All the Way There and Back: Inertial-Based, Phone-in-Pocket Indoor Wayfinding and Backtracking Apps for Blind Travelers

CHIA HSUAN TSAI*, FATEMEH ELYASI*, PENG REN*, and ROBERTO MANDUCHI, University of California, Santa Cruz, USA

We describe two iOS apps designed to support blind travelers navigating in indoor building environments. The Wayfinding app provides guidance to a blind user while following a certain route. The Backtracking app records the route taken by the walker towards a certain destination, then provides guidance while re-tracing the same trajectory in the opposite direction. Our apps only use the inertial and magnetic sensors of the smartphone, and thus require no infrastructure modification (e.g., installation and support of BLE beacons). Unlike systems that use the phone's camera, users of our apps can conveniently keep their phone tucked inside their pocket, while interacting with the apps using a smartwatch. Routing directions are given via speech. Both apps were tested in a user study with seven blind participants who used them while navigating a campus building. Participants were able to successfully use the Wayfinding app to complete the prescribed paths (3 paths each), although the app had to be restarted for the first three participants in one path due to incorrect step length measurements (the app was later modified to track the users' step length). The Backtracking app worked well in most cases, although in 6 trials (out of 21) the app lost track of the participant's location.

CCS Concepts: • **Human-centered computing → Accessibility technologies** ; *Human-centered computing*.

Additional Key Words and Phrases: wayfinding, orientation, mobility

ACM Reference Format:
Chia Hsuan Tsai, Fatemeh Elyasi, Peng Ren, and Roberto Manduchi. 2024. All the Way There and Back: Inertial-Based, Phone-in-Pocket Indoor Wayfinding and Backtracking Apps for Blind Travelers. 1, 1 (September 2024), 36 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Navigating a building can be confusing and disorienting for anyone who is visiting the building for the first time. For those who are blind, this experience can be even more challenging, as these travelers cannot rely on visual feedback. This is compounded by the fact that, compared with outdoor environments, building interiors are often self-similar (e.g., doors repeating at regular intervals along a corridor) and devoid of non-visual features or landmarks that can help with orientation (e.g., the sound of traffic, the presence of posts at specific locations or of curb edges). It is important to note that the problem addressed here is that of *orientation/wayfinding*, which is distinct from what is normally termed *mobility* ("moving safely, gracefully, and comfortably through the environment" [72]; see also [27, 59]). Mobility support is normally provided by the use

*All three authors contributed equally to this research.

Authors' address: Chia Hsuan Tsai, ctsai24@ucsc.edu; Fatemeh Elyasi, felyasi@ucsc.edu; Peng Ren, pren1@ucsc.edu; Roberto Manduchi, manduchi@ucsc.edu, University of California, Santa Cruz, 1156 High Street, Santa Cruz, California, USA.

of long canes and dog guides, with technological alternatives represented by smart canes [62] and robotic dogs [33]).

While multiple research efforts have focused on accessible technology to support independent travel in indoor space, at the time of this writing there are no commercially available systems ready for widespread use. Indoor spaces are challenging because GPS cannot be relied upon. This doesn't mean that indoor localization is not feasible: a variety of techniques have been demonstrated, some of which have been adapted for navigation by blind individuals. One example is NavCog in its various versions [3, 46, 56, 57], which uses Bluetooth Low Energy (BLE) technology for accurate localization. Ultra-wide band (UWB) technology has also been used successfully for positioning [6]. Unfortunately, BLE-based navigation is only possible if a specific infrastructure (a possibly large set of BLE beacons) has been installed, and a laborious calibration procedure (*fingerprinting*) has been conducted. This raises questions of scalability, as the success of this technology hinges on the good will of the agency that manages the building. It would thus be desirable that the technology powering the navigational aid would not depend on external dedicated infrastructure.

In recent years, and with the advent of powerful AI, there has been intense interest in systems that use visual sensors (e.g., the camera embedded in a smartphone) to extract positional data, and to provide users with information about the world around them. For example, navigation apps built on Apple's ARKit have been developed specifically for use by blind travelers [15, 75]. Visual-inertial odometry technology, coupled with Particle Filtering, can produce extremely accurate localization. Its core strength, the use of visual data, is unfortunately also its main drawback. Users of these apps must hold the smartphone (in their hand, or perhaps attached to a lanyard [73]) in such a way that the smartphone's camera has a good view of the environment. This is not always possible, convenient, or desirable. Blind travelers normally use a long cane or a dog guide, and thus have one hand already occupied maneuvering the cane or holding the dog. Indeed, it has often been observed that navigational aid for blind walkers should be hands-free [45, 57].

In this article, we describe an experiment we conducted with 7 blind travelers who tested two apps created in our lab that were designed to assist with navigation in buildings characterized by a network of corridors. Both apps utilize data from the inertial sensors of a smartphone, and thus require no external infrastructure. Since the apps don't use data from the smartphone camera, users can conveniently keep the phone tucked in a pocket (indeed, this is how the apps have been tested in our experiment). The first one is a *Wayfinding* app: it has knowledge of the floor plan of the building to be navigated, computes the shortest length route to destination, and provides navigational support in an accessible way. The second one is a *Backtracking* app. Its sole purpose is to help an individual who has previously walked on a certain route (e.g., from the front entrance of a building to a certain office room) trace back their steps. While arguably less useful than a full-blown wayfinding application, backtracking support can help a blind traveler feel more confident in some situations. Importantly, the Backtracking app requires no prior knowledge of the building layout: it only uses data recorded during the first route traversal (*way-in*) to generate support for the user when walking back (*return*). Both apps have provisions for error recovery (i.e., can provide corrective directions if a participant misses a turn or takes the wrong turn).

These are the main contributions of this work:

- We demonstrate an accessible iPhone app (*Wayfinding*) that uses two different localization technologies, complemented by Particle Filtering, for inertial-based localization in a building with a known floor plan.
- We propose a new hybrid approach for backtracking that uses magnetic and inertial data, with no knowledge of the building layout (and thus usable in any venue, even without access to a floor plan). This algorithm was implemented and tested in the *Backtracking* app.

- We demonstrate the effectiveness of a simple speech-based user interface mechanism, consistent across both apps, designed to mitigate the unavoidable inaccuracy associated with the dead-reckoning nature of inertial-based localization.
- We describe a user study with 7 blind travelers. The participants first traversed three routes (292 meters and 13 turns in total) using our Wayfinding app; then, they traversed the same routes in the opposite direction using the Backtracking app.

This article is structured as follows. After reviewing the related work in Sec. 2, we describe our Wayfinding and Backtracking apps (including their shared user interface mechanisms) in Sec. 3. Our experiment with seven blind participants is described in Sec. 4. The results of this study, along with some limitations of our approach, are discussed in Sec. 5. Sec. 6 has the conclusions.

2 RELATED WORK

Wayfinding systems for blind travelers have been studied for decades. Early work includes TalkingSigns [7], which used beacons of infrared light modulated by a recorded speech signal. A user in the vicinity of the beacon could use a hand-held receiver that, when oriented towards the beacon, would decode and play the recorded speech. Soon after GPS was made available for civilian use in the early 1990s, researchers evaluated the possibility of using this localization technology to support blind navigation in the outdoors ([24], cited by [41]). Several accessible GPS-based apps are currently available, including Apple Maps, Google Maps, GoodMaps Outdoors, Nearby Explorer, and BlindSquare. For GPS-denied outdoor areas (e.g. when view of the satellites is occluded by tall buildings), other complementary technologies must be considered. For example, RouteNav [51] fuses information data from GPS and inertial sensors to overcome temporarily poor GPS localization. For indoor environments, which is the object of this paper, GPS simply cannot be utilized.

Indoor localization techniques can be divided into those that require some external infrastructure to be in place, and those that don't. When available, external infrastructure can enable accurate localization. Infrastructure that facilitates localization includes Bluetooth Low Energy (BLE) beacons [3, 13, 46, 47, 56, 57], RFID tags [26], and color tags [44]). Other localization technologies (e.g., Wi-Fi beaconing [1], and magnetic navigation [28, 52]) do not require specific infrastructure to be installed (e.g., Wi-Fi access points are normally already available), but require a laborious *fingerprinting* operation during which a "signature" feature, such as the vector of received signal strength (RSSI) from the Wi-Fi beacons or the magnetic field vector, is assigned to each location in the environment (note that fingerprinting is also required when using BLE beacons). An operation akin to fingerprinting is also required for localization systems that use images to recognize the user's location [55]. For example, as described in [49], the GoodMaps Explore app enables visual-based localization in indoor environments, but this requires an initial laborious phase where the 3-D structure of the environment is first acquired and annotated.

The second category of localization systems is made by those that do not require dedicated infrastructure or prior fingerprinting. These systems typically rely on dead reckoning. Given a known initial location and reference frame, the user's location is continuously updated with reference to these initial conditions based on sensor measurements. For example, Fusco et al. [15] used Apple's ARKit (a visual-inertial odometry library implemented on an iPhone) to track a blind user moving through a building. Visual odometry is a very effective modality, but requires a camera with an unoccluded field of view. The sensing modality considered in our work (inertial sensors) enables dead reckoning tracking with no constraints on the location of the sensors (e.g., users can keep the smartphone in their pocket).

Research on navigation systems for blind travelers using inertial sensing includes [4, 17, 51, 53]. Riehle et al. [53] described a classic pedestrian dead-reckoning system (PDR) with explicit turn detection, which was tested with 8 blind or low vision participants in a simple indoor route. Participants could request assistance from the experimenter if they felt they needed it. Compared to a condition in which the route was described at the beginning, but no guidance was provided during the trial, it was shown that use of the PDR system resulted in fewer requests for assistance made and in more instances of successful localization of the route's destination. Fallah et al. [17] and Apostolopoulos et al [4] proposed an inertial-based navigation system designed to overcome inherent localization inaccuracy through the *user as a sensor* modality. Routes were represented via a sequence of perceivable landmarks. Participants were asked to follow directions to the next landmark; once arrived at the landmark, they were asked to confirm it on the app. This ingenious strategy allowed the localization system to reset itself at the correct location for each detected landmark. The inertial system was based on step counting (measured by the accelerometer) and orientation sensing (measured by the compass). A Particle Filter was used to track the state (location and orientation) of the user through time. In addition, different methods were considered to estimate the user's step length using the Particle Filter. An experiment with 6 blind and low vision participants was conducted on two floors of a building with 10 routes. One route included climbing a staircase, while another route included taking an elevator. These routes contained multiple landmarks that needed to be discovered by the participants, including doors that needed to be counted, hallways, stairs, and ramps.

Our proposed Wayfinding system utilizes the same basic pedestrian dead reckoning (PDR) technology, complemented with Particle Filtering, as in the work of Apostolopoulos et al [4]. However, there are several critical differences between our work and that described in [4]:

(1) *Different user interface.* The work in [4, 17] relied on the *user as a sensor* modality to overcome localization errors due to accumulating drift. While this modality can be a powerful tool for mitigating ambiguous situations (see e.g. [51], where it was tested successfully in an indoor/outdoor wayfinding system), it can be demanding for the user, who is tasked with detecting and identifying each landmark. With our work, we decided to "push the envelope" and see whether a system that only relies on inertial sensor data, without input from the user, could be feasible. This involved a careful design of the user interface, which must account for the relatively poor spatial accuracy of inertial dead-reckoning.

(2) *More challenging routes.* In their tests with visually impaired participants, Apostolopoulos et al [4] defined ten routes, each with 1 to 3 turns (average: 1.7 turns per route). In our experiments, we considered three routes, two with 4 and one with 5 turns. Even though the total route length was shorter in our case (292 vs. 950 meters), we argue that the routes we selected (longer and with more turns) are a better choice for testing dead-reckoning localization. This is because, as well known, these systems are affected by drift, which accumulates in time and becomes particularly menacing for long trajectories.

(3) *Additional PDR algorithm considered.* While the PDR algorithm used to provide directions in our wayfinding tests (based on step count, step length tracking, and particle filtering) is similar to that of [4], our app also ran a different PDR in parallel, one that is based on a machine learning algorithm, RoNIN [74], followed by particle filtering. One relevant characteristic of RoNIN is that the trajectory it reconstructs is independent of the orientation of the phone with respect to the user. This is of great practical importance, as it allows users to re-orient the phone while walking (e.g. when picking up a call). Our tests show that the reconstruction accuracy with RoNIN and with the step-based algorithm are comparable, suggesting that RoNIN is a viable candidate for wayfinding applications for blind travelers.

(4) *Map-less backtracking modality.* In addition to standard wayfinding (which assumes knowledge of the building's map), we introduce and experiment with an app that supports backtracking of a previously taken route. Even though it still relies on inertial sensors (complemented with the magnetometer), the algorithm used for localization in this application is completely different from the PDR algorithm of [4]. Compared with the earlier version presented in [21], the new Backtracking app supports magnetic-assisted graph-based trajectory matching, enabling much more robust backtracking even when the user misses a turn or takes the wrong turn.

Along with localization techniques, prior research has focused on the design of accessible user interface modalities for navigational systems [31, 35, 38, 60, 77], using modalities such as speech [3, 17, 28], sound [19, 43, 54], and vibration [5, 20, 57]. For example, Fiannaca et al. [19] used sonification feedback to guide a blind user towards a door. The system produced a constant tone when walking towards the target, whose pitch indicated the distance to the target. A sequence of beeps was produced when veering off-target. Similarly, Manduchi and Coughlan [43] used beeps at two different frequencies to indicate the distance to a target. Azenkot et al. [5] used haptic feedback to help a blind user identify the direction to a target or a waypoint. A similar system (called "route compass") was employed in RoutNav, a wayfinding system for indoor/outdoor environments [51]. Flores et al. [20] experimented with a vibro-tactile belt for indoor route guidance. Microsoft Soundscape used 3-D audio cues to increase location and orientation awareness for blind travelers.

3 APPARATUS

We developed two iOS apps for this study: *Wayfinding* and *Backtracking*. The Wayfinding app has access to a floor plan of the building to be navigated. It is designed to find a route from the current user's location to the desired destination, update the route as necessary during traversal, and provide directions to the user as they are following the route. The Backtracking app has no prior information about the building layout. Similarly to [21], Backtracking operates in two phases. In the first phase (*way-in*), it simply tracks the route taken by the user to reach a destination. Then, in the *return* phase, it produces directions to help the user re-trace the same way-in route. To do so, the app progressively matches the partial return route with the way-in route (in reverse).

As will be made clear in the following, the Backtracking app needs to use different localization strategies than the Wayfinding app, as a consequence of its lack of building layout information. The user interface, however, was designed to be almost identical for the two apps, shielding the user from the different underlying mechanisms.

3.1 Wayfinding

3.1.1 Localization. We implemented two different pedestrian dead-reckoning (PDR) algorithms for user localization and tracking using data from the phone's inertial sensors. These two techniques are termed *Azimuth/Steps* and *RoNIN*, respectively [50]. A Particle Filter is applied to the output of either system. During the tests, we ran both systems in parallel, though only one of them was used to provide guidance to the user. The reason for running both algorithms in parallel was twofold. First, we wanted to have a "fail-safe" mechanism: if an algorithm failed to correctly track the participant, we could resort to switching to the other algorithm. Note that this only happened once in the whole experiment. Second, this approach allowed us to comparatively assess localization data from both algorithms in a variety of situations.

