An Approach for Hybrid Indoor/Outdoor Navigation

Stephan Wagner, Ngewi Fet, Marcus Handte, Pedro José Marrón
Networked Embedded Systems, University of Duisburg-Essen, Germany
{stephan.j.wagner, ngewi.fet, marcus.handte, pjmarron}@uni-due.de

***Abstract*—The wide availability of GPS-enabled devices has fueled the development of location-based applications. Among the most popular are navigation applications that guide users along their way. Due to the limited availability and precision of GPS in typical indoor environments, most navigation applications are limited to outdoor scenarios. To mitigate this, researchers and practitioners have been developing alternative systems to capture the position of a mobile device indoors. Using these systems, they clearly show that it is possible to provide precise location information at a reasonable cost. However, past applications in this domain have often focused exclusively on indoor scenarios.**

In this paper, we describe an approach to integrate the indoor and outdoor world to provide a seamless hybrid navigation experience. The approach builds upon our past work on indoor localization and integrates with our existing infrastructure for outdoor navigation that has been developed in the GAMBAS European research project. To evaluate the approach, we present data gathered from a real-world deployment as part of the SIMON European research project.

I. Introduction

Over the past decades, the increasing availability of GPS-enabled mobile devices has fueled the development of location-based services and applications. Among the most popular ones are navigation applications that provide context-aware instructions to guide their users. However, since GPS requires an unobstructed line of sight between the GPS-receiver and multiple satellites, its availability in typical indoor environments is severely limited. As a result, most wide-spread navigation applications focus either on outdoor scenarios or provide only coarse-grained support for indoor environments.

To provide an alternative to GPS, researchers and practitioners have been developing various approaches for indoor localization that leverage vision, sound, IR- or RF-technology, to name a few (c.f. [1]). Using these systems, they clearly show that it is possible to provide precise location information at a reasonable cost. However, most state-of-the-art applications in this domain are focusing on isolated indoor scenarios. As a consequence, they only provide support in a particular indoor environment such as a shopping mall or a university building and thus, they are unable to provide a true end-to-end navigation experience.

A primary goal of the SIMON European research project is to provide navigation support for mobility impaired persons by means of a mobile navigation application that is tailored towards their specific needs. Based on feedback from mobility impaired users, we quickly learned that providing navigation support for this group should not stop at building entrances. Instead, many end users expressed the need to be able to find the elevator locations inside a building or to navigate along specific pavements that have been installed into the ground for blind persons.

To realize such requirements in a seamless manner, it is necessary to extend outdoor navigation, as enabled by GPS, with extensions that transfer this experience to indoor environments. In this paper, we describe the approach that we have taken in SIMON to accomplish this. We have extended our service infrastructure used to implement an outdoor navigation application as part of the GAMBAS European research project. Thereby, we introduce models to capture indoor environments, new services to support integrated indoor and outdoor routing as well as localization algorithms to support indoor navigation. To evaluate our approach, we present data from a real-world deployment in a transit station.

The remainder of the paper is structured as follows. In Section II, we outline the existing service infrastructure for outdoor navigation applications. We then detail the extensions for indoor navigation in Section III. In Section IV, we present an evaluation of the overall system. Finally, Section V describes related work and Section VI concludes the paper.

II. Existing Infrastructure

As a starting point for hybrid navigation, we rely on a service infrastructure for outdoor navigation that we have developed as part of the GAMBAS European research project. As data sources, the infrastructure relies on geo-data provided by the OpenStreeMap project (OSM) and publicly available transit schedule data that adheres to the General Transit Feed Specification (GTFS). As shown in Figure 1, the infrastructure consists of 4 core functions that are exposed via 3 service interfaces as follows:

- *Maps:* The maps service enables a client to retrieve a visual (2D) representation of the user's environment in the form of an image tile. To enable client- and service-side caching and to provide compatibility with existing client-side visualization libraries, the service reuses the addressing conventions of OpenStreetMap in which tiles are addressed via X, Y, Z parameters where Z defines a particular zoom level and X, Y are used to define the longitude and latitude. For more details see [2].
  To create images we use a custom tile renderer that first loads the geometry for a particular tile from a spatial geometry index. Thereafter, it computes a suitable drawing ordering and renders the visible elements sequentially. To do this, the render relies on a set of rules which customize their visual appearance (e.g. line strokes, colors, icons).

- *Places:* The places service enables a client to translate strings into locations (geocoding) and back (reverse geocoding). For the latter, a client can specify both, a single location and a range (i.e. an area) which can, for example, be used to create POI overlays. The resulting locations are categorized (e.g. restaurant, bus stop, metro station, parking spot) and contain the full address details as well as the GPS coordinate. To resolve the queries, the service uses a Lucene-based search index populated from the OSM data (which contains buildings, streets, cities, countries, etc.) and from the GTFS data (which contains transit stops and stations).
- *Directions:* The directions service enables a client to compute routes from an origin to a destination using a particular travel modality (by car, on foot or, via public transportation). In addition, a client may specify restrictions (e.g. avoid toll roads), a departure or an arrival time as well as a maximum number of route alternatives that shall be computed. As a result, the service returns a list of routes that consists of steps and segments. A segment represents an individual instruction and a step groups them for a particular movement modality. Using this model, it is possible to represent multi-modal trips.
  To generate the routes, the directions service contains two routing engines, one for street and one for transit routing. The street routing engine relies on an index that efficiently handles the gradual retrieval of the graph that represents the road network during the route computation. Internally, the engine uses an A* search algorithm (cf. [3], Chapter 4) that can be customized using rules, e.g. to configure weights based on the road type.
  To compute routes that include public transportation, the service relies on a transit routing engine that can interpret data that follows the General Transit Feed Specification [4]. From a high level point of view, the transit router is similar to the street routing engine in that it uses an A* algorithm to compute the fastest route. However, since transit vehicles only stop at their stops and drive on fixed schedules, the transit router must consider the travel time and the way to/from/between the stop(s). For the latter, the transit router leverages the street router to compute the required walking segments.

