Title

Robust Indoor Pedestrian Backtracking Using Magnetic Signatures and Inertial Data

Authors

Hsuan Tsai, Chia

Manduchi, Roberto

Publication Date

2024-10-14

Robust Indoor Pedestrian Backtracking Using Magnetic Signatures and Inertial Data

Chia Hsuan Tsai
Department of Computer Science & Engineering
University of California, Santa Cruz
Santa Cruz, USA
ctsai24@ucsc.edu

Roberto Manduchi
Department of Computer Science & Engineering
University of California, Santa Cruz
Santa Cruz, USA
manduchi@ucsc.edu

***Abstract*—Navigating unfamiliar environments can be challenging for visually impaired individuals due to difficulties in recognizing distant landmarks or visual cues. This work focuses on a particular form of wayfinding, specifically backtracking a previously taken path, which can be useful for blind pedestrians. We propose a hands-free indoor navigation solution using a smartphone without relying on pre-existing maps or external infrastructure. Our hybrid matching method integrates machine learning to enhance positioning accuracy, addressing real-life challenges such as odometry errors or deviations from the correct path. Testing with datasets from visually impaired individuals demonstrates the potential of our approach in providing reliable backtracking assistance.**

Keywords—indoor navigation, accessibility technology, dynamic programming, machine learning,

I. Introduction

For visually impaired individuals, traveling in unfamiliar environments can be difficult and potentially unsafe due to challenges in recognizing distant landmarks or other visual cues. Path integration is a common mechanism used by visually impaired individuals to traverse routes [1]. While some can develop precise spatial awareness, others may only build limited one-dimensional route information [2]. Systems designed to aid in wayfinding can significantly enhance their opportunities for learning, employment, independent living, and social engagement.

A widely used localization technology, GPS, is highly accurate but impractical for indoor navigation due to signal blockages inside buildings. Various studies have explored reliable indoor wayfinding methods for visually impaired individuals, including BLE beacons, RFID, and Wi-Fi-based navigation. These techniques require external infrastructure and/or environment fingerprinting, which can be labor-intensive, time-consuming, and costly [3], [4], [5]. Navigation apps built on Apple's ARKit leverage visual sensors and powerful AI systems to provide real-time positional data and information [6]. However, the requirement for users to carry and orient the smartphone in a specific manner can be impractical for blind travelers. In addition, most applications (except [6]) also require access to indoor maps, which are not always available. These challenges hinder the widespread adoption of indoor navigation systems for visually impaired individuals.

This work focuses on a solution that provides hands-free indoor navigation using a smartphone without relying on pre-existing maps or external infrastructure. Fig. 1 illustrates this concept using a simple example. A blind patient is guided by a receptionist from a waiting room to a doctor's office (*way-in path*). After the appointment, the patient may need to walk back to the waiting room independently if assistance is unavailable. This *return path* (from point B to A) can be facilitated by our backtracking system, which is designed to help users retrace their path to return to the starting point. Similar concepts have been proposed for this application [6], [7], [8], [9]. However, real-life challenges such as odometry errors, mis-detected steps or turns, and deviations from the correct path complicate backtracking. We propose a new hybrid-matching method integrated with machine learning to enhance positioning accuracy, mitigating these challenges and guiding users back on track.

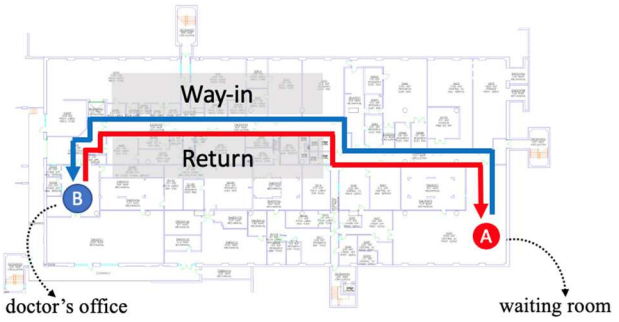

Fig. 1. A hypothetical path of a blind patient for a doctor's appointment. The patient began in the waiting room (A) and was guided by the receptionist to the doctor's office (B). The path from A to B is the way-in path. After the appointment, the patient retraces the route back to the waiting room (from B to A), which is the return path.

