Mind the Gap: Virtual Shorelines for Blind and Partially Sighted People

Daniel Koester Maximilian Awiszus Rainer Stiefelhagen

Karlsruhe Institute of Technology
Institute for Anthropomatics and Robotics
76131 Karlsruhe, Germany
{first.last}@kit.edu

Abstract

Blind and partially sighted people have encountered numerous devices to improve their mobility and orientation, yet most still rely on traditional techniques, such as the white cane or a guide dog. In this paper, we consider improving the actual orientation process through the creation of routes that are better suited towards specific needs. More precisely, this work focuses on routing for blind and partially sighted people on a shoreline like level of detail, modeled after real world white cane usage. Our system is able to create such fine-grained routes through the extraction of routing features from openly available geolocation data, e.g., building facades and road crossings. More importantly, the generated routes provide a measurable safety benefit, as they reduce the number of unmarked pedestrian crossings and try to utilize much more accessible alternatives. Our evaluation shows that such a fine-grained routing can improve users' safety and improve their understanding of the environment lying ahead, especially the upcoming route and its impediments.

1. Introduction

Route planning is a long-standing research area, most prominently by Dutch researcher Edsger Wybe Dijkstra, creator of one of the widest known routing algorithms [8]. This algorithm has been used in various applications, e.g., telecommunications (fastest packet route in a network) or traffic situations (shortest path to a given destination). Drivers have relied on such systems for decades, but only the recent invention of the smartphone has quite literally put these capabilities into everybody's hands. Lately, there is an increased development of routing algorithms suited towards pedestrians, most prominently, *Google Maps*[1] now has a pedestrian mode as of 2015.

Existing systems present a challenge to blind and partially sighted people, not only from an interaction standpoint, as they have different needs to communicate their intentions and receive feedback, but also because general pedestrian routing might not be favourable for them. Large crossings of wide streets, sometimes with multiple lanes, or confusing layouts, should be avoided and more secure alternatives preferred [16]. These could be: crossings with a pedestrian traffic signal, preferably with accessibility features such as aural and haptic features, and even zebra crossings, which are still much safer than informal crossings.

While *Orientation and Mobility Training* teaches how to utilize various cues, follow shorelines and navigate secure and fast from one point to another, such behaviour is not accounted for *at all* by existing systems. Route announcements like *"Turn right in 200 meters."* or *"Follow this road."* do not provide much benefit in this case, either.

Although current *Global Navigation Satellite Systems (GNSS)* provide rather coarse localization, especially in city street canyons, new *GNSS* systems, i.e., *Galileo*[2], are designed towards an average accuracy of less than a single meter. Thus more fine-grained applications, which require very precise location, become possible, e.g., showing drivers precisely which lane to switch to.

This work's focus are specialized routes for blind and partially sighted people. Utilizing publicly available *OpenStreetMap*[3] data, we retrieve roads, building facades, and road crossing locations. We employ a customized routing algorithm that satisfies pre-defined restrictions, e.g., avoid unmarked pedestrian crossings or follow along shorelines. Our algorithm yields routes that can be considered safer, while being of almost identical length. We evaluate it, as well as its modifications, on an exemplary urban area (see Figure 1). Furthermore, we conduct a *Wizard of Oz* experiment [7], where a blind person is following our generated routing instruction along two yet unknown routes.