Both algorithms compute the user's locations in terms of an arbitrary (but fixed) *world* reference frame, with the Z axis pointing downwards (in the direction of gravity). We used a simple initial calibration procedure (described below) to find the angle of the rotation around the Z axis that brings the world reference frame to the frame used to define the floor plan (the *floor plan* frame).

Azimuth/Steps (A/S). This algorithm produces, at each detected step (heel strike), a 2-D "step vector" Δ_p with a length equal to the estimated step length, and direction given by the phone's *azimuth* (or *heading*) angle. This vector is then fed as input to the Particle Filter (discussed later in this section). Heel strikes (steps) are computed by an LSTM-based algorithm that processes data from the phone's inertial sensors [16, 50]. Each participant's step length was estimated by the following calibration procedure. The participant was asked to walk along a straight corridor path (38.25 meters in length) while the step counter kept track of the number of steps taken. By dividing the length of this path by the number of steps, we obtained the user's approximate step length. A similar step length calibration was used in [4, 53]. Note that after calibration, the step length is further updated by the Particle Filter (in a similar fashion to [4]).

Despite its simplicity, A/S was shown to produce good path reconstruction results when coupled with a Particle Filter, which can mitigate the drift effect inherent in the dead-reckoning nature of the algorithm [50]. One drawback of this approach is that the orientation of the step vector Δ_p (i.e., the azimuth angle) is computed with respect to the phone's reference frame, rather than the walker's. This means that moving the phone to a different location on one's body (e.g., pulling it out of the pocket to take a call) may incorrectly be interpreted as a change in walking direction. In our experiment, participants kept the phone tucked inside their back pocket throughout the trials, hence the orientation of the phone with respect to the user's body could be considered to be constant.

RoNIN. RoNIN [29] is a machine learning algorithm for dead-reckoning from inertial data. Its feasibility for the reconstruction of paths taken by blind walkers, using a long cane or a dog guide, was demonstrated in [50] on the WeAllWalk data set [22]. RoNIN produces velocity vectors at a rate of 25 Hz. We used the authors' open-source ResNet18 implementation[1], and integrated the velocity vectors over individual step periods to obtain step vectors Δ_p.

Unlike the A/S algorithm, the step vectors produced by RoNIN are defined with respect to the world frame (rather than the phone's frame). This means that the heading direction estimated by RoNIN is independent of the phone's orientation with respect to one's body [29]. Hence, the user is not constrained to hold the phone in a fixed location. However, RoNIN is as liable as A/S to orientation drift due to the integration of noisy sensor data.

Experiments reported in [50] showed that while RoNIN worked remarkably well for some individuals, it produced less satisfactory results for others. In particular, the magnitude of the velocity vector returned in output for certain users was found to be either smaller or larger than their actual speed. To account for this user-dependent error, we employed the same calibration procedure described above to find an adjustment coefficient (*RoNIN multiplier*) for each participant. In practice, we divided the length of the path traversed during calibration by the estimation of the same path length produced by RoNIN. Then, during the trials with the same participants, we multiplied the velocity vectors produced by the RoNIN multiplier thus computed.

Particle Filtering (PF). Particle Filtering is a form of Bayesian filtering that is commonly used for spatial tracking in the presence of prior information or constraints [23]. In our case, knowledge of the floor plan allows us to define "impenetrable walls" that are unlikely to be traversed by a traveler [76]. Using a Particle Filter, the posterior distribution of the walker's location is expressed by means of a set of samples (*particles*). In our experiments, we used 500 particles, which was found

[1]https://github.com/Sachini/ronin

to be a reasonable trade-off between computational speed and accuracy of tracking (compare e.g. with [51], where 750 particles were employed).

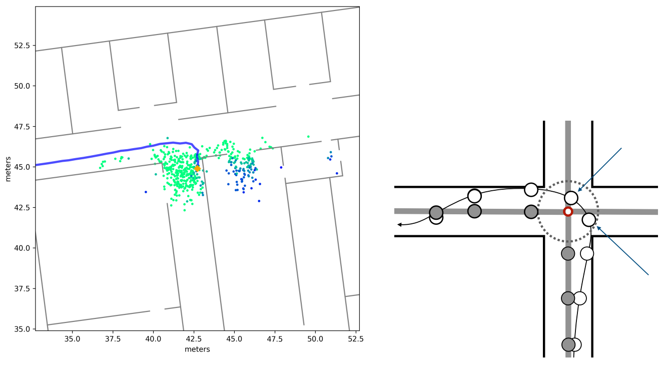
Each i-th particle is characterized by an X-Y location p_i, a drift angle value $\Delta_{\theta,i}$, and, for A/S localization, a step length value s_i. At the beginning of a trial, all particles are located at the (known) initial user location. The drift angles associated with the particles are sampled from a zero-mean Gaussian distribution with $\sigma = 30°$, while the step lengths are sampled from a Gaussian distribution with $\sigma = 6$ cm centered at the step length found during the initial calibration. Each particle has a (positive) weight w_i. Weights are all initialized to $1/500$.

At each time period (in our case, at each detected step), each particle is spatially propagated by a vector that is a function of the step vector Δ_p produced by A/S or by RoNIN. Specifically, Δ_p is first rotated by the particle's drift angle $\Delta_{\theta,i}$, and, for A/S localization, its length is changed to s_i. Gaussian noise is then added to the resulting step vector and to the drift angle $\Delta_{\theta,i}$. The particle's location is then updated by adding the resulting vector to p_i. Additionally, the particle weights can be modified during a positional update as described later in this section (after re-weighting, the weights are normalized to sum to 1.)

At each time step, a new set of 500 particles is resampled with replacement from the current set of particles, with the probability of a particle being sampled equal to its weight. Gaussian noise (σ=10 cm) is added to the X and Y components of each particle's location. The user's location is taken to be the weighted sum of the particles' locations: $p = \sum_i w_i p_i$.

Particles are re-weighted at each time period according to the following rules:

(1) If a particle is found to be crossing a wall, its weight is set to 0.
(2) If a particle is at a distance from the weighted average p larger than a threshold value D, its weight is set to 0.
(3) If at any time a particle is found to be inside a room, its weight is multiplied by 0.9.



(a) (b)

Fig. 1. (a): Blue line: Path from participant P5 measured with A/S. The particle cloud is shown with colors ranging from green (high weight) to blue (low weight). Note that while the bulk of the particles followed the actual path turning into a corridor, some particles entered nearby rooms through their open doors. (b): An illustration of route segment assignment. Four route segments meet at a junction (red circle). The walker's path is shown with white circles, while its projection to the assigned route path is shown with grey circles. When the walker is within a circle with radius T around the junction (a situation shown with two arrows), no segment assignment is made.

Provision (1) reduces the risk of trajectories going through impenetrable walls. Compactness of the particle set is enforced by (2). This is particularly useful for large open spaces, including long corridors, where the particle set could otherwise expand boundlessly. The threshold values D for provision (2) were set (based on trial-and-error experiments) to 5.5 m for A/S localization and to 3.5 m for RoNIN. Provision (3) reduces the likelihood that a user walking in a corridor is mistakenly localized inside a room. It is important to note that all rooms in the building are considered "open" (see Fig. 1 (a)). Even though the selected routes are defined on corridors, walkers could mistakenly enter a room through an open door (as it happened with participant P7; see Fig. 9 (d)), and should be tracked in those spaces as well. However, we noted that without provision (3), sizeable amounts of particles would often enter through the open doors (and get temporarily stuck) in one or more rooms when a participant was walking on a corridor nearby (Fig. 1 (a)). By reducing their weights, we decrease the likelihood that particles in these isolated clusters be selected for resampling.

Resampling also ensures that only angular drift values (and, for A/S, step lengths) that lead to "legitimate" trajectories are preserved. Explicitly modeling the drift angle was shown in [50] to effectively mitigate the effect of accumulating angular drift, a well-known issue associated with dead-reckoning. For A/S localization, we also found that, through resampling, the particles' step length values tended to coalesce towards a value that was often different from that measured during initial calibration, as discussed later in Sec. 4.1.

In our iPhone implementation, Particle Filtering updates are performed each time a footstep is detected. An update normally requires between 0.01 to 0.02 seconds to complete.

Initial Orientation Calibration. For each trial in our experiment, we assumed that participants would start from a known location, and that they initially walked along a known direction. After 6 steps, we found the angle between the reconstructed trajectory (defined, as mentioned earlier, with respect to an arbitrary world reference frame) and the known walking direction, as defined in the floor plan frame.

3.1.2 *Routes and Waypoints.*

The walkable area in a building characterized by a network of corridors can be represented by a set of *waypoints* (located at the corridors' junctions) and *route segments*, which join any two waypoints if there is a traversable straight path between the two. A *route* joining any two waypoints is a sequence of interconnected route segments.

At each time t, the localization algorithm (whether A/S or RoNIN, followed by Particle Filtering) produces an estimated location $p(t)$ of the user. Based on this location, the app issues notifications to the user as appropriate. However, rather than directly using the 2-D locations p, we consider the *projected* locations $\bar{p}(t)$ onto its associated route segments. This approach is justified by the nature of typical buildings with networks of corridors, and simplifies the logic used to produce notifications, which is described in Sec. 3.3.1.

When users walk on a long corridor, far from junctions with other corridors, projecting their location onto the route segment associated with that corridor is a safe operation. More care must be taken when in the proximity of a junction, due to potential localization errors leading to incorrect assignments. For example, as shown in Fig. 1 (b) (see locations marked by blue arrows), associating the user's location with the closest route segment may lead to incorrect results as soon as the distance of the user to the junction is comparable with the radius of localization uncertainty. To deal with such situations, we adopted the following mechanism. A walker whose position is currently associated with a certain route segment maintains this association until the projected location $\bar{p}$ is closer than a threshold distance T to a junction with other route segments (T was set to 1.5 m in our experiment). At that point, route segment association becomes ambiguous, and p is no longer associated with any segment. Segment association is resumed when the projection of p onto any of the route segments ending at that junction is at a distance of T (or larger) from the junction. From that point on, the walker is associated with this new segment (see Fig. 1 (b)).

Our wayfinding app determines the shortest route to a destination using the iOS GameplayKit toolkit[2]. Routes are constantly updated as the user moves, and re-routing is computed as necessary. In addition to waypoints, we defined a number of "landmarks", whose presence was communicated to the walker when nearby. These landmarks, whose location is shown in Figs. 7– 9 (a), are listed in Tab. 3.

3.2 Backtracking

Our Backtracking app is designed to assist a walker who, after traversing a certain route (on their own or accompanied by a sighted guide), wants to walk back on the same route, in the reverse direction, towards the starting point. No knowledge of the building's map is assumed. As mentioned earlier, our Backtracking app is structured in two distinct phases. In the *way-in* phase, the app is in charge of recording the original route. During the *return* phase, the app is in charge of tracking the path of the walker and matching it with the way-in path. In particular, the system needs to determine when the walker is getting close to a location where a turn was taken during the way-in, in which case it can produce specific notifications. The app also needs to recognize if the user missed a turn and provide appropriate remedial directions when necessary.

3.2.1 Path Reconstruction: Way-in. Unlike the Wayfinding app, impenetrable wall constraints cannot be used to reduce orientation drift, because the Backtracking app has no knowledge of the floor map of the building. In buildings characterized by networks of corridors, however, it is conceivable that walkers would proceed along relatively straight paths until they turn at a corridor junction. The angle made by two intersecting corridors, for typical buildings, is often equal to $\pm 90^\circ$ or to a multiple of 45°. This geometric structural constraint can be leveraged to sidestep orientation drift. Following [21, 69], we represent both way-in and return paths as a sequence of straight segments interleaved with discrete angle turns.

We use the robust turn detector described in [50], which processes azimuth information using a Mixture Kalman Filter [12]. Although this algorithm was shown to work well even for multiples of 45° turns, we constrained detection to multiples of 90° for the purpose of this experiment (this reflects the type of junctions found in the building considered for our tests, which were all of $\pm 90^\circ$). The way-in path can be depicted as a 2-D polygonal chain (polyline), where the length of each segment is equal to the number of steps taken in that segment, and consecutive segments have an angle as measured by the turn detector (see e.g. Fig. 3). Steps are detected using the same LSTM network used for A/S localization.

3.2.2 Path Matching: Return. During return, the user is assumed to start from the endpoint of the way-in path, and walk the same path in the reverse direction. The goal of this module is to identify the location in the way-in path that best matches the current location of the user, such that appropriate directions can be provided. Our strategy for matching the return path with the way-in path is based on the coordination of two different algorithms: *projected return sequence* and *sequence alignment.*

Projected Return Sequence. This algorithm reconstructs the return trajectory as a polyline, as described above for the way-in path. By comparing the current polyline reconstructed during return with the polyline built during the way-in, it is possible, in principle, to find the best match in the way-in trajectory to the current user location. For example, one could find the closest point in the way-in polyline to the current user location (the current end point of the polyline built during return).

[2]https://developer.apple.com/documentation/gameplaykit

This simple algorithm works reasonably well in ideal conditions but fails if (1) the walker's step length is different between way-in and return (leading to a different number of steps for the same segment; see Fig. 3 (a)); or (2) the turn detector fails to recognize a turn (e.g., because taken too slowly) or produces a false positive (due to an irregular trajectory); or (3) the walker does not follow exactly the same route (in reverse) as during way-in, e.g. because they missed a turn then turned around. All of these situations may be expected, especially when someone walks without visual feedback.

Sequence Alignment. As described in [52, 69], one may cast the path matching problem as one of (sub)sequence alignment. Assume that the sequence of way-in measurements has been reversed, which is convenient since the route is being backtracked. At each time during return, we determine the initial way-in subsequence of measurements that best matches the current sequence of return measurements. In symbols: given the (reversed) way-in sequence W of measurements (observations) $o_w(t)$ (i.e., $W = (o_w(1), \dots, o_w(N))$), and the current sequence R of return measurements $o_r(t)$ (i.e., $R = (o_r(1), \dots, o_r(J))$), the goal is to find a sequence of indices $i_1, \dots, i_J$ such that $(o_w(i_1), \dots, o_w(i_J))$ best matches R under an appropriate criterion. For real-time guidance, we are interested in the last matching point i_J: we will assume that the walker at the current return time index J is in the same location they were at time index i_J during way-in. Standard dynamic programming approaches (e.g. Dynamic Time Warping [40, 52]) can then be used to find an optimal match.

The measurements we consider are (1) magnetic field vectors and (2) turns detected. Step detection is also considered implicitly: for both way-in and return, the sequences of time indices are defined such that there are three regularly spaced time intervals between two consecutive detected steps. We found that this choice gives enough spatial granularity for magnetic field matching, while ensuring parsimonious sampling (e.g., no samples are recorded when the user is stationary).

It is well known that the magnetic field recorded in different positions within a building is not uniform, due to reasons such as the presence of large metallic objects and magnetic field generating appliances. "Magnetic signatures" can thus be attached to specific locations, enabling sophisticated localization mechanisms [10, 37, 61, 64] (e.g., IndoorAtlas' magnetic positioning[3] [63].) The magnetic field at a certain location can be measured by a 3-axis magnetometer, of the type embedded in any modern smartphone. (Prior calibration of the magnetometers is necessary for good results.) From the measured 3-D magnetic field vector we derive a 2-D vector that is invariant to the orientation of the phone as described in [18, 39].