In this article, we begin by introducing the basic path-matching algorithm, which utilizes magnetic field data and steps/turns information to backtrack users' positions. We then discuss the challenges associated with backtracking for visually impaired individuals. Subsequently, we propose a new hybrid matching method integrated with machine learning to enhance the system's performance and achieve more robust results.

II. Preliminaries and Related Work

A. Pedestrian Dead Reckoning (PDR)

PDR tracks users' positions based on their steps and azimuth, which is obtained by integrating sensor data from gyros and accelerometers [10]. However, PDR error accumulates over time due to sensor noise. A two-stage system with a "straight-walking" detector and a Mixture Kalman Filter (MKF) has been used to track orientation drift [11]. Step count information can be obtained using an LSTM recurrent network [12]. Following the approach in [7], we consider that in structured buildings, a path can often be described as a sequence of straight segments and turns with discrete turning angles (multiples of 90º or 45º), known as turns/steps representation. This robust turns/steps detector effectively

reduces accumulated PDR error, and is robust to phone placement on one's body [11].

B. Backtracking System for Blind Individuals

The backtracking system for blind individuals was introduced by Flores and Manduchi [7] for navigation in buildings without maps. It uses inertial data to track steps and detect turns as users traverse a path. When retracing steps, the system compares the current position against the recorded path, providing directions based on remaining turns and steps. However, large errors can occur if steps or turns are mis-detected. FollowUs [8] by Microsoft uses magnetic signatures and inertial data to retrace a user's path, but it is designed for sighted users who can manage system errors. Clew [6], based on visual odometry and Apple's ARKit, helps visually impaired users retrace routes but requires a clear camera view, which may be inconvenient for some users.

C. Magnetic Signature

Magnetic fields are increasingly used for indoor navigation due to their unique characteristics within indoor environments [8], [13], [14], [15]. This method relies on the Earth's magnetic field and ferromagnetic objects, creating distinct magnetic signatures. However, the magnetic field can be temporarily affected by other factors (e.g., a running elevator), which may lead to inaccurate positioning. Smartphones with magnetometers provide an affordable and infrastructure-free platform for magnetic-based navigation. From the measured 3D magnetic field vector and from knowledge of the gravity direction (from the phone's accelerometers), one may derive a 2D vector that is invariant to the phone orientation [16], [17]. The equations below describe this 2D vector:

$$M_g = \frac{<\vec{M},\vec{g}>}{||\vec{g}||}, M_h = \left(\left|\left|\vec{M}\right|\right|^2 - \frac{<\vec{M},\vec{g}>^2}{||\vec{g}||^2}\right)^{0.5}$$

where $\vec{M}$ is the 3D magnetic field in the device's reference frame, and $\vec{g}$ is the gravity vector. M_g is the magnetic field in the gravity direction, and M_h is the magnitude of the magnetic field's projection on the horizontal plane.

III. Method

Our system's input data includes step counts, detected 90° or 45° turns, and 2D magnetic field data recorded along the path. The system operates in two phases: *way-in* and *return*. During the way-in phase, PDR reconstructs the user's trajectory as a polyline from the step count and the detected turns (see Fig. 2). In the return phase, the user starts from the endpoint of the way-in path and walks in the reverse direction. The objective of our system is to identify the location on the way-in path that best matches the user's current location during the return. This information can be used to provide appropriate guidance to the walker.

A straightforward solution would be to use PDR to create a polyline for the return path and compare it against the way-in polyline to find the best match to the user's current position. In theory, if accurate odometry data is acquired, all that is needed is to find the closest position in the way-in path to the current location in the return. This is shown in Fig. 2 (a), where positions for the way-in and return paths that are matched in this way are depicted by blue and red dots, respectively.

However, this method faces real-life challenges, such as different step lengths between way-in and return journeys, resulting in different step counts in the paths (see Fig. 2 (b)) or missed/falsely detected turns (see Fig. 2 (c)). These challenges highlight the need for sequence alignment algorithms to improve performance. Our approach is based on a graph-based path-matching algorithm, described below.