III. Infrastructure Extensions

In the following, we describe our approach for extending the existing infrastructure for outdoor navigation to support hybrid navigation. To do this, we first describe the model that we use to represent indoor environments and we discuss its application for indoor routing. Then, we describe our approach for indoor localization and finally, we describe integration issues.

A. Indoor Modelling

To minimize the modelling effort, we introduce a simple model for indoor environments that enables us to supports navigation. Similar to the outdoor model, we need to be able to define locations, compute routes and visualize the

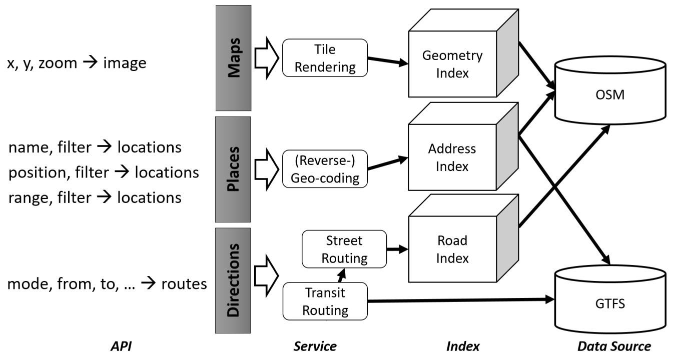

Fig. 1. Outdoor Navigation Infrastructure

environment. However, since buildings may exhibit multiple floors, a 2D model is generally not sufficient. To address this issue, our model introduces a layered 2D coordinate system which extends WGS84 with an integer to model levels. Based on this, we add the elements shown in Figure 2:

- *Building:* As one might guess, this element represents a single building. It consists of a set of layers and links.
- *Layer:* A layer represents a single floor in a building. It consists of an integer to model its level, a set of paths, zones and places as well as one or more images to provide a graphical representation. In order to align an image with the outdoor environment, a layer includes three anchors.
- *Anchor:* An anchor defines the mapping from the coordinate system of an image (i.e. % of width/height) to a longitude and latitude in WGS84. Thus, using three anchors that form a triangle in each coordinate system, it is then possible to compute an Affine Transform that enables the correct embedding (i.e. translation, scaling, rotation and shearing) of the image into a 2D outdoor map.
- *Place:* A place represents a point of interest to the user modeled as a WGS84 coordinate[1] with a name. Similar to the OSM data model, a place can be tagged with key-value pairs to define additional attributes such as its category or its name in different languages.
- *Path:* A path is a polyline that represents a possible path through a layer. A path is modelled as a sequence of coordinates and like places, it can be tagged to include additional semantics, e.g. to differentiate pavements for visually impaired persons from ways for other users. As we explain later on, paths can be configured as being uni- or bi-directional and they can be associated with weights in order to support more complicated routing scenarios.
- *Zone:* A zone represents a particular polygonal area on a floor modelled as a sequence of WGS84 coordinates. Using zones, it is possible to model freely traversable areas such as rooms, hallways, etc., as well as obstacles contained in them. In addition, by attaching tags, we can optionally assign names that are meaningful for the user

[1]Note that places, paths and zones are defined as part of a layer. Thus, there is no need to define their level as this is implied by the model.

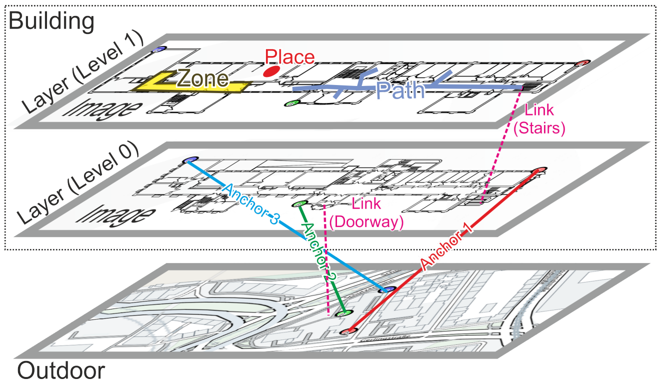

Fig. 2. Indoor Model

(e.g. subway area, shopping zone).

- *Link:* Links are similar to paths, in the sense that they define a way that can be traversed by a user and that they can be tagged to express different types (e.g. stairs, elevators, escalators, etc.). The main difference is that the coordinates in links are associated with layers in order to connect them with each other.