[1] https://www.google.com/maps

[2] http://www.galileognss.eu

[3] http://www.openstreatmap.org

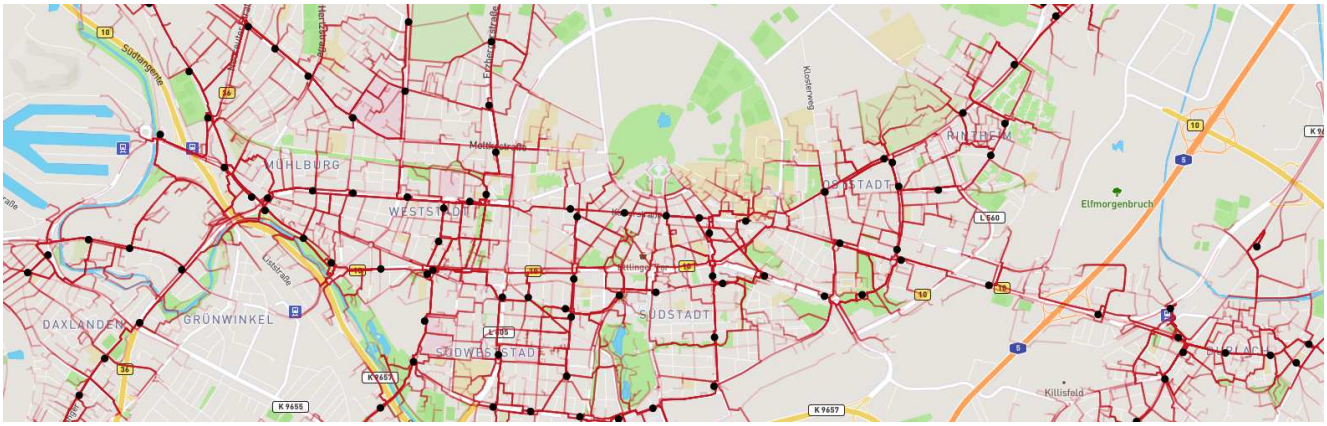


Figure 1: Excerpt of semi-randomly generated routes in an exemplary urban area. Public transit stations (black dots) are used as routing seeds, while the destinations are points in their neighbourhood, usually close to a street or walkway, as this represents a highly likely usage scenario for blind and partially sighted people. A darker segment color on generated routes (red lines) represents more frequently used segments (multiple routes use this segment). The routing algorithm tends to prefer certain connections and intersections, as they best satisfy our defined criteria. Figure best viewed in color.

2. Related Work

Navigation systems have become very widespread in recent years, mostly due to the ubiquity of smartphones and now even more through the addition of specialized navigation modes, i.e., for pedestrians, hikers or cyclists.

Andreev et al. [2] investigate special pedestrian requirements: On the one hand, they are able to utilize large traversable areas more freely to minimize their walking distance, while cars have to follow pre-defined road networks, on the other hand, they will prefer a longer route to utilize pedestrian bridges and underground connections. To generate such routes, numerous services exist today, e.g., *OpenRouteService*[4], *Google Maps*, or *Nokia Here*[5], as well as various based on *OpenStreetMap*, such as Schmitz et al. [20], who use a map based transformation in order to generate a pedestrian walkway network.

An overview of existing routing applications for blind and partially sighted people is given by Csapó et al. [6], but a lot of these will require the use of specialized, rather expensive, hardware that was designed for a single purpose only [3]. Various of these applications will provide *Point of Interest (POI)* related information only, i.e., read out street names in a given direction (*Android's Intersection Explorer* or *Talking Location*) or any other available location information pointed out on the smartphone screen (*Ariadne GPS*). A system that requires separate hardware, but provides specialized routing for blind and partially sighted people is the *Humanware BrailleNote GPS*[6], which consists of an external GPS module and a braille keyboard.

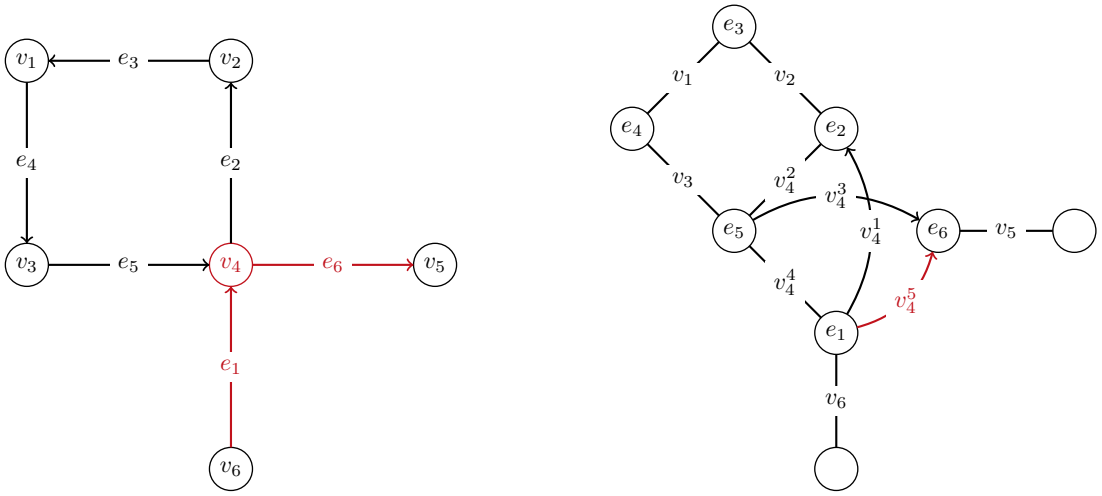
BlindSquare[7] might just be the most widely known and used application of them all. Like many others, it will read out nearby *POIs*, as well as street crossings, and prefer pre-selected favourites. This application benefits from it's inaccessible counterpart, *Foursquare*[8], sharing the same database that already consists of a huge amount of crowd sourced information about a copious number of public places and business alike [17].