Given the measurements, one can create a directed graph $\mathcal{G}$ with nodes indexed as (i, j), where i is a way-in time index, and j is a return time index. (Note that the $\mathcal{G}$ is constantly expanded as the walker progresses along the return path.) A node (i, j) in the graph has only three edges, to $(i + 1, j)$, $(i + 1, j + 1)$, and $(i, j + 1)$, respectively. This is consistent with the assumption that the walker is normally moving in the same direction as in the (reversed) way-in path, but possibly with a different step length (resulting in steps detected in either phase that cannot be matched in the other phase, which are accounted for by the edges to $(i + 1, j)$ and $(i, j + 1)$). The edges to $(i + 1, j)$ and $(i, j + 1)$ carry a non-null edge cost, while node costs are defined as a function of the magnetic field vector discrepancy (measured as the Euclidean difference between the recorded vectors) between the measurements $o_w(i)$ and $o_r(j)$. An example of magnetic costs assigned to the nodes of a graph is shown later in Fig. 13 (b). Sequence alignment is obtained by finding the minimum cost path starting from $(0, 0)$ and arriving at a node (i, J).

Unfortunately, this simple approach, originally proposed in [52], did not produce satisfactory results in our preliminary tests, especially if the return path is not perfectly identical to the way-in

[3]https://www.indooratlas.com

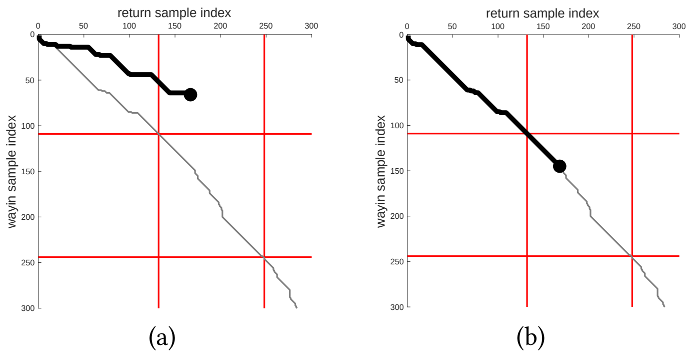


Fig. 2. Examples of optimal paths in the graph $\mathcal{G}$. Both plots are for the same trial. The thin gray line shows the optimal graph path after all return data was available (at the end of the trial). The thick black line is the optimal path computed from return data up to $t = 167$ steps (a) and up $t = 168$ steps (b). Note that the graph path reconstructed in the first case was incorrect, which could potentially result in incorrect or inconsistent notifications given to the user. The red lines represent times at which a turn was detected during way-in (horizontal lines) and return (vertical lines)

path. Indeed, we found that, in some cases, the magnetic field can vary rather dramatically when moving across the width of a building corridor[65]. An improvement to pure magnetic-based alignment can be obtained by considering the turns taken by the walker, which, in an ideal case, should be matched between way-in and return. We use the mechanism described in [69], which considers the difference in walking direction between way-in and return by structuring the graph as a sequence of layered planar graphs, where each layer represents a possible orientation discrepancy between way-in and return.

The minimum cost path in the graph $\mathcal{G}$ is recomputed at each new return sample. We use the incremental Dynamic Time Warping (iDTW) proposed by Riehle et al. [52], which uses a sliding window defined around the endpoint of the previously found optimal path (we set the window size equal to 200 samples). Although this algorithm produces a suboptimal solution, it represents a good compromise between precision and computational cost.

Ultimately, this algorithm produces, at each time index in the return sequence, the best matching time index in the way-in sequence on the basis of the *whole* sequence of measurements available. While this approach provides robust sequence-to-sequence matching, it is important to note the best matching sequence can change when more data is available (i.e., as the user continues to walk along the return path). More precisely: assume that the way-in time sequence $i_1, \ldots, i_J$ is the best match (i.e., a minimum-cost path in $\mathcal{G}$) for the return time sequence $1, \ldots, J$. It is possible that the best way-in time matching sequence to $1, \ldots, J, J+1$ *may not* contain the sequence $i_1, \ldots, i_J$, that is, the minimum cost path in the graph may change once the graph is updated to include the new measurement $o_r(J+1)$. An example of this phenomenon is shown in Fig. 2.

In practice, this means that if the algorithm determines that, at return time J, the best way-in match is i_J, later on it may discover that this match was incorrect. This can be problematic, as shown by a simple example. Suppose that at time i_J, the way-in path (in reverse) was close to a turn. The system may notify the user to prepare for the incoming turn. At a later time, the estimated user location has changed, and the system may end up notifying the user to walk straight instead. This “jittery” localization may thus cause instability when producing guidance notifications. In the following, we describe our strategy for mitigating the effect of unstable localization.

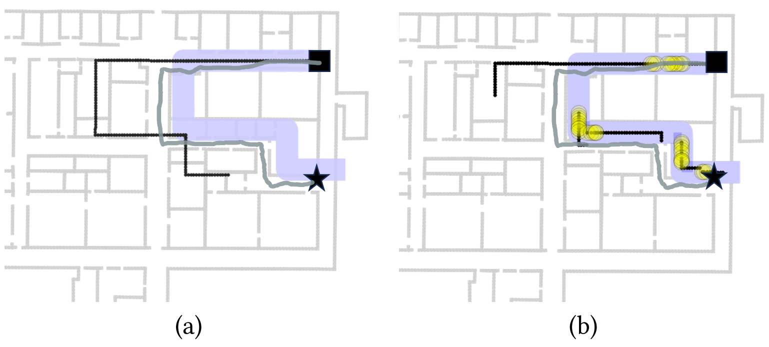


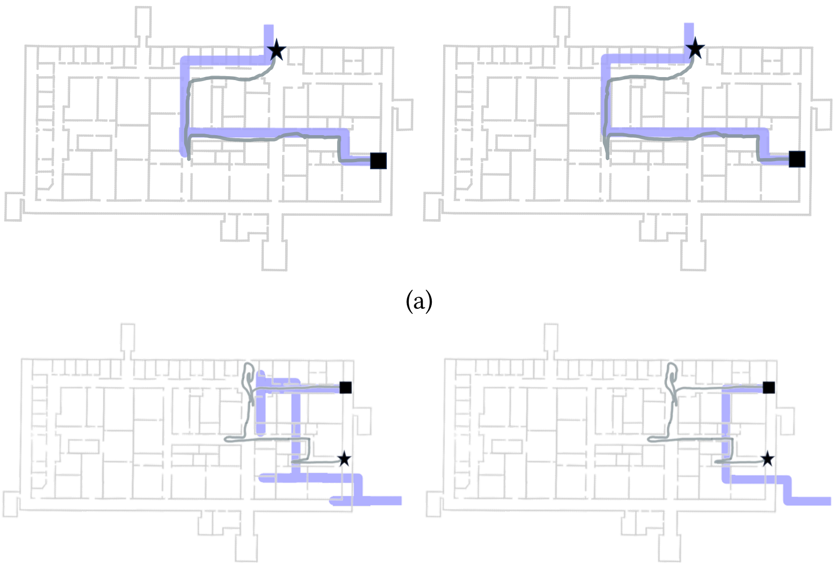
Fig. 3. Examples of return path matching using *projected sequence* (a) and *hybrid matching* (b). The way-in path is shown with a thick purple line ending at the black square. The length of each segment is given by the number of steps recorded, multiplied by the step length measured during calibration. The approximate path of the participant during the return phase, obtained by visual inspection, is shown by a gray line. Reconstructed return paths are shown with a black line. Note that in (a) (*projected return sequence*), the initial return segment (originating from the black square) appears to be longer than during way-in, possibly because the walker took shorter steps during return or because they took additional steps while looking for a place where to turn. As a consequence, the two reconstructed polylines (way-in and return) are quite different from each other. Finding the location in the way-in path matching the current user location at any given time during return could be challenging and error-prone. In (b) (*hybrid matching*), reliable matches are shown as yellow circles. Even though occasionally the reconstructed return path is different from the way-in path, the return path is corrected as soon as a new reliable match is found.

Hybrid Matching Strategy. In order to reduce the risk of inconsistent notifications produced as an effect of unstable matching as discussed above, we propose a simple solution, based on the notion of *last reliable match* . We use local properties of the current minimum cost graph path to decide, at the current time J, whether the match (i_J, J) can be considered "reliable", meaning that it is likely to be preserved even after later observations are recorded. In practice, we look at the terminal part (the last 21 samples) of the minimum cost path in the graph ending at (i_J, J), and judge this last match to be reliable if this path segment is well approximated by a line with unitary slope.

When a reliable match is detected, a new sequence is initialized at that match. As more measurements are taken, the return path is updated using the *projected return sequence* method described above until a new reliable match is found, at which point the return path reconstructed is again reinitialized. Guidance notifications are produced based on the walker's location identified using this projected sequence. As the example of Fig. 3 (b) shows, re-initialization at each detected reliable match helps "reset" the reconstructed return path, while the use of projected sequences when matches are not reliable minimizes the risk of inconsistent notifications being issued.

3.2.3 Way-in Path Simplification. In some cases, the way-in path contains redundant turns or loops. This may happen when, for example, the walker took a short detour (perhaps because they were unclear about the route to take) or if they followed a zig-zag course instead of walking on a straight line. It would be desirable to remove these redundant features before backtracking. In our experiment, we implemented a very simple "path simplification" algorithm for the way-in paths. In short, we define a conservative radius of uncertainty D equal to 7 steps. We then merge together nearby parallel edges that are at a distance of D or less from each other. We do the same with turn locations (vertices). Then, we re-create a simplified path as a shortest length polyline that goes through the remaining edges and vertices.

An example of successful way-in path simplification is shown in Fig. 4 (a). Note that the original path (shown to the left) contains a short detour with a 180° turn, which was due to the walker originally missing a turn. This spurious piece was correctly removed by the simplification algorithm. Fig. 4 (b) shows a case with a complex way-in path, containing multiple loops. Although our algorithm was able to remove these loops, and the resulting simplified path maintained the correct path geometry, its first segment turned out to be substantially shorter than in the original path, which led to an unsuccessful backtracking trial.


(a)

(b)

Fig. 4. Examples of successful (a) and unsuccessful (b) way-in path optimization. The original reconstructed way-in path is shown in the left panels with a thick purple line, ending at the black square, along with the approximate actual path taken by the walker (as measured from the video), shown with a gray line. The optimized way-in paths are shown in the right panels.

3.3 User Interface

In the following, we list the general principles that guided our interface design for the two apps.

1. Consistency. Even though the two apps utilize different localization technologies, the user interface is almost identical between the two. Both apps ultimately provide directions toward a destination in similar environments (networks of corridors); thus, it is only natural that they should afford similar user experiences. The small differences (highlighted in Sec. 3.3.1) are a consequence of the different prior knowledge the apps can rely on.

2. Robustness to localization inaccuracy. The localization system of both apps relies on dead reckoning from a smartphone's inertial sensors. In spite of mitigation techniques (Particle Filtering for Wayfinding, sequence alignment for Backtracking), localization errors of 2-3 meters should be expected. Neglecting these potential errors when issuing notifications may lead to catastrophic results. For example, suppose that the app issues a notification shortly before it locates the walker at a junction. As shown in Fig. 5 (a), the walker's real position may happen to be at some distance before or after the junction. In either case, if the walker were to turn, they would face a wall. What's worse, they would have no way to know whether to search for the junction to their right or to their left.

To account for the expected localization inaccuracy, we made the following design choice: the presence of a junction where a turn should be taken is announced with substantial advance notice, enough to ensure that, with a high likelihood, the walker has not yet reached the junction. Although conceptually very simple, it was not obvious *a priori* that this approach would be viable and acceptable. The notification is given at a variable distance from the junction (depending on the current localization error). Users are thus in charge of searching for the next available opening upon hearing the notification, which involves active exploration with the long cane or with the dog guide. An important goal of our experiments was to verify whether our participants would be successful in this task and, importantly, whether they would find this strategy acceptable.

Based on preliminary tests, we decided that a notification should be issued when the walker is located by the app at 7 meters from the next waypoint (for the Wayfinding app) or 14 steps from the next turn point in the reversed way-in path (for the Backtracking app). While this distance may seem large, it is important to observe that our participants often kept walking while listening to the notification; by the time the notification was completed and they processed it, they may have already advanced by 2-3 meters.

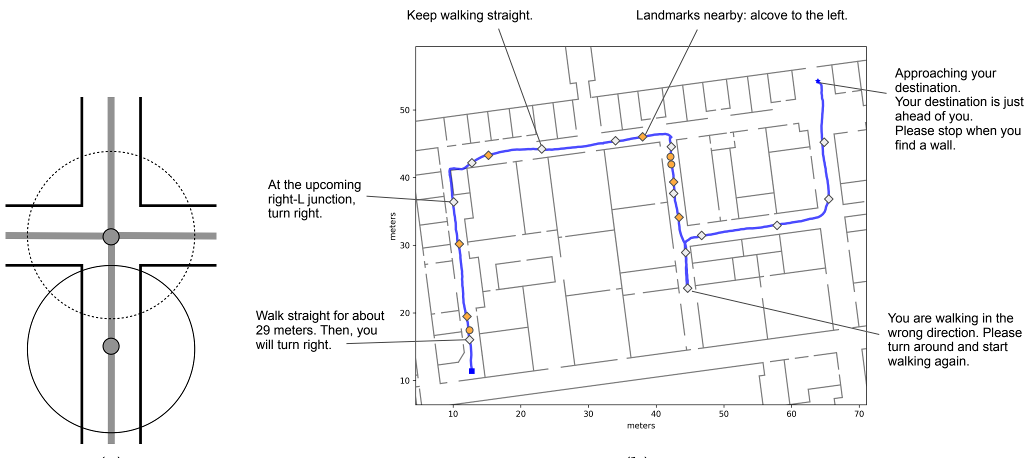

Keep walking straight.
Landmarks nearby: alcove to the left.
Approaching your destination. Your destination is just ahead of you. Please stop when you find a wall.
At the upcoming right-L junction, turn right.
Walk straight for about 29 meters. Then, you will turn right.
You are walking in the wrong direction. Please turn around and start walking again.
meters
meters
(a)
(b)

Fig. 5. (a): A hypothetical junction, with the walker's location (as estimated by the app) shown by either gray-filled circle. The larger circles represented the expected radius of location uncertainty. If the walker is located by the app at the junction center (top circle), their actual location can be anywhere in the dashed circle of uncertainty (including before or after the junction). If the app locates the walker at the lower circle, the actual location of the walker is certainly before the junction. (b): Examples of notifications produced during route traversal (participant P5, route R1W). Directional notifications are shown as symbols filled in gray, while landmark notifications are shown as filled in orange. For the symbols shown as diamonds, an actual notification was produced. For the symbols shown as circles, a notification was not produced as it would have interrupted an ongoing notification. For a few selected notifications, the content of the speech produced is also shown.

3. Minimal disruption. Walking without visual feedback in an unknown space requires a good deal of concentration. It is important that the notifications generated during walking be designed to minimize the required cognitive load for processing, and that they are not distracting. Our apps issue notifications in the form of short synthetic speech sentences (Sec. 3.3.1). For the most time-critical notifications (e.g., announcing an upcoming turn), the notification is preceded by a short chime and accompanied by a short vibration on the Watch. The user can, at any time, have the last notification repeated or hear a description of the remaining route. In addition, walkers heard a short sound at each footstep. Originally implemented for debugging purposes, we decided to leave this feature in the apps, giving each participant the option to turn it off. We note that other types of sound-based or vibration-based interface could be utilized, including those mentioned in Sec. 2, and we will consider those in future work.