B. Indoor Routing

Given the indoor model described above, there are different approaches to support routing. First, we could require that the model should contain zones for all rooms as well as links between the layers. This would allow us to create a routing graph by a) creating edges between neighboring polygons and b) creating edges between polygons that share a link and c) creating edges inside of each polygon among all pairs of its vertices (including the vertices resulting from links, places and obstacles that lie inside of it) and then d) removing all edges that will intersect with the polygon boundaries or the boundaries of an obstacle polygon. An example of a result of this construction is depicted in Figure 3a) which shows a (non-convex) room containing an obstacle. The room has two doors (start, end) and contains one place. The dashed lines represent the generated edges.

However, while an automatic construction would guarantee optimality, there is little control over the output and the resulting routes would often deliberately stick to following the walls (as indicated by the route from start to end in Figure 3a). From a pure geometrical perspective, the latter problem could be solved by using a polygon offset algorithm (e.g. [5] to ”shrink” the polygon by some threshold distance (e.g. 1 meter), followed by using the vertices of the ”smaller” polygon to compute the graph. The loss of control, however, is more problematic in practical applications. As an example consider that in a transport scenario, we might want to guide a stream of passengers along a particular way in order to minimize collisions of people traveling in opposite directions.

As an alternative approach, we could require that the model contains a complete set of paths between all elements for all layers. Then, we could simply construct a routing graph by combining the paths that share a common vertex. By using

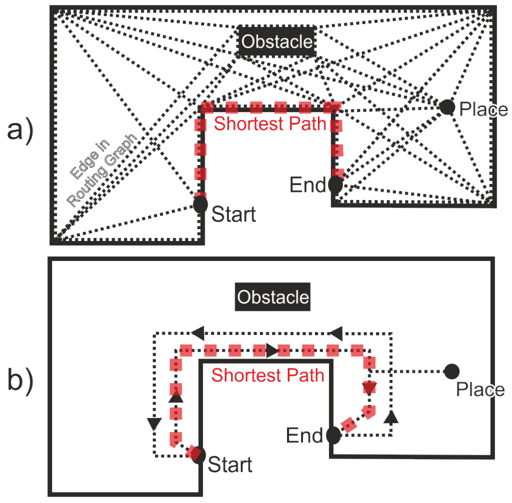

Fig. 3. Indoor Routing

uni-directional paths, we could even separate the route output depending on the route direction. This is depicted in Figure 3 b). However, as the number of possible endpoints increases, the explict model becomes increasingly complicated. As an example, consider that computing the shortest route from ”end” to ”place” with the explicit paths shown in Figure 3 b) would result in two 90 degree turns, even though there is a straight line that connects them. Fixing this by means of explicit modeling would require several additional edges. In fact, if we allow the route computation to start from any possible location in the room, explicit modelling falls apart.

To combine the flexibility of explicit models with the optimality of automatic graph generation, we combine the two approaches as follows. First, we model the ways that the routing engine should propose to direct a user throughout the building by means of links and paths. For simple scenarios with bidirectional paths, this requires the definition of a spanning tree that connects all zones with the entrances/exits of the building. Once this is done, we integrate places (or arbitrary route origins and destinations) automatically by using the polygon-based generation procedure described above. Thereby, we mark the auto-generated edges.

To compute a route, we first get (or add if necessary) the origin and destination points and run Dijkstra’s shortest path algorithm [6]. However, instead of using the geometric distance as edge weights, we use a linear (distance-based) weighting function, i.e. for each edge $e_{i,j} = (v_i, v_j)$ we define a weight w_e as $w_e = a_e * d_e + b_e$ with $d_e := distance(v_i, v_j)$ where a_e is an edge specific weighting factor and b_e is an edge specific weighting constant.

For edges derived from the model, we extract the factor and the constant either from the tags of the model element or use a reasonable default in case there is no explicit specification. For example, for paths we can set the default to $a = 1, b = 0$ which results in a purely distance-based weight. For stairs, we might set $a = 1.5, b = 0$ in order to model the higher walking effort. For elevators, we could set $a = 0, b = 60$ in order to model the fact that on average, there will be a certain waiting time and so on. For auto-generated edges, we set relatively higher values for a and b to ensures that modelled links are preferred.

To realize different routing profiles for different user requirements (e.g. wheelchair accessible, visually impaired, etc.), we use different sets of default values and different tag names to enable profile-specific weight factors and constants. Thereby, we include a special weight constant to express absolute exclusions (e.g. escalators for visually impaired or stairs for wheelchair accessible routing profiles) that should not be returned in any result.

C. Indoor Localization

For indoor localization, we build upon our previous work on RF-based fingerprinting [7], [8]. Although many details can be found in [7], we briefly describe the main ideas to keep this paper self contained.

After deploying a set of n signal emitters (beacons) in the environment, fingerprinting-based localization is performed in two phases. In the first phase, we collect signal strength measurements ($RSSI$) from the beacons at a number of *known* locations. These measurements are stored with their location (LOC) as a vector $(LOC, RSSI_1, RSSI_2, ..., RSSI_n)$ which is refered to as fingerprint. Once the data collection is completed, the resulting fingerprints are uploaded to a localization service which uses them during the second phase to compute locations. For this, a user's device performs a signal strength measurement at an *unknown* location and sends the vector $(RSSI_1, RSSI_2, ..., RSSI_n)$ to the service. The service then matches the vector received from the client against the fingerprints and returns the location of the best match(es).