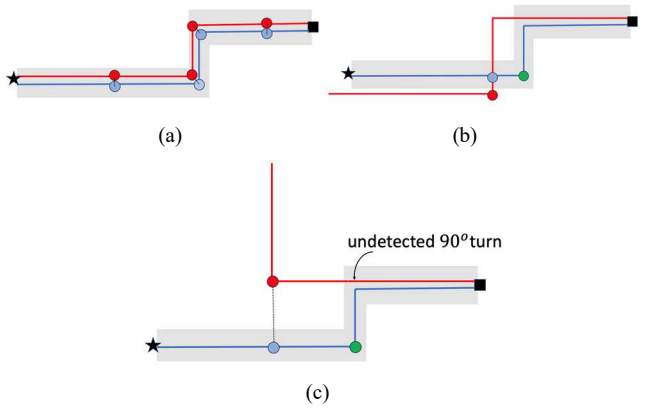

Fig. 2. Real-life challenges in matching the way-in and return paths. Red and blue lines represent return and way-in trajectories reconstructed by PDR. The start and end points for the way-in path are indicated by a square and a star. Blue and red dots show matched positions based on the closest distance. Green dots in (b) and (c) represent the correct positions of the red dots relative to the way-in path. Each segment's length is proportional to the number of steps in that segment. (a) Ideal situation with easily matched positions. (b) Shorter steps during the return result in more steps taken for the same segment length and thus, longer segment lengths in the return polyline, leading to incorrect matches. (c) The first turn in the return path went undetected.

A. Path-Matching Algorithm

Instead of matching spatial locations, we use a path-matching algorithm to match time sequences based on an appropriately defined graph $\mathcal{G}$ and dynamic programming to find the minimum cost path. This method, originally introduced by [18], was later developed in [9], [19]. We mainly consider the magnetic field vectors and detected turns to define the node cost and edge cost in the graph $\mathcal{G}$. Step detection is considered implicitly: the sequences of time indices are defined such that there are three regularly spaced time intervals between two consecutive detected steps. A primary source of node cost is the 2D magnetic discrepancies between matched way-in and return samples, while edge costs are based on orientation discrepancies associated with detected turns. This algorithm can handle variations in step length and mis-detected turns. (For details on this algorithm, please see[9], [19].)

Fig. 3 shows a simple graph $\mathcal{G}$, where j samples of the return have been recorded in this case. The sequence of matched way-in and return samples is determined by the minimum cost path $S(j)$ in the graph computed using dynamic programming. $S(j)$ is formed by the pairs of matched indices terminating in (i_j, j). Note that the minimum cost sequence is computed from all way-in samples and the available return samples (up to j). When a new return sample is available, the minimum cost path is recomputed. It is important to note that the new minimum cost sequence $S(j+1)$ may or may not contain the previous sequence $S(j)$. In particular, the last pair of matches $(i_{j+1}, j+1)$ in $S(j+1)$ may, in some circumstances, be largely different from (i_j, j).

Research reported in this publication was supported by the National Eye Institute of the National Institutes of Health under award number R01EY029260-01. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

This is relevant because the last matches in a minimum cost sequence represent the system's knowledge about the walker's current position, which is used to provide guidance. An abrupt change in the estimated position may lead to inconsistent guidance. As an example, consider the situation shown in Fig. 4. In this figure, horizontal and vertical lines represent left or right turns detected during the way-in and return, respectively. During the return phase, the sequence of matched indices shifted significantly between return sample indices #150 and #180. The walker did not change their orientation during this period. While utilizing this information for navigation, at return sample index #150, the system might instruct the user to make a **left turn** at the next junction, but at return sample index #180, it may change to instruction of a **right turn**. These contradictory guidance instructions are clearly undesirable.

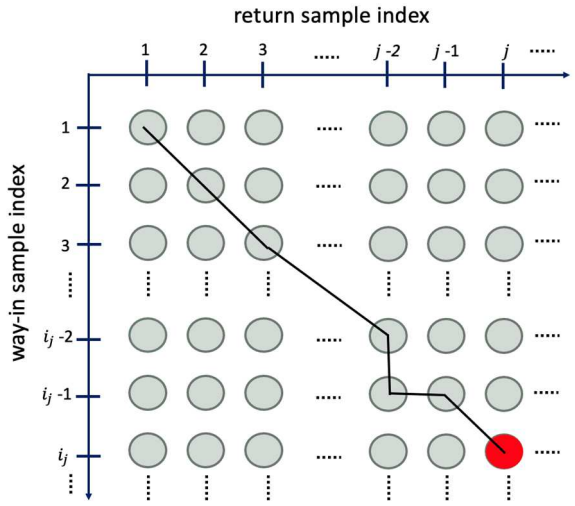

Fig. 3. Calculated global path (black line) in the graph $\mathcal{G}$ for return samples up to index j. The end of global path $S(j)$ is the red dot at (i_j, j), indicating the mapped position for return sample index j relative to the way-in path.

