. Experiments

A. 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. 3 shows two examples of paths taken in two different buildings.

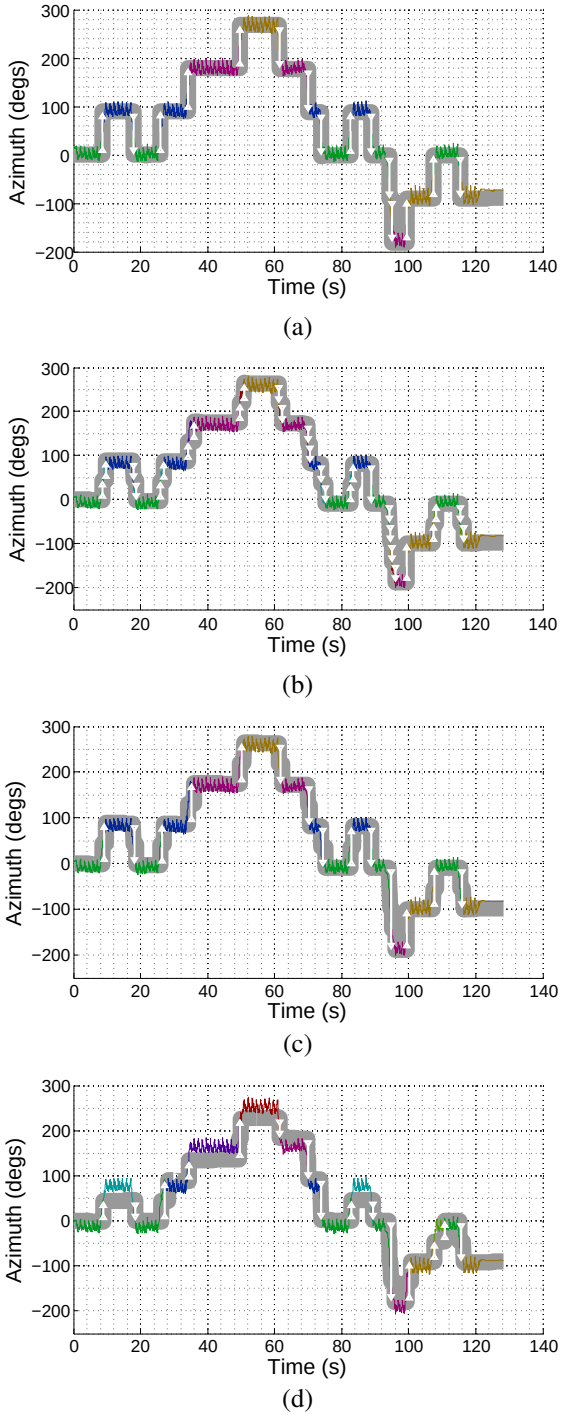

Fig. 5. Results on a path using Algorithm 2. (a) Only multiple of 90° turn angles considered. (b) Multiple of 45° turn angles considered. (c) Same as (b), with clustering postprocessing. (d) Same as (c), but without bias computation ($\theta^0 = 0^\circ$).

B. 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. One possibility could be to directly compare the sequences of "turn" events in the two cases, for example using a string matching algorithm. We chose a different approach, which 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:

$$\theta^t = \sum_{n:t^n \leq t} h^n \tag{10}$$

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:

$$E^2 = \sum_{t=0}^{T} (\theta^t_{\texttt{ld}} - \theta^t_{\texttt{out}})^2/(T+1) \tag{11}$$

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.

C. 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 $P(d^t|s^{t-1}_0) = P(-d^t|s^{t-1}_0) = p_m$, 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. III-A. 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.

D. Results

Fig. 4 shows the cumulative distribution of the errors E, defined in Eq. (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. III-B1 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).

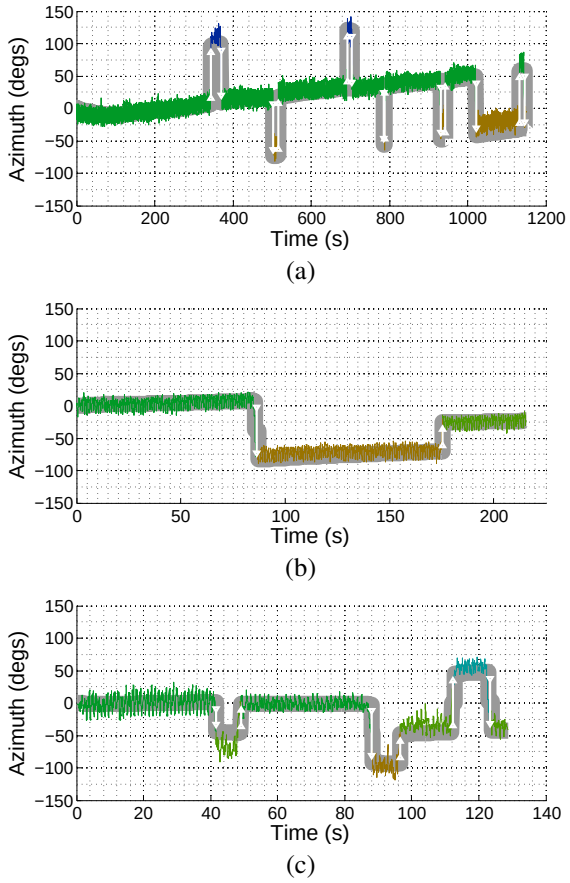

Fig. 6. Examples of outdoor paths, processed by Algorithm 2 (multiple of $45°$ turn angles considered, clustering postprocessing).