Different approaches primarily differ in a) the preprocessing of the fingerprints after the data collection, b) the matching metric and c) the post-processing of the matches. For our system, we rely on the procedures described in the following:

1) Training Phase: To speed up the data collection, we continuously collect timestamped fingerprints while walking along a predefined path (polyline) throughout the environment. Whenever a vertex on the line is passed, we manually add a timestamped marker to the measurements. When the data is then uploaded to the localization service, we interpolate an estimated location for each measurement based on its timings and the timestamped markers. Based on our experiences, the resulting location estimates are sufficiently accurate as long as the user does not vary the walking speed between two vertexes.

For the preprocessing, we first apply our model on signal attenuation described in [9] and then we overlay the target area with a fixed size grid and aggregate the measurements that have been made inside the same cell. Thereby, we estimate the average RSSI μ as well as the standard deviation σ. The result is a set of $n * m$ pairs $(\mu_{i,j}, \sigma_{i,j})$, one for each beacon ($i$) in each of the cells ($j$).

2) Localization Phase: For the matching, we first compute the probability of having received a certain fingerprint inside a cell by assuming independence between the measurements of individual signal sources and assuming that measurements would follow a normal distribution modelled by $(\mu_{i,j}, \sigma_{i,j})^2$. This means that for each cell, we compute the cell probability p_j as the normalized product of the individual beacon probabilities $p_{i,j}$ of the cell. To avoid unnecessary computations, we use the clustering approach of Horus [11]. This means that we start off computing the probabilities from the strongest to the weakest RSSI and abort the computation if the probability becomes (very close to) zero.

$$p_{j,i}(RSSI) = \frac{1}{\sigma_{i,j}\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{RSSI-\mu_{i,j}}{\sigma_{i,j}})^2} \tag{1}$$

$$p_j((RSSI_1, RSSI_2, ..., RSSI_n)) = \prod_{i=1}^{n} \frac{p_{j,i}(RSSI_i)}{\sum_{j=1}^{m} p_{j,i}(RSSI_i)} \tag{2}$$

The set of cell probabilities is then used as an input into a regular Particle Filter (cf. [3], Chapter 15). Applying such a filter requires knowledge about the physical distance between the cells as well as an assumption on the maximum movement speed. The latter can be estimated from the scenario (e.g. 5-7 m/s if we assume that people will be running fast). The former can be computed directly from the indoor model. However, instead of repeatedly using Dijkstra' algorithm to compute the distances between cells, we use a Floyd-Warshall algorithm [12] to compute the shortest paths between all cells since it requires less space and is faster. In each filtering step, the resulting distance matrix is then first used to redistribute the prior probabilities among the reachable cells and then multiplied with the cell probability (of the measurement). Finally, the filtered matrix of cell-probabilities is post-processed to compute a final location. For this, we compute the location returned to the user as the average of the m most probable cells where m is a configurable parameter which we adjust based on the deployment.

3) Disconnected Operation: While fingerprinting has the advantage of being precise, it requires the availability of a network connection to the localization service. Ensuring this is often not a problem since mobile network operators spend effort to optimize their network coverage. However, due to the specific construction of (some) buildings, there can be spots which are not covered or exhibit a very weak connectivity. Some obvious examples are elevator shafts, metallic staircases or floors that are multiple levels under ground. In addition, the approach does not lean towards a client-side implementation since it requires access to the fingerprint data which – depending on the size of the building and the number of beacons – can become fairly large.

To enable disconnected localization without requiring the devices to download large amounts of data, we use a simple RSSI-based trilateration approach as a fallback source of location information. However, instead of using a fixed function

[2] Existing literature indicates that these assumptions may not hold, e.g. [10], so there may be room for optimizations. Yet, our experiences with a number of different deployments indicate that this approach is robust and works well.

to model the relation between distance and RSSI, we use our training data collected during fingerprinting to compute an n-th degree polynomial function for each individual beacon. To do this, we estimate the beacon location as the location with the highest RSSI measurement. Then, we take the RSSI measurements of the beacon (that are on the same level as the beacon) and compute their (2D) distance to the beacon location. We then use the resulting pairs of RSSI values and distances to perform a polynomial regression. Finally, for each beacon, we memorize the beacon's location (including its level), the beacon's polynomial constants as well as the minimum and maximum RSSI value used to derive the constants. The latter represents the range of inputs (i.e. the domain) that should result in reasonable distance estimates.

During the localization, we use this to estimate the distance between a beacon and the location of the user using its measured RSSI value as follows:

1) If the received RSSI of a beacon is lower than the minimum RSSI used to compute the polynomial, we discard the beacon and do not attempt to compute a distance estimate.
2) If the received RSSI of a beacon is higher than the maximum RSSI used to compute the polynomial, we estimate the distance as 0.
3) In all other cases, we compute the distance estimate as the polynomial value for the measured RSSI.

Using at least three distance estimates d_i at the positions (x_i, y_i), we then formulate the localization problem for a user at (x, y) with the ranging error e_i as:

$$\sqrt{(x - x_i)^2 + (y - y_i)^2} = d_i + e_i \tag{3}$$

Finally, we use a Gaussian solver to compute (x, y) while minimizing the square sum of the ranging errors, i.e. $min(\sqrt{\sum e_i^2})$. Thereby, we limit the solver to a small number of iterations which trades off precision for low computational cost.