B. *Hybrid-Matching Algorithm Integrated with Machine Learning*

In order to reduce the likelihood of inconsistent directions, we propose a hybrid-matching algorithm that combines the path-matching and PDR approaches. As mentioned earlier, path-matching offers higher resilience than PDR to variable step length and incorrect turn detection but is liable to provide inconsistent direction when the minimum cost path changes abruptly upon receiving new return samples. We introduce a new measure of *reliability* for the current minimum cost path, where *reliability* measures the likelihood that the last match (i_j, j) of the current sequence $S(j)$ will be preserved in future sequences $S(k),\ k > j$. If the latest minimum cost path is deemed unreliable, we switch to the PDR algorithm, which, as explained earlier, generates a sequence of user positions based on the minimum distance matching of the reconstructed path. In this case, the reconstructed return sequence starts from the *last reliable position (LPR)*, that is, the position in the reconstructed way-in path corresponding to the last endpoint (i_j, j) of the latest minimum cost sequence in the graph that was found to be reliable. As soon as a minimum cost path is found to be reliable, the system switches back to the path-matching algorithm. In the following, we describe two approaches to estimate the reliability of a minimum cost path.

C. *Hybrid-Matching Algorithm Integrated with Machine Learning*

In order to reduce the likelihood of inconsistent directions, we propose a hybrid-matching algorithm that combines the path-matching and PDR approaches. As mentioned earlier, path-matching offers higher resilience than PDR to variable step length and incorrect turn detection but is liable to provide inconsistent direction when the minimum cost path changes abruptly upon receiving new return samples. We introduce a new measure of *reliability* for the current minimum cost path, where *reliability* measures the likelihood that the last match (i_j, j) of the current sequence $S(j)$ will be preserved in future sequences $S(k),\ k > j$. If the latest minimum cost path is deemed unreliable, we switch to the PDR algorithm, which, as explained earlier, generates a sequence of user positions based on the minimum distance matching of the reconstructed path. In this case, the reconstructed return sequence starts from the *last reliable position (LPR)*, that is, the position in the reconstructed way-in path corresponding to the last endpoint (i_j, j) of the latest minimum cost sequence in the graph that was found to be reliable. As soon as a minimum cost path is found to be reliable, the system switches back to the path-matching algorithm. In the following, we describe two approaches to estimating the reliability of a minimum cost path.

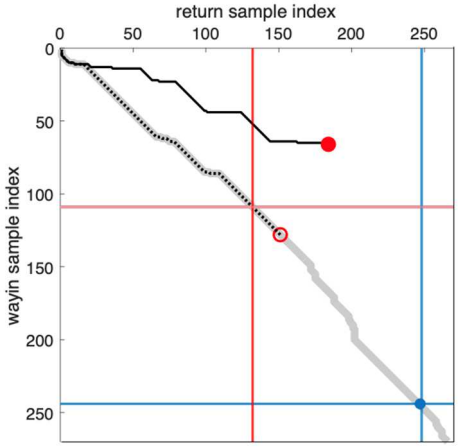

Fig. 4. Examples of paths in the graph $\mathcal{G}$. The thick gray line shows the global path $S(j = the\ end\ of\ return\ path)$ after all return data is available. The black line and dashed black line are global paths computed from return data up to sample indices #180 and #150, respectively. Solid red and hollow circles at the end of global paths indicate the current match observed for different return sample indices. Red and blue lines represent times at which a left turn and right turn were detected during the way-in (horizontal lines) and return (vertical lines).

Linearly Defined LRP

A simple method to find the LRP (i.e., to determine whether a minimum cost path is reliable) was originally proposed in [9], and briefly summarized here. This algorithm measures the local properties of the current minimum cost path in the graph. In practice, we examine the last N matches (N=21, corresponding to samples recorded in the last 7 steps) in the path and assume that the path is reliable if these matches form a line in the graph with a unitary slope. In practice, this means that consecutive time instants during return are matched one by one with consecutive time instants during way in. We found empirically that when this is the case, the path is normally reliable. The LRP thus found is denoted as LRP_{local}.