4. Watch-based control. In our experiment, we asked our participants to keep the phone in a pocket throughout the trials. We also asked our participants to wear an Apple Watch, that they used to control the app. These are the gestures considered for the Watch:

- *Before the beginning of a trial:* Participants were asked to select a specific route (e.g., "Path number 2") from a list. The list could be traversed in both directions through left and right swipes on the Watch's face, with the name of the current item in the list read by VoiceOver.
- *To start the app:* Participants were asked to start either app by rotating the Watch's crown and keep rotating (in either direction) until they heard a "ding" sound from the Watch, signifying that the app had started. At that time, participants heard the notification "Please start walking."
- *To hear the last notification again:* At any time during a trial, participants could do a right swipe on the Watch's face to hear the last notification issued by the app.
- *To hear a route description:* At any time during a trial, participants could do a left swipe on the Watch's face to hear a description (in terms of route segments and turns) of the remaining route, from their current location till destination.
- *To stop the app:* Upon arrival at destination, or if the trial was aborted, participants were asked to again rotate the Watch's crown to stop the app.

3.3.1 Notifications. Notifications are produced based on the current route and the location of the user. More specifically, we consider the distance of the user's location (projected on the associated route segment for the Wayfinding app, or on the associated way-in path segment for the Backtraking app) to the next waypoint in the route or to nearby landmarks (Wayfinding), or to the next turn in the reversed way-in (Backtracking).

Our apps support up to four distinct types of notifications. Note that, in case of conflict (a notification triggered while a prior notification was being delivered), the ongoing notification is never interrupted, except in case the new notification is of type (1) as listed below (alerting the user of an upcoming turn). This type of notification is considered time-critical, and therefore takes priority. Also note that the same notification is never repeated, except for notifications of type (3).

(1) *When the next waypoint (Wayfinding) or turn (Backtracking) is at less than 7 meters.* If a turn should be taken at that waypoint, the notification is: "At the upcoming [X, left/right L,T] junction, turn [left, right]." Note that the junction morphology (X, L, T) requires access to a floor plan, and for this reason, this detail is not announced by the Backtracking app.
If the walker is not supposed to turn at the next waypoint, the message is simply: "Keep walking straight." This last notification is meant to reduce the risk that the walker, having found a junction, may mistakenly turn there. Note that this information is not available for the Backtracking app, which is only aware of the turns taken during the way-in (and thus is unaware of the presence of a junction, if the user did not take a turn at the junction during way-in).

If the next waypoint is the destination (final) waypoint, or the end of the way-in route, the notification is: "Approaching your destination, Your destination is just ahead of you". If the route ends at a wall or at a closed door, this notification is followed by: "Please stop when you find a wall".

(2) *Upon entering a new route segment (Wayfinding) or when the last reliable match is updated to a new segment(Backtracking).* Notification: "Walk straight for about XX [meters/feet/steps]. Then, you will turn [left, right]." The last sentence was removed for the Backtracking app after participant P4, for reasons that will be made clear later. Some route segments may contain protruding obstacles (e.g., a steel cabinet) on either side. In this case, the notification is preceded by "Please keep to the [left/right]".

(3) *When the user has been walking in the wrong direction on a route segment for at least 4.5 meters (Wayfinding), or is at a distance of more than 16 steps from the way-in path (Backtracking).* "You are walking in the wrong direction. Please turn around and start walking again." This length (4.5 m) was chosen through trial and error in initial experiments and is consistent with the expected radius of uncertainty of localization. This notification is repeated if they continue walking in the wrong direction for another 4.5 meters.

(4) *When the next landmark is at less than 2 meters (Wayfinding only).* "Landmark nearby. [Landmark name] to the [left/right]." Note that a shorter threshold distance than for approaching junctions is used here. This is justified by the fact that, while a junction needs to be announced *before* the walker has reached it, advance notice is less critical for announcing nearby "landmarks".

4 EXPERIMENT

4.1 Population

We recruited 7 participants for this experiment (see Fig. 6). The participants' characteristics are summarized in Tab. 1. All participants were blind, with at most some residual light perception. All participants were experienced independent walkers, although P6 had recently switched to using a long cane after many years of walking with a dog guide, and was still re-learning to use the cane. P5 had a hearing impairment and used hearing aids. All participants were iPhone users, except for P7, who used a cell phone with a physical keypad. Only P1 regularly wore a smartwatch (Apple Watch). Tab. 1 also shows the average step length values measured during the initial calibration, as well as the average step length from the particles at the end of the trials (remember that step length was added as a state in the Particle Filter only beginning with P4). Note that the step length measured in the initial calibration was always found to be larger than or equal to that found by the Particle Filter. This suggests that participants may have walked with a different step length during calibration (which was conducted in a "safe", straight corridor stretch) and during the actual test, where they might have felt less confident and therefore walked with shorter steps.

4.2 Procedure

4.2.1 Environment. The trials were conducted on the second floor of a campus building. In order to test both the Wayfinding and the Backtracking apps, we designed the following procedure. First, a participant would use the Wayfinding app to traverse three routes, where the beginning of each route coincided with the end of the previous route. During this traversal, the Backtracking app (running on a different iPhone, carried by the participant in a different pocket than the iPhone running the Wayfinding app) recorded measurements (magnetic field, steps, turns) from each route. Effectively, these routes were considered as way-in routes by the Backtracking app. At the end of the third route, the participant was asked to re-trace each route (in reverse order, starting from the

Table 1. Characteristics of the participants in our study. For blindness onset, 'B' indicates 'since birth', while 'L' indicates 'later in life'. Step length (final) represents the average step length value over all particles at the end of the trials (averaged over the three Wayfinding trials). Note that the Particle Filters included step length as a state only beginning from the trials with P4.

	Gender	Age	Blindness onset	Mobility aid	Step length (cm) (calibration)	Step length (cm) (final)	RoNIN multiplier	Preferred units
P1	F	73	L	Dog	48	–	0.96	Steps
P2	M	69	B	Cane	51	–	1.08	Feet
P3	M	53	B	Cane	54	–	1.14	Feet
P4	F	69	B	Cane	51	44	1.0	Feet
P5	M	75	L	Cane	44	41	1.21	Meters
P6	F	76	L	Cane	40	40	1.11	Steps
P7	F	72	L	Dog	63	58	1.08	Feet

last one) in the opposite direction. During this phase, the participant would receive notifications from the Backtracking app, now set to the "return" phase. As discussed earlier, we implemented two location tracking algorithms for Wayfinding (A/S and RoNIN). By default, Wayfinding uses location data from A/S to produce navigation direction, with the possibility of switching to RoNIN in case of failure of A/S. This occurred only once, as discussed later. However, we ran both algorithms in parallel and recorded location data from both for later comparison (Sec. 4.3.1).

Three chosen routes are shown in Figs. 7–9 (a). When testing the Wayfinding app, participants first traversed route R1W (Fig. 7 (a); length: 123 m), then route R2W (Fig. 8 (a); length: 97 m), and finally route R3W (Fig. 9 (a); length: 72 m). Note that the endpoints of these routes were at the location of closed exit doors in the building. R1W traversed one L-junction, two T-junctions, and two X-junctions, and included 4 turns. R2W contained two L-junctions, two T-junctions, and two X-junctions, with 5 turns. R3W contained two L-junctions and two X-junctions, with 4 turns. Some corridors were wide (6 meters in width), while other were narrow (1.9 m). Multiple types of obstacles were present, typically located near the walls, in different places: printers, lab carts loaded with vials, couches, pillars, and an emergency eye-washing fountain protruding from a wall. The building was generally quiet, with students and researchers occasionally encountered in the corridors. As mentioned earlier, all junctions in this building were at 90°. A few challenging situations are shown in Fig. 6, and include a narrow door (always kept open) that had to be traversed (a); multiple alcoves that had to be negotiated (c); a side staircase in a narrow corridor that had to be avoided (e); a wide (10 m × 6 m) open space, partially visible in (f). We should note that large open spaces are notoriously challenging for blind travelers, due to the lack of tactile or aural references nearby.

When testing the Backtracking app, the same routes were traversed in the opposite direction and in the reverse order (these routes are called R1B, R2B, and R3B). So in practice, each participant first walked R1W, R2W, and R3W, using the Wayfinding app, while the Backtracking app (in a different smartphone) collected way-in data for each route. Then, they tested the Backtracking app by traversing R3B, R2B, and R1B. We would like to emphasize that the Backtracking app did *not* have knowledge of the building layout. We should also note that, in a practical scenario, walkers would only use one of the two apps. If a map of the building is available, the Wayfinding app provides guidance to a desired destination. The Backtracking app would only be used for route reversal when a map is not available. We tested both apps in the same study for experimental convenience only.

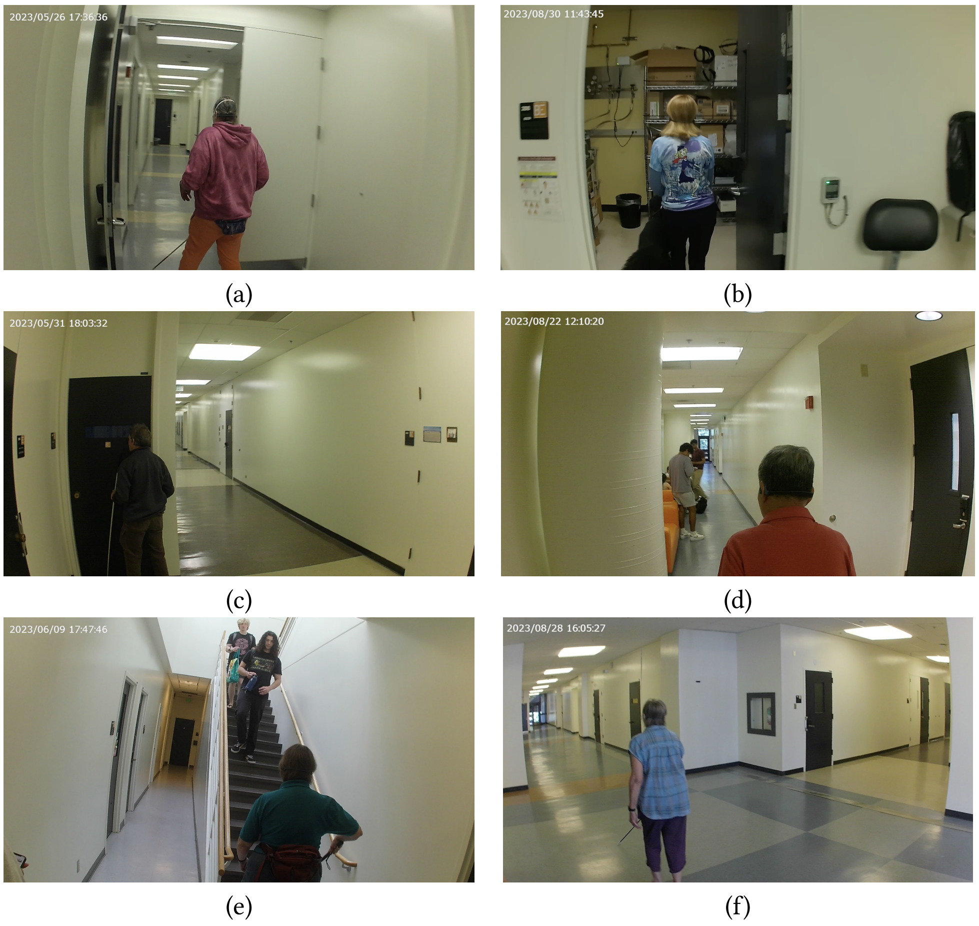


Fig. 6. Pictures of our participants during the trials. (a): P2 negotiating a narrow door in R2W. (b) P7 mistakenly entering a room in R3W. (c) P3 negotiating an alcove in R1W. (d) P5 entering a narrow corridor in R1W. (e) P4 exploring the bottom of a staircase with her cane in R1W. (f) P6 in a large hall in R2W.

A separate building was used for the practice trial when participants were shown how to use the two apps. The practice trial route was simpler, with only 2 turns, for a total length of 63 meters. The two buildings were at a short distance from each other.

4.2.2 Modalities. The experiment was conducted following a human subject protocol approved by our University's Institutional Review Board. After being consented, each participant was explained the purpose of both apps and their functions and encouraged to ask questions if something was not clear. Particular care was taken to impress upon the participants that notifications about upcoming turns would be produced with advance notice. We explained that, upon hearing such a notification, they would need to search for the first place where they could take a turn. This place could be in their close proximity, or a few meters down the way.

After this initial phase, participants underwent the simple calibration procedure described in Sec. 3.1.1. Then, we walked with them to the location where the practice trial would start. Each participant carried two iPhones in their pants pocket (note that we asked the participants prior to the experiment to please wear pants with pockets on the experiment day). Both phones needed to be carried for the initial (Wayfinding) trials: while one of them (an iPhone 12) ran the Wayfinding app, the second one (an iPhone XR) recorded way-in data. Participants also wore a wireless bone conduction headset (Shokz OpenRun), through which they could receive notifications from apps. They also wore an Apple Watch Series 8, used to interact with the apps. Before beginning the practice trial, we made sure that the VoiceOver speed and sound volume were set to the desired level. We asked each participant whether they preferred directions to be given in units of meters, feet, or steps, and set the apps' parameters accordingly (see Tab. 1). Note that for the first three participants (P1–P3), due to an implementation mistake, the Backtracking app produced distances only in units of steps.

We asked the participants to practice making left and right swipes on the Watch. All participants eventually learned how to do this, though P2 had some difficulties at the beginning, as he interpreted "right" or "left" as referring to the axis of his left arm (where he wore the Watch), i.e. in a direction orthogonal to the forearm, rather than parallel to it.

At this point, the practice trial started: the practice route was traversed using the Wayfinding app; once at the destination, it was traversed in the reverse direction using the Backtracking app. After the practice trial, participants were asked whether they wanted to turn off the sound of individual detected footsteps. All participants chose to leave the footstep sound on, with some commenting that hearing it reassured them that the system was functioning. We also asked them whether they wanted to turn off notifications of nearby landmarks (a few landmarks were announced during the practice trial). They all chose to leave landmark notifications on.

Upon completion of the practice trials, the participant and the experimenters moved to the building where the actual experiment was to take place, and in particular to the starting point of the initial route (R1W). From there, the sequence of trials described in Sec. 4.2.1 was started. At the beginning of each trial, the participant was accompanied to the starting location of the route, and their body was oriented along the initial walking direction. Then, they were asked to select the next route from the list using the Watch. Once the route was selected, they were asked to rotate the Watch's crown to start the app. We also asked them to make a left swipe on the Watch, to hear a description of the whole route. Then, the participant started walking the route. Upon arriving at the destination, participants were asked to stop the app by rotating the Watch's crown. They were asked if they wanted to rest a bit before starting the next trial. Then, they were re-positioning to the starting point of the next route (same as the endpoint of the previous route) and oriented correctly, before starting the next route. During the trials, the experimenters followed the participants at a safe distance, taking care not to influence their routing decision.

In the transition between trials with the Wayfinding app and trials with the Backtracking app, participants were handed a new headset and Watch (same models) and asked to substitute the old ones with these new ones. Both devices can be paired with one iPhone at a time; this switch was necessary since the Backtracking app ran on a different iPhone than the Wayfinding app.