D. Integration

To integrate the indoor model, routing and localization into the outdoor infrastructure, we extend the services as follows:

- *Maps:* To visualize the indoor environment, we extend the tile-based interface with an optional parameter to specify the level. If no level is provided, we render the outdoor map as is. If a level is provided, we render the images from the indoor plan intersecting the tile. To do this properly, we draw the images using the affine transform defined by the associated anchors. As a result, a mobile device can show a combined visualization of an indoor and an outdoor environment by drawing the indoor tile (of a desired level) on top of the outdoor plan.
- *Places:* We extend the location model of the address index by adding an integer to capture the level of a location to integrate the indoor places into the place service. Similar to the maps extension, we then enable mobile devices to define specific levels and level ranges as part of the query specification.
- *Directions:* To compute hybrid routes, we extend the interface of the directions service with an optional level for the route origin and destination. In addition, we extend the street router as follows. When computing a walking route, we first determine whether the origin or destination is inside a building. If they are both in the same building, we compute an indoor route. If they are both outside, we compute an outdoor route. In all other cases, we first compute an indoor route from the origin or destination to the exit that is closest to the destination or origin respectively. Then we use the exit coordinate(s) as input to the outdoor street router. This ensures that the resulting routes are not going through buildings. However, the resulting routes may not be optimal. To guarantee optimality, we could integrate the indoor and outdoor routing graphs but for our deployment the route resulting from this extension did not exhibit detours. In addition, we must also modify the transit router. Since our GTFS data does not contain the precise location of stops in buildings and since it does not allow the modelling of levels, we extend the transit router to query the indoor model for refined locations. To do this, we run queries against the place service upon start up to determine the (possibly refined) location of the stops. Thereafter, we use the updated locations to compute transfers using our modified street router resulting in hybrid routes.

In addition to these extensions, we add a new service for indoor localization. To support (online) fingerprint-based localization, the service accepts RSSI measurements and returns computed locations. To support offline localization, the service additional enables clients to download the beacon data (i.e. beacon id, position, polynomial coefficients and domain) for an area. Aside from this beacon data, the service also returns an approximate bounding box for the building(s) with the associated levels which we use to customize the map visualization when the user's viewport lies over a building.

IV. Evaluation

Over the last three years, we have been developing and testing the infrastructure as part of the SIMON European research project. Thereby, we have modelled the Moncloa transit station[3] in Madrid and we have deployed an indoor localization system in approximately two-thirds of its area. Next, we first describe experiments with the individual components, before we present some experiences from their integration.

A. Modeling

As basis for navigation, we created a model of the station that covers all 4 floors and includes images, places, paths, links and zones. Specifically, it contains 17 places that represent platforms, stops and information desks, 33 links representing

[3]Moncloa is one of the largest stations in the city that connects intercity buses with local buses and two metro lines.

elevators, escalators, stairs and ramps, 9 zones (with 271 vertices) to model different areas and 23 paths (with 187 vertices) between them. The resulting routing graph exhibits 279 connections, but as described previously, additional connections are automatically added for the origin and destination using the zones. Over time, we adapted the model several times to meet user requirements. However, despite the changes, we estimate that the total modelling time taken lies under 8 hours.

B. *Routing*

To capture the overhead for indoor routing, we instrument the routing service and install it on a laptop (Intel Core I7 6650U, 2.2GHz, 16GB RAM) to execute a number of experiments. First, we use the graph to compute all (shortest) indoor routes between all places. This results in 272 (16*17 places) routes with an average length of 124m (max. 278m) each of which is computed in 0.68ms on average (max. 2.11ms). As a result, we can expect that the addition of indoor routing will introduce only marginal additional delays.

To verify this, we compute a transit route with an instrumented client program from the Moncloa Station to the Madrid city center using our original routing service (that does not support indoor routing) and compare it to the integrated service (which performs hybrid routing). To minimize the measurement variability, we run both, the client program and the services, on the laptop and we repeat the route request 1000 times. On average, the original system requires 148.77ms (max. 280ms) to compute the outdoor route. For the same input, the hybrid service requires 165.72ms (max. 343ms) which corresponds roughly to a 11% increase. However, in a less synthetic setting, the absolute overhead of 17.02 ms can easily be tolerated as it would be outweighed by the variabilities resulting from the (mobile) network.

C. *Localization*

To support indoor localization, we deployed 49 beacons throughout the 4 floors of the station and captured 9000 fingerprints. The deployment took one day, of which approximately two hours were spent collecting fingerprints. The remaining time was spent to attach the beacons. One day later, we collected 400 fingerprints to evaluate the performance.

Figure 4 depicts the cumulative error distribution for the online (fingerprinting) and offline (ranging) algorithms in the Moncloa station. With an average accuracy of 4.15m and a 90th percentile error of 7.9m, the fingerprinting algorithm provides a localization performance that is sufficient for step-by-step navigation. To put these figures in perspective, consider that the grid cells depicted in Figure 5 are 5x5m wide. Thus, in the vast majority of cases, the location will be either in the correct or in the neighboring cell.