LRP Defined Through Machine Learning

The linearly defined LRP method described above is rather simplistic, and in addition, it does not consider information about the magnetic field that could be useful for reliability determination. We thus introduce a new method based on machine learning to incorporate more information in identifying the LRP. For time instant J during the return, we consider the following information: (1) the terminal part (last N samples) of the current minimum cost path $S(J)$; (2) the 2D magnetic field measured in the return and way-in paths for these matches; (3) the matched pairs (i_j, j) that are the endpoint of the last N global paths $S(j)$ for $J - N < j \leq J$. This data is fed to a neural network tasked with determining whether the current minimum cost path is reliable (see Fig. 5.) The LRP thus found is denoted as LRP_{NN}.

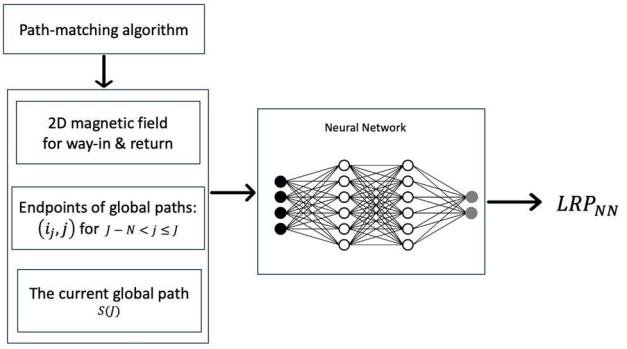

Fig. 5. Neural network architecture for determining the last reliable position (LRP_{NN}).

To train the neural network, we need to collect a representative data set, where each minimum cost path is correctly labeled as reliable or not reliable. This ground truth data is denoted as LRP_{gt}. We recorded datasets from seven blind individuals in a user study on a prior version of the system [9] (*user study dataset*), along with another dataset collected by members of our team walking in a university building (*E2 dataset,* from the name of the building). The *user study* dataset (4392 data samples) contains 12 paired paths with 4 to 5 turns in each path, for distances ranging from 72m to 123m. The *E2 dataset* (42313 data samples) from controlled experiments with intentional path variations in the return paths, includes 63 paired paths with 2-8 turns in each path, and distances ranging from 67m to 220m. Both datasets include recordings of magnetic data, steps, and turns.

Determination of reliability of any minimum cost path $S(j)$ at any return time is obtained by evaluating whether the endpoint (i_j, j) of the path is contained in the *global* minimum cost path, that is the minimum cost path from *all* samples in the recorded return. If this is the case, $S(j)$ is deemed to be reliable. In practice, we relax this criterion by allowing (i_j, j) to be within a maximum distance from the global minimum cost path. Specifically, this maximum distance is approximately 7.5 sample units or 2.5 steps, given that there are three samples taken between each step. This number was determined through trial and error in our initial experiments.

We tested five different types of networks to determine the last reliable position LRP_{NN}: Fully-Connected Network (FCN), Long Short-Term Memory (LSTM), 1D Convolutional Network (CONV), Graph Neural Network (GCN) [20] and Graph Attention Network (GAT) [21]. Each network has a similar number of parameters, including one input layer, one output layer, and one hidden layer, totaling around 7K parameters. In the case of FCN, the input data were flattened and fed into the network. For GAT and GCN, we used a graph representation of the data, dividing it into two groups of nodes: way-in nodes and return nodes. Each group contains the following features: two-dimensional magnetic field data, matched indices pair (i_j, j) for $J - N < j \leq J$, and the current global path $S(J)$.

IV. Results

As mentioned earlier, we have collected two datasets to train the neural network. The user study dataset serves primarily for testing purposes because it is the actual dataset collected from seven visually impaired participants and all the datasets, including the user study dataset and E2 dataset, were both used for training. We ran experiments using the leave-one-person-out modality: during testing, the model was tested on the "left-out" participant in the user study dataset while being trained by the combination of the rest of the data in the user study dataset and the E2 dataset.

A. Error Metrics

The error is calculated based on the reconstructed return path trajectory generated by different methods and compared against the trajectory generated by the ground truth LRP (LRP_{gt}). The average error per sample is given by :

$$E_{gt} = \frac{\sum_{i=1}^{M} ||(\hat{x}_i, \hat{y}_i) - (x_{gt,i}, y_{gt,i})||}{M},$$

where $(x_{gt,i}, y_{gt,i})$ are the ground truth coordinates, and $(\hat{x}_i, \hat{y}_i)$ are the calculated coordinates based on the proposed methods.