At the end of the last trial (route R1B with the Backtracking app), participants and experimenters walked back to the first building, where participants were asked to participate in a questionnaire including the ten System Usability Scale (SUS) questions, as well as a number of open-ended questions.

4.3 Observations

4.3.1 Tracking and Guidance Performance: Wayfinding. Tab. 2 reports the duration of successful route traversals. The average speed (route length divided by traversal time) was 0.50 m/s for all three routes. The first route (R1W) required an intervention from the experimenters for participants P1, P2, and P3. For P1, the localization algorithm used to generate notifications (A/S by default)

Table 2. Summary of the experiment for the Wayfinding and Backtracking routes. For successfully completed routes, we report the duration (in seconds). When displayed with a grey background, the participant missed one or more turns, or took a wrong turn, but was able to walk back and complete the route with guidance from the app. *R*: required system reset. *E*: route was completed, but verbal input from an experimenter was needed at some point. ×: trial had to be aborted due to the app's inability to track the participant. In two cases, a second attempt was made after a trial had been aborted.

	P1	**P2**	**P3**	**P4**	**P5**	**P6**	**P7**	**Length**
R1W	261 (R, E)	355 (R)	297 (R)	216	271	223	206	123 m
R2W	304 (E)	209	134	163	211	262	171	97 m
R3W	125	170	98	127	144	139	330	72 m
R3B	180	×	115	134 (E)	136	×	×	72 m
R2B	182 (E)	232	×	238	173	154	149	97 m
R1B	187	206	149	167	×, 163 (E)	184	×, ×	123 m

Table 3. List of landmarks (see Figs. 7–9).

Landmark ID	Landmark type
1	alcove
2	benches
3	staircase
4	photocopiers
5	alcove
6	alcove
7	pillar
8	couches
9	pillar
10	pillar
11	door
12	cabinets

had to be switched to RoNIN after the first turn because A/S was unable to correctly track the participant (see Fig. 7 (b), red line). For P2 and P3, we had to manually reset the user's location in the app. In all three cases, the remainder of the Wayfinding trials were completed without any issues (except for P1 in R2W, as discussed later). It is important to note that for these three participants, the step length update s_i had yet to be incorporated in our Particle Filter implementation (Sec. 3.1.1), hence A/S relied solely on the step length measured during calibration. The first route segment of R1W is a long (more than 40 meters) and narrow corridor, with several obstacles and a staircase (see Fig. 6 (e)) on its right side (participants were advised not to walk on the staircase). This caused the participants to walk with a substantially smaller stride length than during calibration. By the time they reached the end of the corridor, the accumulated length error was such that, even with Particle Filtering, tracking became exceedingly challenging. For P1, switching to RoNIN (which produced excellent results in this case; Fig. 7 (b), blue line) did the trick. However, RoNIN also occasionally gave poor results in the same area (e.g., it was unable to track P2). After we upgraded the Particle Filtering to include step length values, this problem no longer occurred, and we were able to use A/S successfully for all other participants.

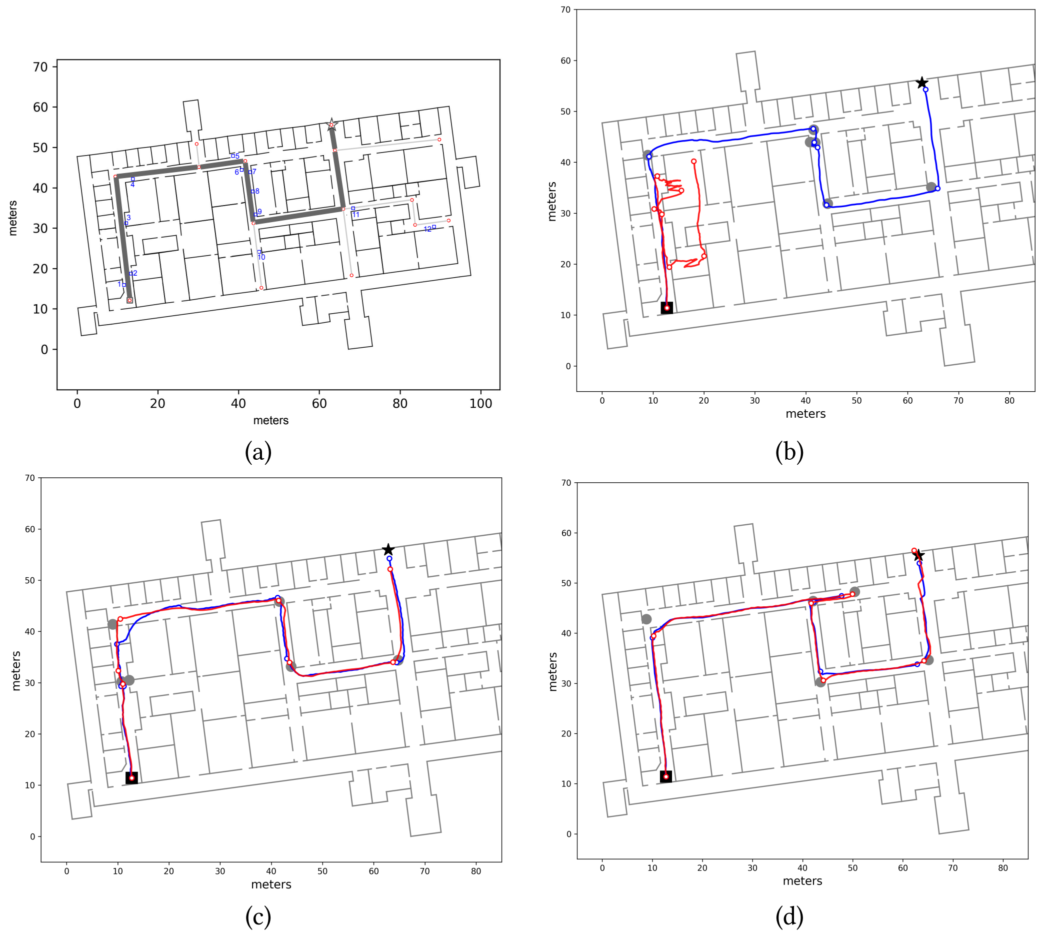


Fig. 7. Route R1W. (a): Floor plan of the considered building. Waypoints are shown in red, traversability graph edges are shown in gray. The start and end waypoints are marked with a square and a star, respectively. The shortest path is shown with a thick dark gray line. Landmarks are shown in blue and enumerated (see Tab. 3 for landmark listing.) (b)–(d): Recorded paths using A/S (red line) and RoNIN (blue line). The grey disks represent "ground truth" locations, while the small colored circles show the estimates by A/S and RoNIN for the same locations (b): P1. (c): P4. (d): P7.

Tab. 2 also shows (cells filled in gray) situations in which participants successfully completed the trial but occasionally had to trace back their path (as prompted by the app) because they had missed a turn. These situations are clearly visible in Fig. 7 (d), Fig. 8 (c) and (d), and Fig. 9 (c) and (d). Later analysis, using the recorded video and logged data from the app, brought to light the reasons for these missed turns. In some cases, participants were distracted by passers-by (P5 in R1W), or they were talking at the time notification was issued and thus missed it (P2 in R2W; Fig. 8 (c)). P6 missed a turn in R3W while walking in a large open space (Fig. 9 (c)). Probably due to

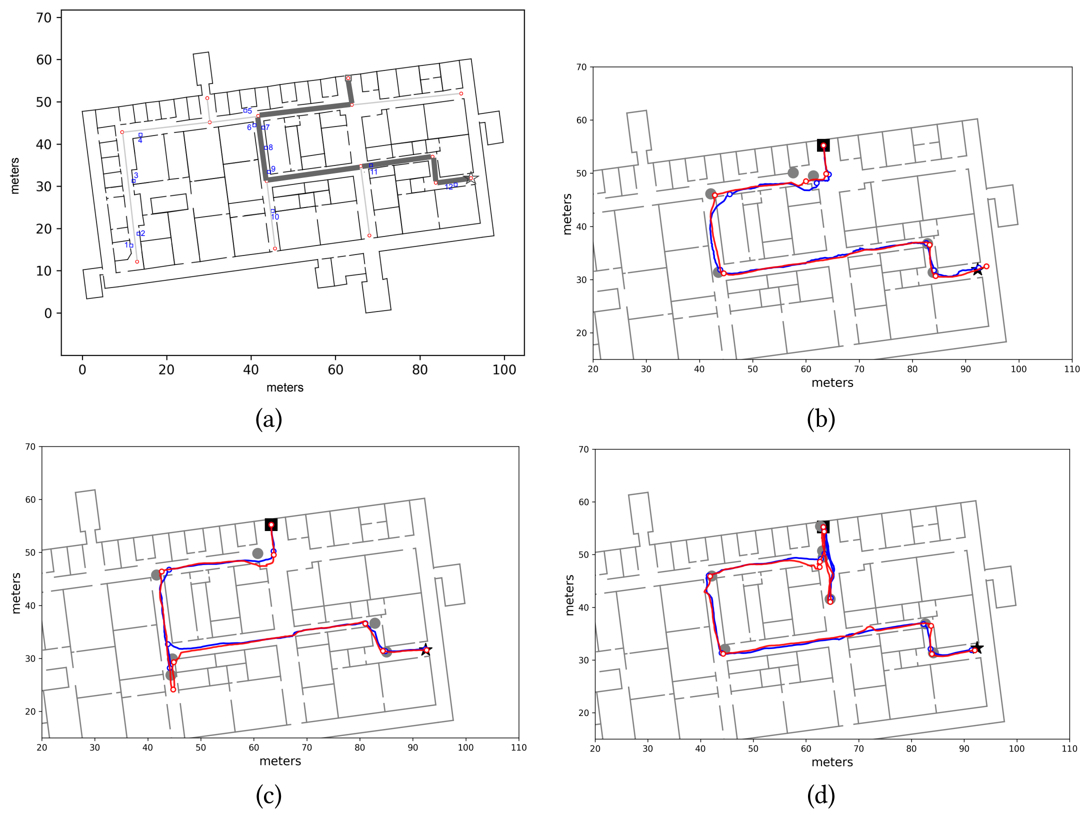


Fig. 8. Route R2W. See caption for Fig. 7. (b): P5. (c): P2. (d): P1.

the lack of a nearby wall, she might have been unsure about where she should turn exactly, so she continued walking until she perceived a wall to the right. By that time, she was notified to turn around. P7, who missed one turn in R1W and one in R2W, as well as multiple turns in R3W, walked very fast with a dog guide. In these situations, she had already walked past the junction by the time the notification was completed and she could process it. Of note, on route R3W, soon after the beginning of the trial, P7 walked straight into a small room whose door was open (Figs. 6 (b) and 9 (d)). She then missed other turns in the same route, one of which multiple times, until eventually she found the correct final route segment (Fig. 9 (d)). Only one time (P6 in R2W) did the app issue a notification shortly after the participant had passed the junction, causing her to miss the turn. This was due to the localization algorithm momentarily undershooting her position. However, after being notified that she should turn around, she successfully negotiated the turn.

Participants using a long cane appeared to generally react proactively to early advance notice of upcoming turns. They typically would begin walking closer to the wall on the side where they were expected to turn, and tap the wall with their cane until they found an opening (Fig. 6 (f)). In some cases, an alcove (wall recess) was located right before the corridor junction, and some participants got "stuck" in this alcove before finding their way out and proceeding to the junction (Fig. 6 (c)).

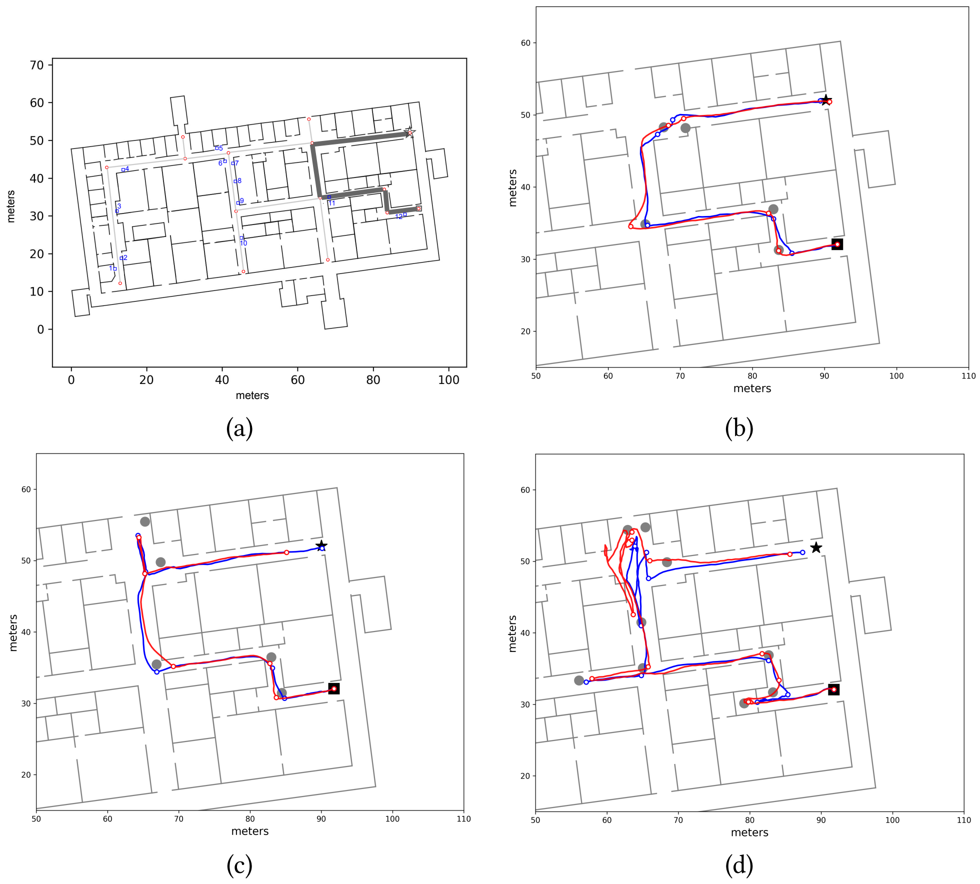
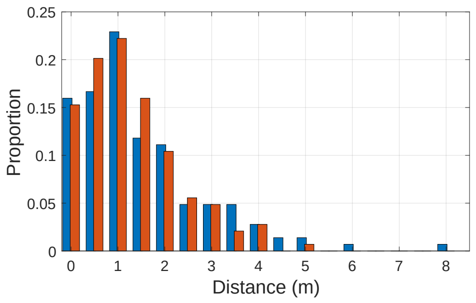


Fig. 9. Route R3W. See caption for Fig. 7. (b): P3. (c): P6. (d): P7. Note that P7 mistakenly entered a room (through an open door) soon after the beginning of the trial.