Figure 4 clearly indicates a noticeable drop in both average accuracy (7.49m) as well as 90th percentile error (12.96m), when looking at the ranging algorithm (with quadratic polynomial-based range estimates), used to support disconnected operation. While overall, this is good enough to bridge intermediate disconnections, we further analyzed the

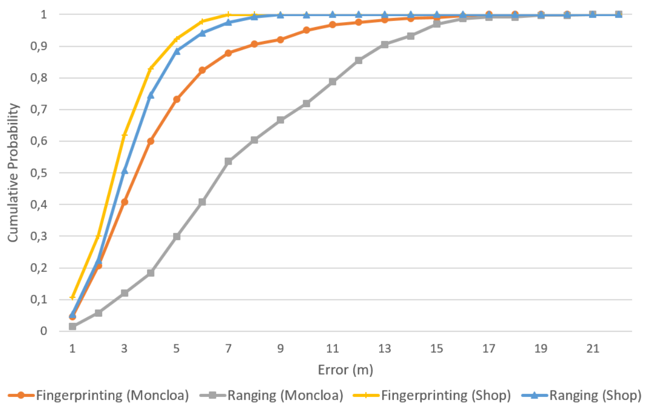

Fig. 4. Localization Performance (CDF) for Moncloa Station and Shop

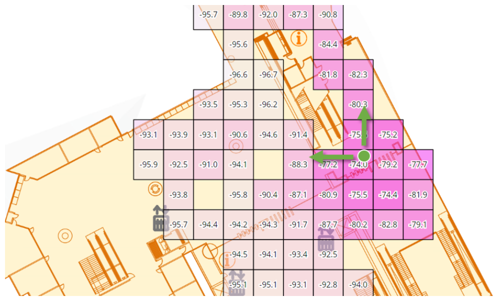

Fig. 5. Example: Non-radial Signal Propagation of a Beacon (5x5m Grid)

data to identify the reason for the degradation. During the analysis, we found that increasing the order of the polynomials only caused marginal improvements. Thus, we speculated that the drop was primarily caused by obstacles and multi-path effects. A closer look at the signal propagation further strengthened this assumption.

Figure 5 shows an example for this. The figure shows the average signal strength of a beacon on one of the floors aggregated in 5x5m cells. The beacon location is indicated by the point. As hinted by the non-radial color intensity of the cells, the signal strength of the beacon not only depends on the distance but also on the direction. For example, when comparing the 2-hop neighbor to the north with the 2-hop neighbor to the west, there is an 8db difference. Consequently, it is not possible to accurately capture this with a one-dimensional (i.e. distance-based) function.

To verify this, we run the algorithms over a data set that we collected in a large shop (c.f. Figure 4). Within the shop we installed 50 beacons on the ceiling. The shopping area is about one-third of the size of the Moncloa station and contains shelves that are approximately 1.2m high. Due to the denser deployment, the average accuracy of the fingerprinting algorithm rises to 2.75m (4.73m error at 90%). When comparing this with the ranging algorithm, the drop in performance is significantly lower (i.e. avg. 3.18m with 5.30m at 90%). This is a direct result of the fact that due to the low shelf sizes and the beacons mounted at the ceiling, most beacons exhibit a radial

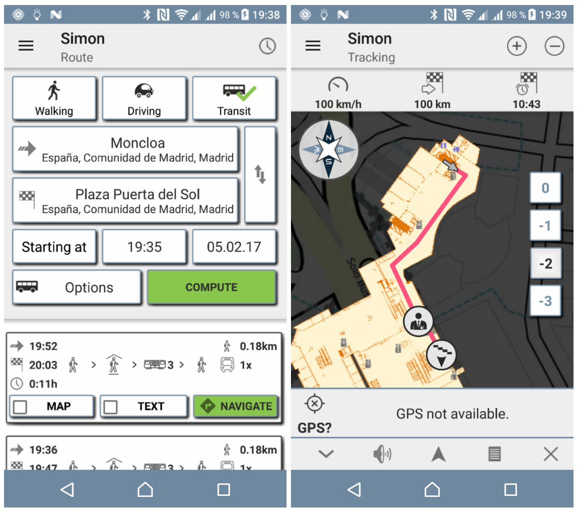

Fig. 6. Simon Mobile for Android

pattern that can be accurately captured using a polynomial.

Similar to the Moncloa deployment, the data for the shop indicates that using a quadratic polynomial already provides a good estimate and thus, we think that this is a good choice in general. As a result, the amount of data that must be transfered to the mobile device (i.e. beacon id, position, polynomial coefficients and domain) to enable offline localization will typically not exceed a few kilobytes. However, the achievable accuracy will vary based on the environment.

D. Integration

To test the integration, we have implemented a mobile application for Android and IOS that supports hybrid navigation. Using the extended infrastructure, the application is able to compute multi-modal routes using various public transport options available in Madrid. When used in Moncloa, it additionally enables indoor navigation as part of the trips. This is depicted in Figure 6 which shows an indoor route going through one of the floors of the station (right) as well as the route planning interface (left) with a multi-modal route that includes indoor navigation as the second step.

During the implementation and testing, we found that all components described in this paper are essential. Despite the fact that the Moncloa station lies up to three levels below the ground, most of the station is very well covered by mobile networks. However, there are a few areas where the connectivity is weaker. For these areas, the offline mode has turned out to be a suitable replacement.

Over the last 18 months, the application has been downloaded by more than 1000 users. During the tests performed in SIMON, the feedback on the user experience was generally positive and several users stressed the usefulness of hybrid navigation, especially for mobility impaired persons.