Völkel et al. [22] developed another collaborative system to generate routes for a multitude of mobility impairments. User groups can share routes and adapt them to individual needs, while preserving individual users' privacy. Such collaborative efforts are already today an increasing factor in the generation of geolocation databases for various use cases [10].

Most, if not all, humans will already face problems just to walk in a straight line for an extended period without any other orientation aide, and sooner or later start to veer of to either side, or even walk in circles [21]. Thus, works like Panëels et al. [18] can already provide a great benefit to blind and partially sighted people alike. The authors use a smartphone's gyroscope and its compass to assist a partially sighted person walk on a straight line, i.e., without veering of, sending aural warning signals via headphones.

Lots of work has furthermore been performed to retrieve geospatial features from aerial imagery automatically, e.g., road networks [19, 11] or zebra crossings [12, 1, 14], but to the best of our knowledge, no work has so far tried to use this multitude of information to generate adaptable routes for users with specialized requirements.

[4] http://www.openroutservice.org
[5] http://www.here.com
[6] http://humanware.com
[7] http://www.blindsquare.com
[8] http://www.foursquare.com



(a) Directed Graph

(b) Edge Expanded Graph

Figure 2: An example *Directed Graph* (a) shown in comparison to its transformed *Edge Expanded Graph* (b) (please note the duality of node and edge layouts). While we cannot model *Turn-Restrictions* in (a), this becomes possible in (b): To disallow a right turn from edge e_1 on intersection v_4 into edge e_6, we can remove the highlighted edge v_4^5 in the *Edge Expanded Graph* (b). Similarly, confusing or dangerous intersection crossings could be avoided through modification of the corresponding weights, while an easy crossing at the same intersection would still be included in the model.

3. Generating Specialized Routes

The *OpenStreetMap (OSM)* project [9] is a collaborative effort to create an open map database, free to use for everybody. While the crowd sourced approach guarantees a certain amount of availability, the quality also varies greatly, but it has improved consistently since its creation in 2004 [24]. *OSM* data consists of open source formats and a plethora of programs have been created to view, update or transform its data, which has resulted in a strong ecosystem of applications depending on *OSM* data and a vibrant user community as well. One such project is the *Open Source Routing Machine (OSRM)* [15] that provides a user configurable routing engine to the public.

We use *OSM* data and the *OSRM* system exclusively in our experiments. Nonetheless, this approach is of course also possible using other data sources as well as other routing systems. Instead of relying on crowd sourced data, it would be very beneficial to acquire the same data from land registry offices, which often have such data available in a very high quality, but do not always easily provide it to the public at all, even less likely in a readily machine readable format.

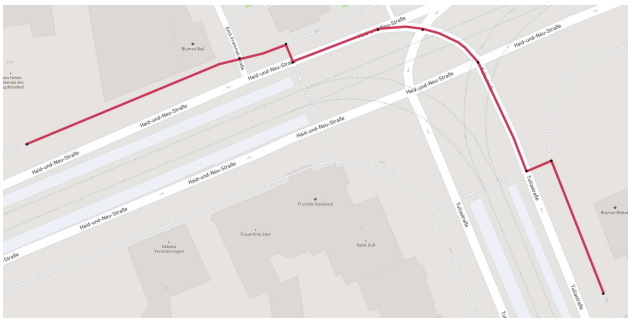
3.1. Map Data and Routing Graph

OSM data can be retrieved directly from the project in various formats. Most commonly, dataset consist of a list of points with longitude, latitude and various other informations, such as a list of connections to other points or metadata specifying the type of object they belong to. These *maps* can be transformed into a *Directed Graph*, a presentation more suitable for algorithms commonly used in routing scenarios. However, such a *Directed Graph* inhibits one major drawback: It models only the connections between its points, while the edges usually contain information about the types of road, their distance or speed limits. For our scenario, we would also like to model *Turn-Restrictions* [13] in order to prevent certain transitions and assign different weights to intersections themselves, depending on their level of difficulty. Using *OSRM*, we can transform our existing *Directed Graph* (Figure 2a) into an *Edge Expanded Graph* (Figure 2b) [5, 23].