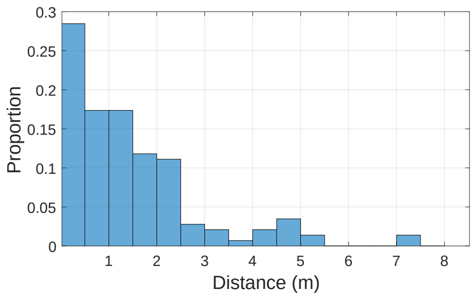
The mobility technique of P2 represented a remarkable exception. Rather than scanning the ground surface in front of himself, P2, a seasoned independent traveler, used the cane to periodically tap the floor surface and made judgments about nearby surfaces based on the sound produced by this tapping. Upon hearing a notification of an upcoming junction, he would move somewhat closer to the corresponding side of the corridor (without tapping the wall) until he perceived the presence of an opening to his left or to his right, at which point he would make a 90° turn into it. However, in one situation in the R3W route, he missed a turn and had to then turn back as guided by the system (he then commented "I knew where that was exactly", indicating that he may have indeed perceived the opening already the first time around.)

The two participants with a dog guide displayed very different behaviors. Since dog guides are normally trained to understand "left" or "right" directions, the participants simply needed (in principle) to convey the direction from the app verbally to their dogs. For the case of P7, this mechanism worked well, except for the fact that, as mentioned earlier, she walked really fast with her dog, thereby often passing by a junction to then receive a "turn around" notification. P1, on the other hand, worked with a slower and more cautious dog. For example, in route R1W, right after the second turn, they got stuck in an alcove to the right and needed help from the experimenter to get out of it (marked as *E* in Tab. 2). It is conceivable that if P1 had been using a cane, she could have searched left and right to find a way out of the alcove. Her dog initially refused to turn into a specific corridor in two other situations. In one such case, at the beginning of R2W (Fig. 8 (d)), she missed the first turn because, by the time she gave her dog a "right" command after the notification, they had already walked past the intersection. Afterwards, they correctly turned around as advised by the app. However, as she issued a "left" command to return to the route, the dog refused to turn and instead walked straight towards the exit door. At that time, we intervened, lest they would leave the building. It took some time before P1 managed to convince her dog to walk again on that corridor. After the study was completed, P1 explained to us that her dog might have felt nervous due to the circumstances of the trials.

Both inertial sensing algorithms (A/S and RoNIN) were able to successfully track the participants through the routes, except for the initial part of R1W for the first three participants before the implementation of the adaptive step length mechanism, as discussed earlier. Occasionally, the reconstructed path "cut" some corners (see e.g. Fig. 7 (c), RoNIN) or momentarily overshot the location (Fig. 9 (b), A/S), but the drift tracking mechanism of our Particle Filtering implementation [50] seemed to have worked well, at least for these trials. In particular, it is remarkable that both algorithms were able to track the rather chaotic path taken by P7 in R3W (Fig. 9 (d)).



(a)



(b)

Fig. 10. (a) Histogram of the distances between ground truth locations (shown as gray disks in Figs. 7–9 and locations reported by A/S (blue bars) and RoNIN (red bars). (b) Histogram of the distances between the locations reported by A/S and RoNIN at all times.

In order to benchmark the accuracy of these algorithms, we used the following strategy. From the videos taken of the participants, which were synchronized with the timestamped data recorded by the phones, we selected a number of time instances approximately corresponding to each turn taken by the participants. For each such instance, we identified the "ground truth" location of the participant on the map through careful visual inspection, by comparing the participant's position with landmarks (wall corners) of known location visible in the video. While this visual procedure is prone to error, we estimate that the error was generally less than 1 meter. The ground truth locations are shown as grey disks in Figs. 7–9. Then, from the recorded localization tracks produced by either algorithm, we identified the estimated location at the same time instances considered above (shown as small colored circles in Figs. 7–9), and computed localization errors as distances to the corresponding ground truth locations. Fig. 10 (a) shows the histograms of the localization errors for both algorithms. Paired t-test yields a value of $p = 0.054$, which suggests that the null hypothesis of no difference in means cannot be rejected. This data must be analyzed with the caveat that, as mentioned earlier, the ground truth data itself has a radius of uncertainty of 1 meter. We thus consider an upper bound to the actual errors by adding 1 meter to all measured localization error values. Using this conservative measure, we found that the localization error at the selected locations was less than 2 meters for 44% of the A/S measurements and for 48% of the RoNIN measurements. The 90 percentile error was at 3.4 meters for A/S and 3.75 meters for RoNIN. It is also interesting to measure the discrepancy in the location computed by A/S and RoNIN. Fig. 10 (b) shows the histogram of distances at all times between A/S and RoNIN localization. This distance was found to be less than 2 meters in 75% of the cases. The 90 percentile distance was 3.11 meters.

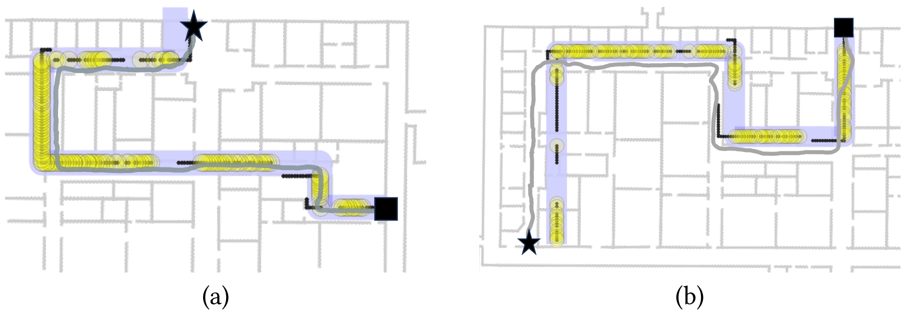


Fig. 11. Examples of successful backtracking trials (hybrid matching). (a): Route R2B for participant P5. (b): R1B for P4. Left panel: The way-in path is shown with a thick purple line ending at the black square. The length of each segment is given by the number of steps recorded, multiplied by the step length measured during calibration. A gray line shows the approximate path of the participant during the return phase, computed by visual inspection. Reliable matches are shown as yellow circles. Projected sequences are shown with black lines.

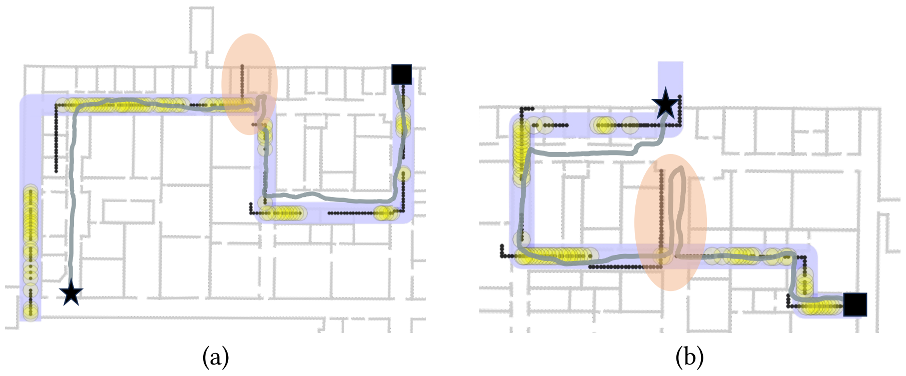


Fig. 12. See caption of Fig. 11. (a): route R1B, participant P2. (b): R2B, P2. Highlighted are situations in which the participant took a wrong path then walked back as directed by the app.

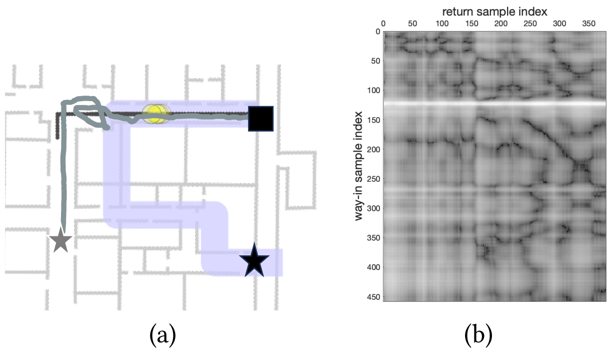


Fig. 13. See caption of Fig. 11. (a) Route R3B, participant P6. In this case, the app failed to track the participant. The gray star represents the point at which the trial was aborted; the black star is the desired destination. (b) Magnetic discrepancy for all pairs (i, j) of samples from way-in (vertical axis) and return (horizontal axis). Lighter gray indicates larger discrepancy.

4.3.2 Tracking and Guidance Performance: Backtracking. Examples of successful trials with the Backracking app are shown in Figs. 11–12. Note that the length of the segments in each route, obtained, as mentioned earlier, by multiplying the number of steps taken in the segments by the calibrated step length, did not necessarily match the length of the corresponding floor plan, due to the variable step length. Note that this is not a problem in itself: the goal of the Backtracking app is simply to match the walker's location during return with the same location during the way-in, in order to produce correct guidance notifications. Whether these reconstructed routes are metrically consistent with the floor plan is irrelevant for our purposes.

As shown in Tab. 2, six trials with the Backtracking app had to be aborted because the app failed to track the walker during the return route. In one of these trials (P3 in R2B), the participant had taken a wrong turn (for a reason described later in this section). The app correctly issued a "turn around" notification. However (as he later told us), P3 decided to instead keep walking, as he mistakenly thought that he remembered the path he had taken. As he kept walking away from the route, the app soon became unable to track him.

Two trials had to be aborted for route R3B. One of them (with P7) was due to incorrect way-in path simplification, consequent to a "meandering" way-in path (Fig. 4 (b)). In the case of P2 and P6, the culprit was a large magnetic field discrepancy. This is visible for P6 in Fig. 13 (b), which shows the magnetic discrepancy for all pairs (i, j) of samples from way-in (vertical axis) and return (horizontal axis) (see Sec. 3.2.2). The white horizontal line, visible for way-in sample index 120 and nearby samples, signifies that the magnetic field recorded at that location was significantly different from the magnetic field recorded at any location during return.

An interesting situation was encountered in route R2B. As can be seen in Fig. 8 (a), the return route (starting from the star in that figure) goes through two L-junction, then crosses an X-junction. In our original implementation of the Backtracking app, the notification issued upon entering a new segment was similar to that of the Wayfinding app: "Walk straight for about XX [meters/feet/steps]. Then, you will turn [left, right]." This is the type of notification that was issued after the second L-turn in the return route R2B, indicating a final right turn after walking for approximately 15 meters. What happened with P2, P3, and P4 is that, rather than walking through the X-junction, they turned right on it, presumably because they knew they had to turn right at some point. This situation did not happen with the Wayfinding app (in the reverse route) because, as discussed in Sec. 3.3.1, in the proximity of the X-junction, the Wayfinding app would notify the walker to

"Keep walking straight". This option is not available in the Backtracking app. The reason is that the Backtracking app is only aware of the way-in route: it cannot possibly know that the user traversed an X junction at some point, and thus cannot remind the user, on the return route, to "walk straight" when in the proximity of the junction. Thus, after P4, we decided to slightly change the notification format by removing the last sentence ("Then, you will turn [left, right]") when entering a new segment. This ostensibly minimal interface modification was sufficient to avoid incorrect turns at this junction for the remaining participants.

Participant P5 was not able to complete R1B, a route that initially proceeds straight through an X-junction (Fig. 7 (a)). P5 soon started veering to the left, hit a wall, then started walking around trying to find his way, taking multiple turns that the app was unable to match against the way-in route. We decided to abort that trial and to try again from the starting point. This time, P5 turned right instead of going straight. He told us that he knew he had to turn right at some point, based on the initial route description provided (through a left swipe on the Watch), which stated that he should walk for 27 meters, then turn right. Note that this is a similar situation as experienced in the X-junction for R2B, as described above. One experimenter made him notice that he had not walked for 27 meters yet. At that point, P5 changed course, and was able to successfully complete the trial.

While traversing route R1B, participant P2, after being notified by the app of an upcoming left turn, encountered difficulties when trying to locate the turn, and ended up getting trapped in an alcove before reaching the junction. After leaving the alcove, P2 faced another alcove at the other side of the corridor. Eventually, following directions from the app, the participant was able to return to the correct route.

The trial for P7 on route R1B had to be aborted; a second trial was likely unsuccessful. P7, as mentioned earlier, walked with a fast and very confident dog, and zipped through the X-junction where she was supposed to turn right. By the time she processed the notification asking her to turn around, she was already a long way down the corridor. Unfortunately, the Backtracking app was not able to recover tracking afterwards.

4.3.3 Final Questionnaire. The responses of the participants to the System Usability Scale questionnaire, administered at the end of the trials, are reported in Tab. 4. The overall score [8] was 80.36. Though the interpretation of SUS scores is the object of debate [9], this score converts to a percentile rank of 90% based on the distribution of scores reported in [58].

The open-ended questions, along with a summary of the responses, are listed below.

1. Do you think that the system always knew your location? Most participants replied with variations of "yes, most of the times" (99% of the times according to P7, 80% according to P1). However, P2 and P3 replied with "No". According to P2, there were moments in which localization was correct. P3 felt that localization was sensitive to how one walks.

2. Do you think that the system gave you the correct directions? P4–P7 replied "yes". P2 thought that it gave correct directions most of the time (80% of the time, according to P1). P3 said: "Yes, when it knew what it was doing."

3. The system often gives turning directions (such as "at the next junction, turn right") with some advance notice, which means that you need to find the turn using your cane/dog. Was this a problem for you? All participants stated that this was not a problem in general, with some adding specific comments. P1 said: "Once I got used to it, I sort of got it." P2 commented that it was not a problem unless there were obstacles he could bump against when moving closer to the wall in preparation for the turn. P4 said that only one time this created a problem, and precisely when, in R2B, she took a right turn too soon (a situation discussed in Sec. 4.3.2). She then added that she realized that she should have waited for a notification prompting her to take the turn. P6 said that, while not a

Table 4. System Usability Scale (SUS) responses. The overall SUS score [8] was 80.36.

	P1	P2	P3	P4	P5	P6	P7	Mean
1. I think that I would like to use this system frequently.	3	4	1	5	5	5	4	3.86
2. I found the system unnecessarily complex.	2	1	1	2	2	4	1	1.86
3. I thought the system was easy to use.	4	4	5	5	5	5	5	4.71
4. I think that I would need the support of a technical person to be able to use this system.	1	1	1	1	4	3	1	1.71
5. I found the various functions in this system were well integrated.	4	2	1	5	4	5	5	3.71
6. I thought there was too much inconsistency in this system.	3	3	1	1	2	4	1	2.14
7. I would imagine that most people would learn to use this system very quickly.	3	3	5	4	5	5	5	4.29
8. I found the system very cumbersome to use.	2	1	1	1	2	1	1	1.29
9. I felt very confident using the system.	4	4	3	5	4	5	5	4.29
10. I needed to learn a lot of things before	2	1	1	2	1	4	1	1.71

problem in itself, it would have been preferable if the notification times were more "consistent", in the sense of producing a prompt always at the same distance from the junction (whereas, due to localization errors, this distance was often variable). P7 commented that this was the only thing in the system that was not as accurate as she thought.

4. Were the notifications understandable? Too many notifications? Too few? All participants found that the notifications were "fine" or "just right". P2 elaborated on his reply, commenting that the system produced exactly the type of information he needed: it gave him a rough distance to the next turn, and then it gave him a heads-up before the turn. P1 added that she would have liked to know if there was a door to open. (Note the routes considered in the study did not require walkers to open any doors.) P4 and P6 commented very positively about landmark notifications. For example, P6 mentioned two announcements ("stairwell to the left", "bench to the right") as examples of useful information. Likewise, P4 was enthusiastic about the landmark notifications and thought they could be useful in practice. For example, she mentioned that sometimes she may feel thirsty, and it would be nice to know if she was passing by a drinking fountain. She also mentioned that landmark notification would be particularly useful when someone visits a place for the first time; this type of notification could be turned off once one is more familiar with the environment. Similarly, P2 thought that the amount of landmarks announced in the trials was just right. However, if he was really concentrated on the route, he probably would want fewer notifications.