Table I shows the errors E_{gt} (in meters) of the reconstructed trajectory based on different methods of predicting LRP. It is important to note that in the last row of the table, the "Baseline" method refers to the basic path-matching algorithm. The last two columns show the mean and standard deviation for each method. Additionally, since no training is involved for the "Linearly defined LRP" and "Baseline" methods, the results for each participant column represent the corresponding test outcomes directly generated by the methods.

The lowest error is obtained based on the neural network, specifically FCN, while several neural networks outperform the "Linearly defined LRP" and "Baseline" methods, as shown in the gray-highlighted cells in Table I. This demonstrates the potential of using the hybrid-matching algorithm integrated with machine learning to reconstruct the user's position for backtracking.

Fig. 6 illustrates examples of reconstructed trajectories using different methods: the basic path-matching algorithm (a) hybrid matching (b) and traditional PDR (c). In Figure Fig. 6 (a), the trajectory reconstructed using PDR shows a shorter return path than the way-in path, likely due to the walker taking larger steps during the return, resulting in fewer steps detected. This discrepancy makes it challenging and error-prone to match the current user location with the way-in path. Fig. 6 (b) demonstrates that the hybrid matching method successfully matches the user's position to the destination, even without any reliable match near the end of the route due to a significant magnetic field discrepancy, as illustrated by the white horizontal line in the magnetic discrepancy plot in Fig. 6 (d). In contrast, Fig. 6 (c) shows that the basic path-matching algorithm fails to provide an accurate match near the destination.

Table I.
ERROR METRIC (METERS) COMPUTER BASED ON DIFFERENT METHODS

		Left-out participant								
		P1	P2	P3	P4	P5	P6	P7	Mean	Std
LRP_{NN}	GAT	0.56	1.60	0.90	0.93	0.74	0.95	0.35	0.86	0.39
	GCN	0.90	3.13	1.07	3.21	0.85	1.09	5.38	2.23	1.73
	FCN	**0.53**	**1.05**	0.68	**0.51**	**0.62**	**0.24**	**0.22**	**0.55**	0.28
	LSTM	1.15	3.12	**0.62**	1.12	1.50	1.21	2.65	1.62	0.91
	CONV	0.67	1.66	0.91	1.52	1.09	0.70	0.76	1.04	0.40
LRP_{local} (Linearly defined LRP)		1.63	2.68	0.94	0.85	1.12	0.90	2.63	1.54	0.81
Baseline		0.82	3.20	1.52	3.64	1.98	1.6	3.62	2.34	1.13

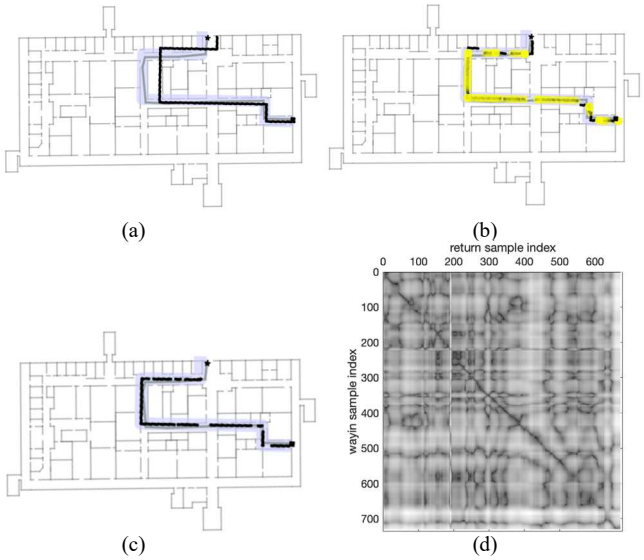

Fig. 6. Examples of reconstructed return paths and the magnetic discrepancy (a) PDR using step and turn information. (b) Hybrid matching with machine learning. (c) Basic path-matching algorithm. (d) Magnetic discrepancy plot for all possible pairs (i_j, j) of samples from way-in (vertical axis) and return (horizontal axis). Lighter gray indicates larger discrepancies. The way-in path is a thick purple line ending at the black square. The gray line is the approximate return path of the participant. Reconstructed return paths are shown in black lines, with reliable matches in (b) indicated by yellow circles.