V. Related Work

In outdoor environments, the Global Positioning System (GPS) network of satellites provide signals which can be used to determine the location of a receiver on the earth's surface [13]. In the past, GPS signals were only available through specialized navigation devices. However, nowadays most mobile devices have embedded chips which can receive and process GPS signals. Other satellite based navigation systems such as GLONASS and Galileo [14] have been developed as alternatives to GPS. Mobile devices and receivers can support combinations of the satellite systems [15], sometimes leading to improvements in accuracy [16].

Providing step-by-step guidance along a route presents additional challenges such as generating optimal routes, tracking movement of user during the trip and providing directions along the route [17]. Outdoor environments have several layers of ontologies such as standardized location models [18], semantic modeling via street names, which provide a framework for developing navigation solutions for outdoor environments on a global scale. There are several services providing outdoor navigation services via mobile applications, such as Google Maps [19] and HERE WeGo Maps, [20] amongst others. These services and others rely on GPS signals to generate routes and provide directions during navigation for various modes of transport [21] [22].

In addition, several indoor location estimation systems have been developed using different signal technologies [23], such as RFID, WLAN, Bluetooth, Infrared, FM radio waves, etc., as well as hybrid combinations thereof [1] and indoor navigation systems based thereon. Indoor navigation comprises the continuous direction of (mainly pedestrian) users through an indoor environment [24]. This poses a different set of challenges to outdoor navigation in all the stages of navigation from locating the user, planning a route, representing the environment and interacting with the user [25]. Due to the lack of standardized location models for indoor environments, there are no universally applicable models which can be applied for generating routes from one point to a destination. Stoffel et al. [26] propose a graph-based spatial model to guide users through buildings. Unlike in outdoor environments where often the shortest path between two points suffice, in indoor environments the shortest path may not always be optimal such as when the use of elevators is required for users with special needs [27]. The authors in [28] seek to provide a classification of indoor objects while taking geometry, semantics and appearance into account in order to improve the quality of directions during navigation by helping the user effectively visualize routing information. The use of technologies such as Augmented Reality (AR) [29] which overlay the route on the environment for the user [30] have been proposed to address challenges with representing the environment and user interaction. However, AR still has several limitations with respect to portability, interaction and latency especially for users with physical or sensory impairments [31].

In addition to indoor and outdoor scenarios enumerated, there is an emerging third category of hybrid navigation services which provide navigation (routes and dynamic directions) for users in both indoor and outdoor scenarios [32]. A 3D pedestrian navigation system [33] based on dead-reckoning has been proposed for ubiquitous positioning. However this system relies on the use of RFID tags which are not in widespread use. [34] applies a similar approach with sensor fusion in order to achieve hybrid positioning in indoor and outdoor environments. However, navigation (and the transition) between outdoor and indoor environments is not addressed. Drishti [35] is an integrated indoor/outdoor for visually impaired. In this work, the user can switch from outdoor to indoor environment with a vocal command. This contrasts with our system, which can dynamically switch navigation modes between indoor and outdoor without user intervention.

VI. Conclusion

Due to the limited availability of GPS in indoor environments, most infrastructures that drive navigation applications are limited to outdoor scenarios. In this paper, we presented an approach for hybrid navigation that we have developed and tested over the last three years. Our approach extends an existing infrastructure for outdoor navigation with an indoor model as well as routing and localization algorithms and services. Based on our experiences in the SIMON project, we argue that the presented approach introduces reasonable costs and overheads while enabling a useful hybrid navigation experience. Since hybrid navigation is likely to become mainstream in the near future, we hope that the discussions in this paper will be valuable for others working in this area.

References

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

[2] OpenStreetMap Project, "OpenStreetMap Slippy Map Addressing," http://wiki.openstreetmap.org/wiki/Slippy_map_tilenames, 2017.

[3] P. Norvig and S. Russell, *Artificial Intelligence: A Modern Approach, 2nd Edition*. Prentice Hall, January 2013.

[4] Google Inc., "General Transit Feed Specification (GTFS)," https://developers.google.com/transit/gtfs/reference, 2016.

[5] X. Chen and S. McMains, "Polygon offsetting by computing winding numbers," in *2nd International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, 2005, pp. 565–575.

[6] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[7] N. Fet, M. Handte, S. Wagner, and P. Marrón, "Enhancements to the LOCOSmotion Person Tracking System," in *Evaluating AAL Systems Through Competitive Benchmarking*, ser. Communications in Computer and Information Science. Springer, 2013, vol. 386, pp. 72–82.

[8] N. Fet, M. Handte, and P. Marrón, "Autonomous adaptation of indoor localization systems in smart environments," *Journal of Ambient Intelligence and Smart Environments*, vol. 9, no. 1, pp. 7–20, 2017.

[9] N. Fet, M. Handte, and P. J. Marrón, "A model for WLAN signal attenuation of the human body," in *2013 ACM international joint conference on Pervasive and ubiquitous computing - UbiComp '13*. New York, New York, USA: ACM Press, 2013, p. 499.

[10] A. M. Ladd, K. E. Bekris, A. Rudys, G. Marceau, L. E. Kavraki, and D. S. Wallach, "Robotics-based location sensing using wireless ethernet," in *8th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '02. New York, NY, USA: ACM, 2002, pp. 227–238.