5. Was it easy for you to use the Watch? All participants except for P2 said that they found it easy. P2 said that "it was an adventure" but also commented that with some more practice, he would have found it easier to use the Watch gestures.

6. What would you like to have in this app that is not already there? Both P2 and P5 said that they would like to have more contextual information about the space they are visiting. P1 asked about how one would find the starting point, and especially the correct direction when starting a route. P3 mentioned that he would like to be able to scroll through a route description one step at a time. (In the current implementation, a left swipe produces a description of the remaining route but does not let the user stop and repeat each step). P4 wanted the ability to manually add landmarks on the fly (e.g. when passing by a certain location of interest). P5 also mentioned that he would like to know which direction he was facing at each time, as this information would help him when he was confused.

7. Did you notice any difference between the Wayfinding system and the Backtracking system? In general, the participants found that the two apps were very consistent with each other (with P5 mentioning that if we hadn't told him that they were different apps, he would not have noticed.) P2 and P3 lamented the fact that the two apps used different units. As mentioned earlier, this was due to an implementation mistake, which was later corrected.

8. Do you think that using this app would make you feel safer or more confident when traveling alone in a new place? [Asked for each app separately.] All participants except for P3 replied with an enthusiastic "yes". P3 thought that he could use the Backtracking app for situations such as when at a conference with multiple tables in a large hall, if, for example, one wanted to go to the restroom then come back to the same table. P2 envisioned using it when visiting a new place (e.g., a large medical building). He would like to be able to have a menu from which he could narrow down the places he would need to go. He also mentioned that, even if the app were not able to take him precisely there, there would be a lot of value in at least helping him to get closer. P4 thought that the Wayfinding app would help even for buildings that she had visited before. She mentioned an example of a large office building she had started visiting recently. While walking there, she stopped to think about where to go next, and immediately several bystanders came to her, offering unrequested (and unwelcome) help. This app would help her in these situations of uncertainty. P4 also said that using this app would relieve her from having to constantly focus on keeping track of her position and deciding where to go - her brain would have more "bandwidth" for other things. This is particularly the case in noisy and distracting environments. She also mentioned a practical example of application of the Backtracking app: walking with a sighted guide in a medical building to reach a doctor's office, then having to walk back when there is no one around to help. P5 thought that the apps helped him create a mental map of the whole route and enjoyed the fact that he didn't have to memorize the whole path because the apps would give him prompts when traversing it.

5 DISCUSSION

By using inertial sensing, our apps did not rely on external infrastructure such as BLE beacons. If BLE beacons are available, they could certainly complement our system. In particular, this infrastructure could help mitigate accumulating drift, and offer a convenient initialization mechanism. We need to emphasize again that infrastructure-less systems that require fingerprinting (e.g., based on Wi-Fi beacons, magnetic field, or images or 3-D data that need to be acquired beforehand) suffer from the same sustainability issues as those who require specialized infrastructure: fingerprinting is a time-consuming and expensive operation, and only a limited number of public buildings and venues have been mapped in this way.

Our Wayfinding app does require a floor plan of the building to be visited. Indeed, it is difficult to conceive of a wayfinding system that would not need a map, with the potential exception of a system only based on verbal directions, which, however, would hardly scale to complex environments. We note that public locations such as schools or medical centers often make floor plan images of their buildings available online. For our tests, we digitized publicly available maps using SIM [67, 68], a web app that enables fast and precise tracing of a floor plan, producing a vectorized GeoGSON file as required by our Wayfinding app. SIM also converts the vectorized map into an embossable map at the desired scale, which can be useful for pre-journey exploration. When a map is not available, our Backtracking app provides guidance for users who want to trace back their path in a building. This can be useful in multiple situations, as highlighted by some of our participants (Sec. 4.3.3).

An alternative to purely inertial sensing is visual odometry using a SLAM system like Apple's ARKit [25]. Indeed, in separate experiments, we found this approach to be substantially more accurate than with inertial sensing alone (e.g., average positioning errors of less than 1 meter are reported in [15]). However, the difficulty and/or inconvenience of having to hold a phone with good, unoccluded field of view while walking with a cane or dog guide may, in our opinion, limit the appeal of this technology until the time wearable cameras (e.g., embedded in eyeglass frames) will become mainstream. In the meantime, our proposed system, which lets users conveniently keep the phone in their pocket, represents a viable alternative to camera-based technology.

One of the goals of our investigation was to evaluate whether the simple interface mechanism introduced in Sec. 3.3, which notifies walkers of an incoming turn with large advance notice, would be feasible and acceptable. Prior work (e.g. [2, 34, 42, 70, 71]) studied optimal sonification techniques to provide users with enriched spatial information, to inform users of the direction to landmarks (e.g., using 3D sound as in the Microsoft Soundscape project), and to ensure that walkers don't veer off their path and that they begin turning at the right time. We made the design decision to only use synthetic speech (accompanied by a chime sound and a vibration of the Watch) to convey navigation directions. In part, this was suggested by the fact that, due to the relatively poor accuracy of inertial odometry [48], information about the actual distance to the next waypoint or about the veering angle of the walker (which can be sonified as in [2]) may be unreliable. For this reason, as explained earlier, we opted for simply providing early notification of upcoming turns, along with general information about the next waypoint after a turn, and corrective directions when one missed a turn. We recognize, though, that some form of sonification could have help reduce some types of errors. For example, as described in Sec. 4.3.1, in some instances participants missed a turn even though a speech notification was produced at the correct time. It is quite possible that with an appropriately designed sonification interface that continuously informed the walker of the approximate distance to the waypoint, participants would have missed turns at a lower rate.

Regarding the early notification mechanism, we were not sure whether participants would have difficulty discovering the exact location of the junction, or whether they would find this modality burdensome or annoying. We were pleased to find that, by and large, our participants were able to negotiate these situations successfully, even in the face of structural impediments (e.g., when they would get stuck in an alcove when looking for an opening). What's more, participants, when asked whether they thought that advance notification was a problem (Sec. 4.3.3, question 3.), overwhelmingly asserted that this was not the case. The positive usability scores recorded from the SUS responses are another indication that our participants were generally satisfied with the interface design. We are aware, however, that the advance notification approach could fail, for example, if there are two openings close to each other on the same side of the corridor (say, within two meters of each other). In this case, the system could produce more contextual information (e.g., notifying the user that they will encounter two junctions and that they should take the second one). We should

In our study, interaction with the apps was only enabled through the Watch, a mechanism that was very well received by the participants. In practical situations, though, users may want to take the phone out of their pocket, e.g. to pick up a call or to send a text. These "natural" types of interactions will be allowed (and encouraged) in future studies. As mentioned in Sec. 3.1.1 the A/S algorithm assumes that the phone does not change orientation with respect to one's body, and therefore may produce incorrect values for the walking direction if one re-orients the phone with respect to their body. In these cases, RoNIN would be a preferable localization algorithm since, as mentioned earlier, the output of RoNIN is independent of the orientation of the phone with respect to the user. Although our tests have been conducted using A/S, our comparative analysis of the localization results (Fig. 10) has shown that RoNIN provides similar localization accuracy as A/S, boding well for its use in lieu of A/S in our Wayfinding app.

It is instructive to compare our user interface with that used in the wayfinding system of Apostolopoulos et al [4], also based on inertial navigation. Their *user as a sensor* mechanism mitigates localization errors by delegating the task of "resetting" one's position to the user, who is in charge of finding specific landmarks. While our tests have shown that the *user as a sensor* modality may not be necessary, at least for the environment considered, we believe that it could certainly complement our system, and represent an effective fallback mechanism in specific situations. For example, in our RouteNav indoor/outdoor navigation systems [51], we successfully used the *user as a sensor* to negotiate the entrance of tunnels and walking ramps, which had proven challenging.

5.1 Limitations

Environment. The building in which we tested our systems contained narrow and wide corridors, as well as fairly large open spaces. It thus can be considered representative of many public spaces (office buildings, schools, health centers). However, larger buildings (e.g., airports, transit hubs, large shopping malls) may be challenging for inertial-only wayfinding systems: in very large open spaces, particles disperse isotropically, and thus the Particle Filter may not be able to successfully correct for accumulating drift. The building considered in our study did not have junctions with corridors intersecting at other than 90 degrees. While arbitrary intersection angles may not represent a problem for Particle Filtering (which can adapt to any wall layout), a higher turning angle granularity may increase the error rate of the turn detector used for the Backtracking app (though 45 degrees turns were not shown to create problems in simulations with the WeAllWalk data set [22, 69]).

Starting location. Perhaps the most critical limitation of our approach is that users of both systems must start a route from a certain location and begin walking in a certain direction. In fact, the same requirement is found in prior work with inertial-based wayfinding [4, 17] and is a consequence of the dead-reckoning nature of this method. Note that, in principle, it is possible for the system to function even without knowledge of the starting position and orientation. As shown in [15], one may distribute a large number of particles uniformly across the environment, then after a certain period (several minutes) of walking time, the surviving particles would converge with good likelihood around the user's position. However, this would require the user to walk aimlessly and without guidance from the system for a while, something that may not be desirable in practice. We envision two possible solutions to the problem of starting location/orientation determination. One possibility is to use the "user as a sensor" paradigm. Users may start from a location that is well perceivable, such as the main entrance of a building or other identifiable landmarks, such as the end of a staircase, and be advised to start walking in a certain known direction (in our example, straight across building entrance or coasting a wall to the left or to the right of the staircase entrance.) A different solution, one that we are currently exploring, is to create a hybrid system that can use visual data (e.g. via automatic landmark recognition [11]) for sporadic "fixes" using computer vision techniques when the location and orientation of the user need to be ascertained. After that, the user may move the smartphone back to their pocket and be tracked by the inertial system. A promising direction we are currently considering is the possibility of self-localization by matching a picture taken from the current location directly against a floor plan without the need for prior pictures taken of the same environment [14, 30, 32].

Backtracking performance. Our Backtracking app cannot make use of Particle Filtering, for the simple reason that this app is unaware of the layout of the walls in the building. In order to increase resiliency to accumulating orientation drift, we use structural geometric constraints (representing paths as sequences of straight segments and turns at discrete angles), graph-based sequence alignment, and magnetic field consistency, to match a return path with its way-in counterpart. The relatively large number of failed trials (6 out of 21) highlights the difficulty of the task and the need for further research. In general, we found that our Backtracking app works well when users do not deviate too much from the way-in path. Large detours from the original route are difficult for the algorithm to handle, as it becomes liable to mismatches. Another problem is related to the spatial variability of the magnetic field within the width of a large corridor or hallway, which may cause mismatches when, during return, the user walks on a different trajectory within the same space. In future work, we will consider more adaptive statistical models for the magnetic anomalies in indoor environments [36], also relying on existing magnetic field data sets [66].

Population sample. The age of our participants (median: 72 years) was relatively high. Tests with a population with a wider range of ages would allow one to establish whether age represents a factor in the one's ability to follow directions from the apps. Likewise, a larger sample of dog guide users would help identify specific user interface parameters that could be optimized for these walkers viz-a-viz long cane users.

6 CONCLUSIONS

We have presented the results of an experiment with seven blind participants who used two custom-designed iPhone apps for wayfinding and backtracking. These apps use different inertial-based mechanisms to track the user in a known floor plan (Wayfinding) or to match the user's path with a previously taken path (Backtracking). Our experiments showed that inertial-based tracking, coupled with a carefully designed user interface, is a suitable technology for wayfinding in a building characterized by a network of corridors when the initial position/orientation of the user is known. Our study also showed that our approach to backtracking (which assumes no knowledge of the building's layout) has good potential, but improvements are needed to increase its robustness in practical scenarios.

Inertial-based localization is attractive because it does not rely on external infrastructure, and because it doesn't require use of the smartphone's camera, which would force the user to hold the phone in one hand while walking, or attach it to their clothing or to a lanyard. Our experimental results suggest that this approach is effective in the case for buildings characterized by networks of corridors. Confirmation from users when they reached some well-perceived landmarks could be integrated for improved robustness.

We were heartened to hear from our participants that they found that the apps they tested could make them feel safer and more confident in their independent travel. While we are aware of the limitations of our systems, this type of feedback (along with the good scores from the SUS questionnaire) confirms that our proposed technology has serious potential for practical use as a navigational aid.

7 ACKNOWLEDGMENTS

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. Special thanks go to the participants who volunteered for this study.

REFERENCES

[1] Iyad Abu Doush, Sawsan Alshatnawi, Abdel-Karim Al-Tamimi, Bushra Alhasan, and Safaa Hamasha. 2017. ISAB: integrated indoor navigation system for the blind. *Interacting with Computers* 29, 2 (2017), 181–202.

[2] Dragan Ahmetovic, Federico Avanzini, Adriano Baratè, Cristian Bernareggi, Marco Ciardullo, Gabriele Galimberti, Luca A Ludovico, Sergio Mascetti, and Giorgio Presti. 2023. Sonification of navigation instructions for people with visual impairment. *International Journal of Human-Computer Studies* 177 (2023), 103057.

[3] Dragan Ahmetovic, Cole Gleason, Chengxiong Ruan, Kris Kitani, Hironobu Takagi, and Chieko Asakawa. 2016. NavCog: a navigational cognitive assistant for the blind. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services*. 90–99.

[4] Ilias Apostolopoulos, Navid Fallah, Eelke Folmer, and Kostas E Bekris. 2014. Integrated online localization and navigation for people with visual impairments using smart phones. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 3, 4 (2014), 1–28.

[5] Shiri Azenkot, Richard E Ladner, and Jacob O Wobbrock. 2011. Smartphone haptic feedback for nonvisual wayfinding. In *The proceedings of the 13th international ACM SIGACCESS conference on Computers and accessibility*. 281–282.

[6] Leo Botler, Michael Spörk, Konrad Diwold, and Kay Römer. 2020. Direction finding with uwb and ble: A comparative study. In *2020 IEEE 17th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*. IEEE, 44–52.

[7] Lesley A Brabyn and John A Brabyn. 1983. An evaluation of" Talking Signs" for the blind. *Human Factors* 25, 1 (1983), 49–53.

[8] John Brooke. 1996. Sus: a “quick and dirty’usability. *Usability evaluation in industry* 189, 3 (1996), 189–194.

[9] John Brooke. 2013. SUS: a retrospective. *Journal of usability studies* 8, 2 (2013), 29–40.

[10] Lina Chen, Jinbin Wu, and Chen Yang. 2020. MeshMap: A magnetic field-based indoor navigation system with crowdsourcing support. *IEEE Access* 8 (2020), 39959–39970.

[11] Lidong Chen, Yin Zou, Yaohua Chang, Jinyun Liu, Benjamin Lin, and Zhigang Zhu. 2019. Multi-level scene modeling and matching for smartphone-based indoor localization. In *2019 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*. IEEE, 311–316.

[12] Rong Chen and Jun S Liu. 2000. Mixture Kalman Filters. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 62, 3 (2000), 493–508.