When there is a mis-detected turn (false-positive turn), or the user deviates from the correct path, such as missing a turn and traversing a previously unvisited path, the hybrid matching integrated with machine learning can accurately locate the user's position and provide path-recovery notifications to guide them back. Fig. 7 illustrates such an example. In Fig. 7 (a), the reconstructed path based on PDR is entirely off due to a falsely detected turn near the beginning of the path, resulting in a misaligned return trajectory. Consequently, later in the journey, when the user misses a turn, PDR cannot map the user's position to the way-in path. In contrast, the hybrid matching method is able to map the user's position despite the mis-detected turn. As shown in the green highlighted area in Fig. 7 (b), there is a temporary misplacement of the user for a short period, however, as the user progresses along the return route, the method successfully remaps the user to the correct path. Additionally, when the user takes the wrong path, shown in the orange highlighted area, the hybrid matching with machine learning can project the user's position and guide them back to the correct path. The basic path-matching algorithm, however, is not capable of mapping the user's position when they deviate from the correct path. As mentioned previously, this method always calculated a matched position with respect to a position on the way-in path.

Fig. 8 shows that using LRP_{NN} determined by FCN can accurately map the user's position. The linearly defined LRP (LRP_{local}) misplaces the position after the last turn (highlighted area in Fig. 8). This demonstrates the superior performance of LRP_{NN} for hybrid matching, as demonstrated in the prior Table I.

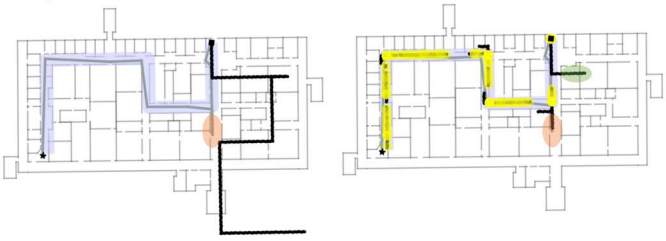

Fig. 7. See caption of Fig. 6. Examples of paths with a mis-detected turn (false-positive) and user deviation. (a) PDR. (b) Hybrid matching with machine learning. Orange highlights indicate wrong paths taken during return.

Fig. 8. See caption of Fig. 6. Comparison of return paths using hybrid matching with different LRP definitions. Black lines represent paths using hybrid matching with machine learning (LRP_{NN}, determined by FCN). The green line is the path using hybrid matching with a linearly defined LRP (LRP_{local}).

V. Discussion and Conclusion

Backtracking without a map is challenging, especially with odometry errors like mis-detected steps and turns. We developed a hybrid matching algorithm with machine learning to improve backtracking accuracy. This method uses step counts, turns, and magnetic signatures, alternating between the path-matching graph and PDR to perform hybrid matching. A neural network was adopted to determine the "last reliable position" (LRP) to assess current mapped positions. This hybrid approach is particularly helpful when, due to magnetic field fluctuations, the path-matching graph method gives inconsistent results (in which case, the system temporarily resorts to PDR, which is only based on steps/turns). We tested the system with datasets from seven visually impaired individuals and another dataset collected in a university building.

We used various neural networks to test the system. Results showed that FCN significantly improves path-matching accuracy. However, further research is needed to understand why other networks, like LSTM and GCN, did not perform as well. The limited dataset might have restricted these networks' ability to learn features during training.

The system assumes indoor environments consist of networks of corridors intersecting at certain angles (e.g., 90° or 45°; the dataset only contains turns of 90°). Its performance degrades with different angles or in open spaces. Using computer vision to periodically adjust the walker's position can improve performance but should minimize the need for the walker to hold the phone constantly.

Despite these limitations, the hybrid matching method with machine learning shows promising results in backtracking, especially when the user deviates from the correct path. Compared to the original path-matching algorithm, this approach offers enhanced reliability and accuracy.

REFERENCES

[1] J. M. Loomis, R. L. Klatzky, R. G. Golledge, J. G. Cicinelli, J. W. Pellegrino, and P. A. Fry, "Nonvisual navigation by blind and sighted: assessment of path integration ability.," J Exp Psychol Gen, vol. 122, no. 1, p. 73, 1993.

[2] R. G. Golledge, "Human wayfinding and cognitive maps," in The colonization of unfamiliar landscapes, Routledge, 2003, pp. 49–54.