[11] M. Youssef and A. Agrawala, "The Horus location determination system," *Wireless Networks*, vol. 14, no. 3, pp. 357–374, Jan. 2007.

[12] R. W. Floyd, "Algorithm 97: Shortest path," *Commun. ACM*, vol. 5, no. 6, pp. 345–, Jun. 1962.

[13] I. A. Getting, "Perspective/navigation-the global positioning system," *IEEE Spectrum*, vol. 30, no. 12, pp. 36–38, Dec 1993.

[14] B. Hofmann-Wellenhof, H. Lichtenegger, and E. Wasle, *GNSS–global navigation satellite systems: GPS, GLONASS, Galileo, and more*, 1st ed. Springer Science & Business Media, 2008.

[15] P. Enge and P. Misra, "Scanning the Issue / Technology - Special issue on Global Positioning System," *Proceedings of the IEEE*, vol. 87, no. 1, pp. 3–15, 1999.

[16] A. Schmid, A. Neubauer, H. Ehm, R. Weigel, N. Lemke, G. Heinrichs *et al.*, "Combined Galileo/GPS architecture for enhanced sensitivity reception," *AEU - International Journal of Electronics and Communications*, vol. 59, no. 5, pp. 297–306, 2005.

[17] H. A. Karimi, *Outdoor Navigation*. Boston, MA: Springer US, 2011, pp. 17–57. [Online]. Available: http://dx.doi.org/10.1007/978-1-4419-7741-0_2

[18] S. A. True, "Planning the future of the world geodetic system 1984," in *PLANS 2004. Position Location and Navigation Symposium (IEEE Cat. No.04CH37556)*, April 2004, pp. 639–648.

[19] (2017) Google maps. Accessed: 2017-02-02. [Online]. Available: https://www.google.com/maps

[20] (2017) Here wego. Accessed: 2017-02-02. [Online]. Available: https://wego.here.com

[21] B. Elias, "Pedestrian navigation - creating a tailored geodatabase for routing," in *2007 4th Workshop on Positioning, Navigation and Communication*, March 2007, pp. 41–47.

[22] J. M. Krisp and A. Keler, "Car navigation computing routes that avoid complicated crossings," *International Journal of Geographical Information Science*, vol. 29, no. 11, pp. 1988–2000, 2015. [Online]. Available: http://dx.doi.org/10.1080/13658816.2015.1053485

[23] H. Liu, H. Darabi, P. Banerjee, and J. Liu, "Survey of Wireless Indoor Positioning Techniques and Systems," *IEEE Transactions on Systems, Man and Cybernetics, Part C*, vol. 37, no. 6, pp. 1067–1080, Nov. 2007.

[24] H. A. Karimi, *Indoor Navigation*. Boston, MA: Springer US, 2011, pp. 59–73. [Online]. Available: http://dx.doi.org/10.1007/978-1-4419-7741-0_3

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

[26] E.-P. Stoffel, B. Lorenz, and H. J. Ohlbach, "Towards a Semantic Spatial Model for Pedestrian Indoor Navigation," in *Advances in Conceptual Modeling Foundations and Applications*, ser. Lecture Notes in Computer Science. Springer, 2007, vol. 4802, pp. 328–337.

[27] P. M. Dudas, M. Ghafourian, and H. A. Karimi, "ONALIN: Ontology and Algorithm for Indoor Routing," in *2009 Tenth International Conference on Mobile Data Management: Systems, Services and Middleware*. IEEE, 2009, pp. 720–725.

[28] B. Hagedorn, M. Trapp, T. Glander, and J. Döllner, "Towards an Indoor Level-of-Detail Model for Route Visualization," in *2009 Tenth International Conference on Mobile Data Management: Systems, Services and Middleware*. IEEE, 2009, pp. 692–697.

[29] A. Mulloni, H. Seichter, and D. Schmalstieg, "Handheld augmented reality indoor navigation with activity-based instructions," in *13th International Conference on Human Computer Interaction with Mobile Devices and Services - MobileHCI '11*. New York, New York, USA: ACM Press, 2011, p. 211.

[30] J. Kim and H. Jun, "Vision-based location positioning using augmented reality for indoor navigation," *IEEE Transactions on Consumer Electronics*, vol. 54, no. 3, pp. 954–962, Aug 2008.

[31] D. Van Krevelen and R. Poelman, "Augmented reality: Technologies, applications, and limitations," 2007.

[32] H. A. Karimi, *Anywhere Navigation*. Boston, MA: Springer US, 2011, pp. 89–104.

[33] S. Koide and M. Kato, "3-D Human Navigation System Considering Various Transition Preferences," in *2005 IEEE International Conference on Systems, Man and Cybernetics*, vol. 1. IEEE, 2005, pp. 859–864.

[34] M. Kamil, M. Haid, T. Chobtrong, and E. Guenes, "Hybrid indoor and outdoor positioning with mems-based inertial motion sensors," in *Sensors and Measuring Systems 2014; 17. ITG/GMA Symposium*, June 2014, pp. 1–5.

[35] L. Ran, S. Helal, and S. Moore, "Drishti: an integrated indoor/outdoor blind navigation system and service," in *Second IEEE Annual Conference on Pervasive Computing and Communications, 2004.* IEEE, 2004, pp. 23–30.