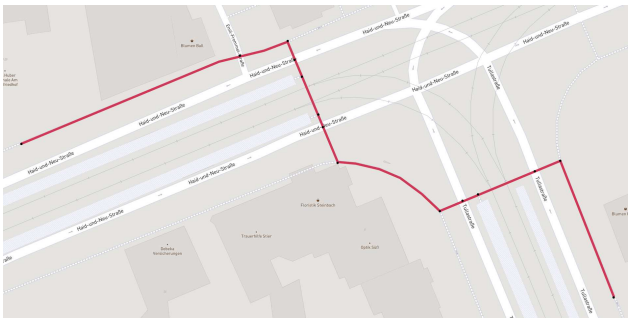
3.2. Base Route Generation

We use the *OSRM* engine to generate a coarse, mostly road network based route. This route also directly represents our further search area for future steps, greatly reducing computational overhead. We are considering urban areas only, however *OSRM* generated routes can contain all kinds of *defects*, rendering them largely unsuitable for pedestrian navigation. The route is then stored in a *GeoJSON* [4] format for further processing. We rely on the *Mapbox Qt SDK*[9] to generate our examples and for algorithmic debugging purposes, as it provides a highly customizable way to visualize map and route data.

[9] http://github.com/mapbox/mapbox-gl-native



(a) OSRM



(b) Ours

Figure 3: An optimized pedestrian walkway in an exemplary urban area. We compare the original *OSRM* routing algorithm (a) with our modified version (b). Please note how our algorithm crosses the two streets perpendicular at the intersection and uses pedestrian walkways, while the original routing algorithm follows the road itself around the intersection's corner.

3.3. Routing Prioritizations

3.3.1 Walkways

By default, *OSRM* considers road networks and walkway paths equally, i.e, the chance of integration of a road surface is the same as that of a pedestrian walkway. Thus, one can simply decrease the maximum possible speed on all roads to be less than that of walkways, which modifies their edge weights correspondingly in the edge expanded graph. We also apply equal constraints to bicycle lanes that are separated or integrated into the pedestrian walkway, to avoid cyclist collisions whenever possible and reduce potentially harmful walkway stretches when there is interleaved cyclist traffic. An exemplary walkway based prioritization is provided in Figure 3, where it can be seen that the resulting route is much *saner* if judged by a human, as it uses the existing, more *natural* street crossings for pedestrians, instead of following the road around the intersection's corner.

3.3.2 Accessible Pedestrian Signals

Similar to the walkway prioritization, we consider pedestrian traffic signals to be a preferred pedestrian crossing opportunity, i.e., if they are accessible pedestrian signals [16]. To address this problem, we distinguish between different crossing types and prefer their usage in the listed order: accessible pedestrian signals with aural and/or haptic signals, inaccessible pedestrian signals, zebra-crossings and uncontrolled pedestrian crossings (informal crossings). An exemplary pedestrian traffic signal based prioritization is provided in Figure 4b. It can be seen, that depending on the chosen parameters, i.e., the incurred penalties for informal crossings, the route might change dramatically and as shown in this selected case, seems unintuitive to an observer, but still provides a measurable safety benefit for blind and partially sighted people.