[3] S. A. Cheraghi, V. Namboodiri, and L. Walker, "GuideBeacon: Beacon-based indoor wayfinding for the blind, visually impaired, and disoriented," in 2017 IEEE International Conference on Pervasive Computing and Communications (PerCom), IEEE, 2017, pp. 121–130.

[4] R. Sammouda and A. Alrjoub, "Mobile blind navigation system using RFID," in 2015 Global Summit on Computer & Information Technology (GSCIT), IEEE, 2015, pp. 1–4.

[5] J. Hurtuk, J. Červeňák, M. Štancel, M. Hulič, and P. Fecil'ak, "Indoor navigation using IndoorAtlas library," in 2019 IEEE 17th International Symposium on Intelligent Systems and Informatics (SISY), IEEE, 2019, pp. 139–142.

[6] C. Yoon, R. Louie, J. Ryan, M. Vu, H. Bang, W. Derksen and P. Ruvolo, "Leveraging augmented reality to create apps for people with visual disabilities: A case study in indoor navigation," in The 21st International ACM SIGACCESS Conference on Computers and Accessibility, 2019, pp. 210–221.

[7] G. Flores and R. Manduchi, "Easy return: an app for indoor backtracking assistance," in Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, 2018, pp. 1–12.

[8] Y. Shu, Z. Li, B. Karlsson, Y. Lin, T. Moscibroda, and K. Shin, "Incrementally-deployable indoor navigation with automatic trace generation," in IEEE INFOCOM 2019-IEEE Conference on Computer Communications, IEEE, 2019, pp. 2395–2403.

[9] C. H. Tsai, F. Elyasi, P. Ren, and R. Manduchi, "All the Way There and Back: Inertial-Based, Phone-in-Pocket Indoor Wayfinding and Backtracking Apps for Blind Travelers," arXiv preprint arXiv:2401.08021, 2024.

[10] Y. Jin, H.-S. Toh, W.-S. Soh, and W.-C. Wong, "A robust dead-reckoning pedestrian tracking system with low cost sensors," in 2011 IEEE International Conference on Pervasive Computing and Communications (PerCom), IEEE, 2011, pp. 222–230.

[11] P. Ren, F. Elyasi, and R. Manduchi, "Smartphone-based inertial odometry for blind walkers," Sensors, vol. 21, no. 12, p. 4033, 2021.

[12] M. Edel and E. Köppe, "An advanced method for pedestrian dead reckoning using BLSTM-RNNs," in 2015 International Conference on Indoor Positioning and Indoor Navigation (IPIN), IEEE, 2015, pp. 1–6.

[13] B. Li, T. Gallagher, A. G. Dempster, and C. Rizos, "How feasible is the use of magnetic field alone for indoor positioning?," in 2012 International Conference on Indoor Positioning and Indoor Navigation (IPIN), IEEE, 2012, pp. 1–9.

[14] W. Storms, J. Shockley, and J. Raquet, "Magnetic field navigation in an indoor environment," in 2010 Ubiquitous Positioning Indoor Navigation and Location Based Service, IEEE, 2010, pp. 1–10.

[15] J. Kuang, X. Niu, P. Zhang, and X. Chen, "Indoor positioning based on pedestrian dead reckoning and magnetic field matching for smartphones," Sensors, vol. 18, no. 12, p. 4142, 2018.

[16] X. Fan, J. Wu, C. Long, and Y. Zhu, "Accurate and low-cost mobile indoor localization with 2-D magnetic fingerprints," in Proceedings of the First ACM Workshop on Mobile Crowdsensing Systems and Applications, 2017, pp. 13–18.

[17] Apple, "deviceMotion," Apple Developer Documentation. [Online]. https://developer.apple.com/documentation/coremotion

[18] T. H. Riehle et al., "Indoor magnetic navigation for the blind," in 2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society, IEEE, 2012, pp. 1972–1975.

[19] C. H. Tsai, P. Ren, F. Elyasi, and R. Manduchi, "Finding Your Way Back: Comparing Path Odometry Algorithms for Assisted Return," in 2021 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops), IEEE, 2021, pp. 117–122.

[20] J. Zhou et al., "Graph neural networks: A review of methods and applications," AI open, vol. 1, pp. 57–81, 2020.

[21] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," International Conference on Learning Representations (ICLR), 2018.