[13] Seyed Ali Cheraghi, Vinod Namboodiri, and Laura Walker. 2017. GuideBeacon: Beacon-based indoor wayfinding for the blind, visually impaired, and disoriented. In *2017 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 121–130.

[14] Hang Chu, Dong Ki Kim, and Tsuhan Chen. 2015. You are here: Mimicking the human thinking process in reading floor-plans. In *Proceedings of the IEEE International Conference on Computer Vision*. 2210–2218.

[15] Ryan Crabb, Seyed Ali Cheraghi, and James M Coughlan. 2023. A Lightweight Approach to Localization for Blind and Visually Impaired Travelers. *Sensors* 23, 5 (2023), 2701.

[16] Marcus Edel and Enrico Köppe. 2015. An advanced method for pedestrian dead reckoning using BLSTM-RNNs. In *2015 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. IEEE, 1–6.

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

[18] Xirui Fan, Jing Wu, Chengnian Long, and Yanmin Zhu. 2017. 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*. 13–18.

[19] Alexander Fiannaca, Ilias Apostolopoulous, and Eelke Folmer. 2014. Headlock: a wearable navigation aid that helps blind cane users traverse large open spaces. In *Proceedings of the 16th international ACM SIGACCESS conference on Computers & accessibility*. 19–26.

[20] German Flores, Sri Kurniawan, Roberto Manduchi, Eric Martinson, Lourdes M Morales, and Emrah Akin Sisbot. 2015. Vibrotactile guidance for wayfinding of blind walkers. *IEEE transactions on haptics* 8, 3 (2015), 306–317.

[21] German Flores and Roberto Manduchi. 2018. Easy return: an app for indoor backtracking assistance. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–12.

[22] German H Flores and Roberto Manduchi. 2018. Weallwalk: An annotated dataset of inertial sensor time series from blind walkers. *ACM Transactions on Accessible Computing (TACCESS)* 11, 1 (2018), 1–28.

[23] V Fox, Jeffrey Hightower, Lin Liao, Dirk Schulz, and Gaetano Borriello. 2003. Bayesian filtering for location estimation. *IEEE pervasive computing* 2, 3 (2003), 24–33.

[24] J. Fruchterman. 1996. Talking maps and GPS systems. Paper presented at The Rank Prize Funds Symposium on Technology to Assist the Blind and Visually Impaired, Grasmere, Cumbria, England.

[25] Giovanni Fusco and James M Coughlan. 2020. Indoor localization for visually impaired travelers using computer vision on a smartphone. In *Proceedings of the 17th International Web for All Conference*. 1–11.

[26] Aura Ganz, James Schafer, Siddhesh Gandhi, Elaine Puleo, Carole Wilson, and Meg Robertson. 2012. PERCEPT indoor navigation system for the blind and visually impaired: architecture and experimentation. *International journal of telemedicine and applications* 2012 (2012).

[27] Nicholas A Giudice. 2018. Navigating without vision: Principles of blind spatial cognition. In *Handbook of behavioral and cognitive geography*. Edward Elgar Publishing, 260–288.

[28] Nicholas A Giudice, William E Whalen, Timothy H Riehle, Shane M Anderson, and Stacy A Doore. 2019. Evaluation of an accessible, real-time, and infrastructure-free indoor navigation system by users who are blind in the mall of america. *Journal of Visual Impairment & Blindness* 113, 2 (2019), 140–155.

[29] Sachini Herath, Hang Yan, and Yasutaka Furukawa. 2020. RoNIN: Robust neural inertial navigation in the wild: Benchmark, evaluations, & new methods. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 3146–3152. Code available at https://github.com/Sachini/ronin.

[30] Harlan Hile and Gaetano Borriello. 2008. Positioning and orientation in indoor environments using camera phones. *IEEE Computer Graphics and Applications* 28, 4 (2008), 32–39.

[31] Md Elias Hossain, Khandker M Qaiduzzaman, and Mostafijur Rahman. 2020. Sightless helper: an interactive mobile application for blind assistance and safe navigation. In *Cyber Security and Computer Science: Second EAI International Conference, ICONCS 2020, Dhaka, Bangladesh, February 15-16, 2020, Proceedings 2*. Springer, 581–592.

[32] Henry Howard-Jenkins, Jose-Raul Ruiz-Sarmiento, and Victor Adrian Prisacariu. 2021. Lalaloc: Latent layout localisation in dynamic, unvisited environments. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 10107–10116.

[33] Hochul Hwang, Hee-Tae Jung, Nicholas A Giudice, Joydeep Biswas, Sunghoon Ivan Lee, and Donghyun Kim. 2024. Towards Robotic Companions: Understanding Handler-Guide Dog Interactions for Informed Guide Dog Robot Design. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. 1–20.

[34] Roberta L Klatzky, James R Marston, Nicholas A Giudice, Reginald G Golledge, and Jack M Loomis. 2006. Cognitive load of navigating without vision when guided by virtual sound versus spatial language. *Journal of experimental psychology: Applied* 12, 4 (2006), 223.

[35] Elmar Krainz, Viktoria Lind, Werner Moser, and Markus Dornhofer. 2016. Accessible way finding on mobile devices for different user groups. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct*. 799–806.

[36] Jian Kuang, Taiyu Li, Qijin Chen, Baoding Zhou, and Xiaoji Niu. 2022. Consumer-grade inertial measurement units enhanced indoor magnetic field matching positioning scheme. *IEEE Transactions on Instrumentation and Measurement* 72 (2022), 1–14.

[37] Jian Kuang, Xiaoji Niu, Peng Zhang, and Xingeng Chen. 2018. Indoor positioning based on pedestrian dead reckoning and magnetic field matching for smartphones. *Sensors* 18 (2018), 4142. Issue 12.

[38] Bineeth Kuriakose, Ida Marie Ness, Maja Å skov Tengstedt, Jannicke Merete Svendsen, Terese Bjørseth, Bijay Lal Pradhan, and Raju Shrestha. 2023. Turn Left Turn Right-Delving type and modality of instructions in navigation assistant systems for people with visual impairments. *International Journal of Human-Computer Studies* (2023), 103098.

[39] Binghao Li, Thomas Gallagher, Andrew G Dempster, and Chris Rizos. 2012. How feasible is the use of magnetic field alone for indoor positioning?. In *2012 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. IEEE, 1–9.

[40] Jennifer Listgarten, Radford Neal, Sam Roweis, and Andrew Emili. 2004. Multiple alignment of continuous time series. *Advances in neural information processing systems* 17 (2004).

[41] Jack M Loomis, Reginald G Golledge, and Roberta L Klatzky. 2001. GPS-based navigation systems for the visually impaired. *Fundamentals of wearable computers and augmented reality* 429 (2001), 46.

[42] Jack M Loomis, James R Marston, Reginald G Golledge, and Roberta L Klatzky. 2005. Personal guidance system for people with visual impairment: A comparison of spatial displays for route guidance. *Journal of visual impairment & blindness* 99, 4 (2005), 219–232.

[43] Roberto Manduchi and James M Coughlan. 2014. The last meter: blind visual guidance to a target. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 3113–3122.

[44] Elliot W Martin, Emily Farrar, Evan Magsig, Susan A Shaheen, Ashley Auer, Sarah Hoban, Booz Allen Hamilton, et al. 2020. *Accessible Transportation Technologies Research Initiative (ATTRI) Impact Assessment White Paper*. Technical Report. United States. Department of Transportation. Intelligent Transportation ....

[45] Amanda Morris. 2021. Navigational Apps for the Blind Could Have a Broader Appeal. *The New York Times* (20 December 2021).

[46] Masayuki Murata, Dragan Ahmetovic, Daisuke Sato, Hironobu Takagi, Kris M Kitani, and Chieko Asakawa. 2018. Smartphone-based indoor localization for blind navigation across building complexes. In *2018 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 1–10.

[47] Vishnu Nair, Greg Olmschenk, William H Seiple, and Zhigang Zhu. 2022. ASSIST: Evaluating the usability and performance of an indoor navigation assistant for blind and visually impaired people. *Assistive Technology* 34, 3 (2022), 289–299.

[48] Eshed Ohn-Bar, João Guerreiro, Kris Kitani, and Chieko Asakawa. 2018. Variability in reactions to instructional guidance during smartphone-based assisted navigation of blind users. *Proceedings of the ACM on interactive, mobile, wearable and ubiquitous technologies* 2, 3 (2018), 1–25.

[49] Jennifer Palilonis, Caitlin Cambron, and Mianda Hakim. 2023. Challenges, Tensions, and Opportunities in Designing App-Based Orientation and Mobility Tools for Blind and Visually Impaired Students. In *International Conference on Human-Computer Interaction*. Springer, 372–391.

[50] Peng Ren, Fatemeh Elyasi, and Roberto Manduchi. 2021. Smartphone-based inertial odometry for blind walkers. *Sensors* 21, 12 (2021), 4033.

[51] Peng Ren, Jonathan Lam, Roberto Manduchi, and Fatemeh Mirzaei. 2023. Experiments with RouteNav, A Wayfinding App for Blind Travelers in a Transit Hub. In *Proceedings of the 25th International ACM SIGACCESS Conference on Computers and Accessibility*. 1–15.

[52] Timothy H Riehle, Shane M Anderson, Patrick A Lichter, Nicholas A Giudice, Suneel I Sheikh, Robert J Knuesel, Daniel T Kollmann, and Daniel S Hedin. 2012. Indoor magnetic navigation for the blind. In *2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, 1972–1975.

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

[54] David A Ross and Bruce B Blasch. 2000. Wearable interfaces for orientation and wayfinding. In *Proceedings of the fourth international ACM conference on Assistive technologies*. 193–200.

[55] Paul-Edouard Sarlin, Ajaykumar Unagar, Mans Larsson, Hugo Germain, Carl Toft, Viktor Larsson, Marc Pollefeys, Vincent Lepetit, Lars Hammarstrand, Fredrik Kahl, et al. 2021. Back to the feature: Learning robust camera localization from pixels to pose. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 3247–3257.

[56] Daisuke Sato, Uran Oh, João Guerreiro, Dragan Ahmetovic, Kakuya Naito, Hironobu Takagi, Kris M Kitani, and Chieko Asakawa. 2019. NavCog3 in the wild: Large-scale blind indoor navigation assistant with semantic features. *ACM Transactions on Accessible Computing (TACCESS)* 12, 3 (2019), 1–30.

[57] Daisuke Sato, Uran Oh, Kakuya Naito, Hironobu Takagi, Kris Kitani, and Chieko Asakawa. 2017. Navcog3: An evaluation of a smartphone-based blind indoor navigation assistant with semantic features in a large-scale environment. In *Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility*. 270–279.

[58] Jeff Sauro. 2011. *A practical guide to the system usability scale: Background, benchmarks & best practices.* Measuring Usability LLC.

[59] Victor R Schinazi, Tyler Thrash, and Daniel-Robert Chebat. 2016. Spatial navigation by congenitally blind individuals. *Wiley Interdisciplinary Reviews: Cognitive Science* 7, 1 (2016), 37–58.

[60] Farzaneh Shahini, Vanessa Nasr, and Maryam Zahabi. 2022. A Friendly Indoor Navigation App for People with Disabilities (FIND). In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, Vol. 66. SAGE Publications Sage CA: Los Angeles, CA, 1922–1926.

[61] Yuanchao Shu, Zhuqi Li, Börje Karlsson, Yiyong Lin, Thomas Moscibroda, and Kang Shin. 2019. Incrementally-deployable indoor navigation with automatic trace generation. *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, 2395–2403.

[62] Vaibhav Singh, Rohan Paul, Dheeraj Mehra, Anuraag Gupta, Vasu Dev Sharma, Saumya Jain, Chinmay Agarwal, Ankush Garg, Sandeep Singh Gujral, Mahesh Balakrishnan, et al. 2010. 'Smart'Cane for the Visually Impaired: Design and Controlled Field Testing of an Affordable Obstacle Detection System. In *TRANSED 2010: 12th International Conference on Mobility and Transport for Elderly and Disabled PersonsHong Kong Society for RehabilitationS K Yee Medical FoundationTransportation Research Board.*

[63] Martin Štancel, Jan Hurtuk, Michal Hulič, and Jakub Červeňák. 2021. Indoor atlas service as a tool for building an interior navigation system. *Acta Polytech. Hung* 18 (2021), 87–110.

[64] William Storms, Jeremiah Shockley, and John Raquet. 2010. Magnetic field navigation in an indoor environment. *2010 Ubiquitous Positioning Indoor Navigation and Location Based Service*, 1–10.

[65] Kalyan Pathapati Subbu, Brandon Gozick, and Ram Dantu. 2013. LocateMe: Magnetic-fields-based indoor localization using smartphones. *ACM Transactions on Intelligent Systems and Technology (TIST)* 4, 4 (2013), 1–27.

[66] Joaquín Torres-Sospedra, David Rambla, Raul Montoliu, Oscar Belmonte, and Joaquín Huerta. 2015. UJIIndoorLoc-Mag: A new database for magnetic field-based localization problems. In *2015 International conference on indoor positioning and indoor navigation (IPIN)*. IEEE, 1–10.

[67] Viet Trinh and Roberto Manduchi. 2019. Semantic interior mapology: A toolbox for indoor scene description from architectural floor plans. In *24th International Conference on 3D Web Technology*, Vol. 2019. NIH Public Access.

[68] Viet Trinh, Roberto Manduchi, and Nicholas A Giudice. 2023. Experimental evaluation of multi-scale tactile maps created with SIM, a web app for indoor map authoring. *ACM transactions on accessible computing* 16, 2 (2023), 1–26.

[69] Chia Hsuan Tsai, Peng Ren, Fatemeh Elyasi, and Roberto Manduchi. 2021. 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, 117–122.

[70] Bruce N Walker and Jeffrey Lindsay. 2006. Navigation performance with a virtual auditory display: Effects of beacon sound, capture radius, and practice. *Human factors* 48, 2 (2006), 265–278.

[71] Jan M Wiener, Simon J Büchner, and Christoph Hölscher. 2009. Taxonomy of human wayfinding tasks: A knowledge-based approach. *Spatial Cognition & Computation* 9, 2 (2009), 152–165.

[72] William R Wiener, Richard L Welsh, and Bruce B Blasch. 2010. *Foundations of orientation and mobility*. Vol. 1. American Foundation for the Blind.

[73] Michele A Williams, Erin Buehler, Amy Hurst, and Shaun K Kane. 2015. What not to wearable: using participatory workshops to explore wearable device form factors for blind users. In *Proceedings of the 12th International Web for All Conference*. 1–4.

[74] Hang Yan, Sachini Herath, and Yasutaka Furukawa. 2019. Ronin: Robust neural inertial navigation in the wild: Benchmark, evaluations, and new methods. *arXiv preprint arXiv:1905.12853* (2019).

[75] Chris Yoon, Ryan Louie, Jeremy Ryan, MinhKhang Vu, Hyegi Bang, William Derksen, and Paul Ruvolo. 2019. 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*. 210–221.

[76] Tuo Yu, Haiming Jin, and Klara Nahrstedt. 2019. Shoesloc: In-shoe force sensor-based indoor walking path tracking. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 3, 1 (2019), 1–23.

[77] Maryam Zahabi, Xi Zheng, Azima Maredia, and Farzaneh Shahini. 2022. Design of Navigation Applications for People with Disabilities: A Review of Literature and Guideline Formulation. *International Journal of Human–Computer Interaction* (2022), 1–23.