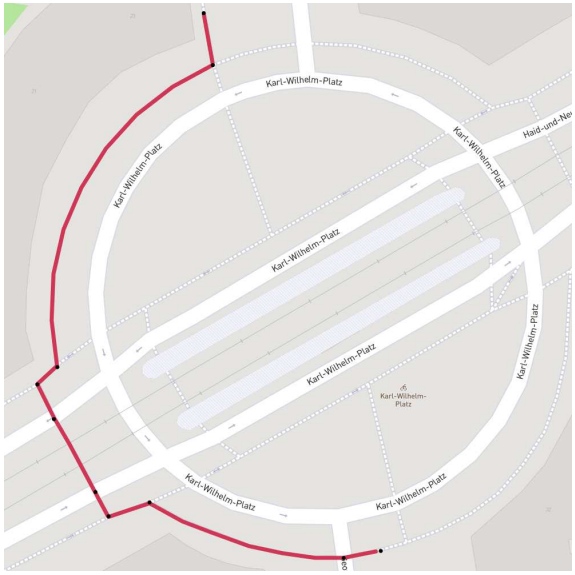
3.3.3 Shorelines – Real and Virtual

Lots of blind and partially sighted people mostly rely on their white cane for close range orientation and guidance, although guide dogs are also common. A very common intuitive and expedient approach is to follow inner or outer shorelines that are easily distinguishable using the cane, e.g., sidewalk curbs or building facades. Sadly, to the best of our knowledge, all currently existing navigation solutions don't incorporate this specific reliance on such *natural* features. This can lead to problematic situations, e.g., when a navigation system asks a user to cross a street or place in a certain position where no such shoreline is available, while there might exist a suitable one nearby. This increases confusion and uncertainty about one's surroundings, as the requested routing might conflict with the safest or *natural* path. In order to adapt our routing to this conflict, we incorporate available shoreline information, which allows us to generate improved and very fine-grained routing steps, closely modeled after *Orientation and Mobility* training.

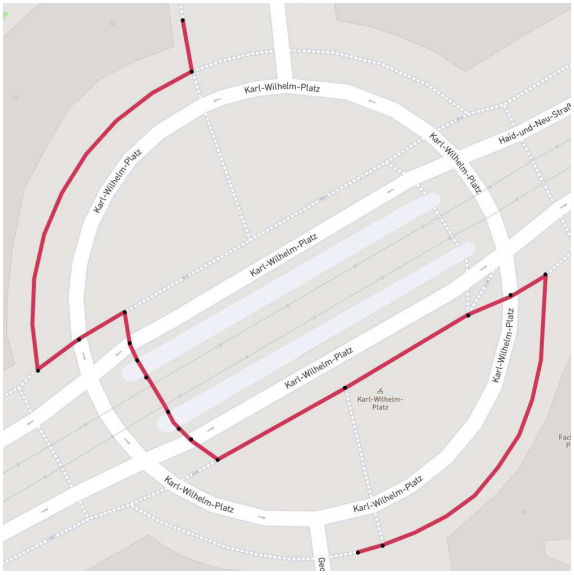
Building Facades Building facades are often used by blind and partially sighted people as an inner shoreline in urban areas. They are considered much safer than sidewalk curbs (often the outer shoreline), as they are further away from road traffic. Usually in urban areas such facades are directly adjacent to the sidewalk, especially in city-centers. Ideally, one would acquire such data from land registry offices to also use other *natural* features, e.g., sidewalk curbs or walkway boundaries, that are not (yet) part of *OSM*.

We create a k-d tree[10] of all the building facades' endpoints in a given map. Using our route as generated so far by

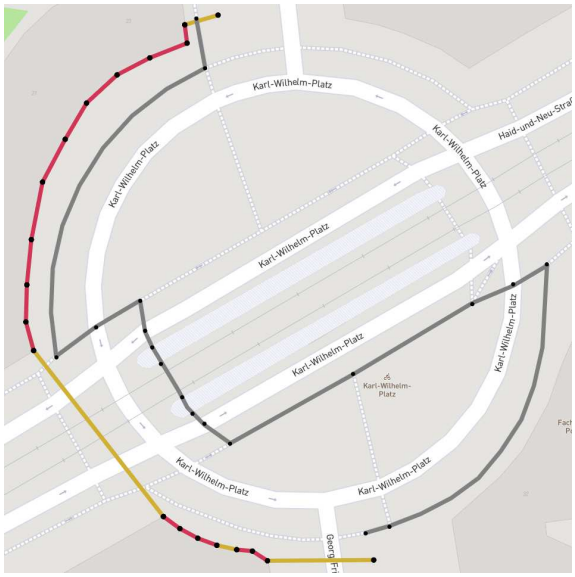
[10] A k-d tree is a space-partitioning (binary) tree. In our case, d is 2, the tree splits its space using north-south and east-west hyperplanes for each inserted point. Once created, the k-d tree allows for very efficient nearest neighbour and range search. This specific k-d tree property allows us to selectively consider only the facades in a route's (segment's) vicinity.