Fig. 5 shows an example with a path processed by various variants of Algorithm 2. Without the clustering mechanism described in Sec. III-B1, $90°$ turns are almost always detected as pairs of $45°$ (Fig. 5 (b)). This situation is corrected for the most part by the clustering algorithm (Fig. 5 (c)). The importance of the correct choice of "bias" θ^0 via the method discussed in Sec. III-A is highlighted by Fig. 5 (d), obtained without bias computation ($\theta^0 = 0°$). Note how, without bias correction, multiple turns are incorrectly measured.

Fig. 6 shows three examples with outdoor paths (Algorithm 2, turns by multiple of $45°$, clustering). The long stretch with very noticeable drift shown in Fig. 6 (a) is well tracked by our system (note how the azimuth θ defined in (4), marked by the thick grey line, precisely models the accumulated drift). Fig. 6 (b) and (c) show example of turns by $45°$, which are correctly detected by the algorithm.

Finally, in Fig. 7, we show 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. Although we have not yet run a formal evaluation with a significant population of blind walkers, these initial trials gave promising results. 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

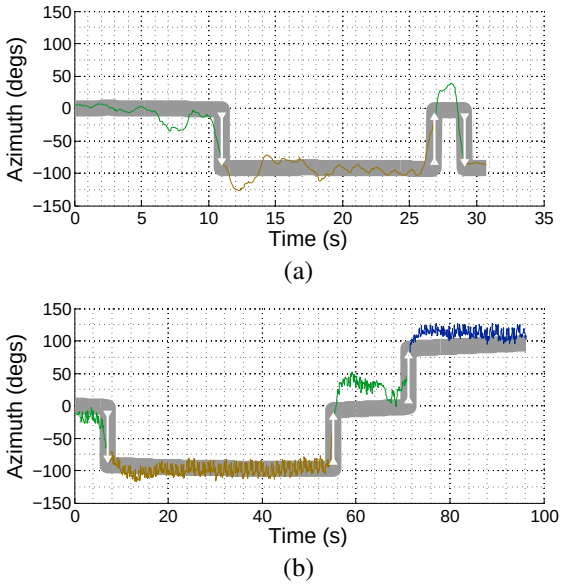

Fig. 7. Examples with data collected by two blind walkers.

they are unfamiliar with), often veer in their path [10], and take turns more slowly. Some of these characteristics are visible in the plots of Fig. 7.

V. Conclusions and Future Work

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°.

In future research we will address two main topics. First, we will run extensive experiments with blind participants, expanding the pilot results described in Sec.IV-D, in order to understand what modifications (if any) are necessary for correct tracking of the gait of blind walkers using a white cane. Second, we will design a "safe return" system that will track the user as he or she is tracing back a path taken previously. This system will run an online version of our turn detector. Based on the representation of the path in terms of turns and number of steps counted between each turn, this system will be able to assess the approximate location of the walker while he or she is retracing the same path, and will provide directions through synthetic speech as to when to expect a turn and which direction to turn in.

Acknowledgment

Professor Stefano Carpin of UC Merced is kindly acknowledged for his help collection the data shown in Fig. 1.

References

[1] H. Liu, H. Darabi, P. Banerjee, and J. Liu, "Survey of wireless indoor positioning techniques and systems," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 37, no. 6, pp. 1067–1080, 2007.

[2] Y. Gu, A. Lo, and I. Niemegeers, "A survey of indoor positioning systems for wireless personal networks," *Communications Surveys & Tutorials, IEEE*, vol. 11, no. 1, pp. 13–32, 2009.

[3] N. Fallah, I. Apostolopoulos, K. Bekris, and E. Folmer, "Indoor human navigation systems: A survey," *Interacting with Computers*, vol. 25, no. 1, pp. 21–33, 2013.

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

[5] T. H. Riehle, S. M. Anderson, P. A. Lichter, N. Giudice, S. Sheikh, R. Knuesel, D. Kollmann, and D. Hedin, "Indoor magnetic navigation for the blind," in *Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE*. IEEE, 2012, pp. 1972–1975.

[6] E. Foxlin, "Pedestrian tracking with shoe-mounted inertial sensors," *Computer Graphics and Applications, IEEE*, vol. 25, no. 6, pp. 38–46, 2005.

[7] S. Beauregard and H. Haas, "Pedestrian dead reckoning: A basis for personal positioning," in *Proceedings of the 3rd Workshop on Positioning, Navigation and Communication*, 2006, pp. 27–35.

[8] I. Apostolopoulos, N. Fallah, E. Folmer, and K. E. Bekris, "Integrated online localization and navigation for people with visual impairments using smart phones," *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 3, no. 4, p. 21, 2014.

[9] J. Z. Flores and R. 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*. Springer, 2014, pp. 411–418.

[10] S. A. Panëels, D. Varenne, J. R. Blum, and J. 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.

[11] J. Liu, R. Chen, L. Pei, R. Guinness, and H. Kuusniemi, "A hybrid smartphone indoor positioning solution for mobile lbs," *Sensors*, vol. 12, no. 12, pp. 17 208–17 233, 2012.

[12] L. Rabiner and B.-H. Juang, "An introduction to hidden markov models," *ASSP Magazine, IEEE*, vol. 3, no. 1, pp. 4–16, 1986.