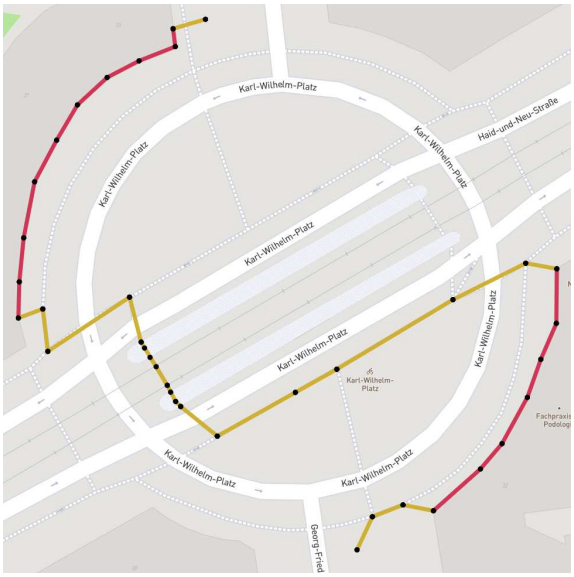
(a) Walkways



(b) Walkways and pedestrian signals



(c) Shorelines

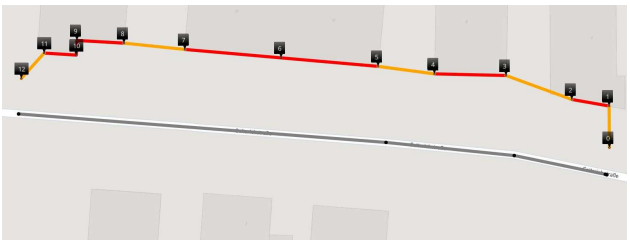


(d) Merged result

Figure 4: Various prioritization approaches in comparison: (a) prioritizes walkways over roads, (b) additionally prefers pedestrian crossings equipped with (accessible) pedestrian signals, (c) uses only facades, directly connecting them into real and virtual shorelines, (d) shows the merged result of the previous three, taking into account all restrictions and prioritizations. Please note the road crossing on the bottom of the images does not contain any kind of pedestrian signal. While the resulting merged route looks much more complicated, it actually uses only crossings that contain a pedestrian signal, and could therefore potentially be much less stressful for blind and partially sighted people.

the aforementioned steps, we calculate the nearest facades on both sides for each segment. These selected facades are then connected by a modified Dijkstra algorithm that joins adjacent facades and bridges gaps, e.g., driveways. Our algorithm also considers facades separated by street crossings by using an additional k-d tree of street sections. Furthermore, this prevents the route generation to continuously change the side of the street. Figure 4c and Figure 5 show rather simple examples for such a fine-grained route, both with real and virtual shorelines.

(a) OSRM and nearby facades



(b) Virtual and real shorelines

Figure 5: A fine-grained route following an inner shoreline: (a) highlights the considered facades (green) in proximity of the pre-calculated base route provided by *OSRM*, while (b) shows the generated shoreline based route and also differentiates between *real* shorelines (red) and *virtual* shorelines (orange). Please note how the algorithm also integrates the connection from and to the given start and end points located on the sidewalk. Figure best viewed in color.

Algorithm 1 Final Route Generation[11]

```
1:  d := {0, ∞, ..., ∞}
2:  prev := {0, −1, ..., −1}
3:  q := {(p_{r=0}, 0)}
4:  while q ≠ ∅ do
5:      p_u := q.pop()
6:      for all l_i ∈ (S'_R ∪ R) do
7:          p_v := f_near(p_u, l_i)
8:          if d[u] + δ_{p_u p_v l_i} < d[v] then
9:              d[v] := d[u] + δ_{p_u p_v l_i}
10:             prev[v] := u
11:             q.push(p_v, d[v])
12:         end if
13:     end for
14: end while
```

3.4. Final Route Generation

Naturally, one will eventually encounter route sections where there is neither a building facade available, nor a pedestrian walkway. Thus, we merge the original base algorithm, the walkway and pedestrian traffic signal prioritized, as well as the shoreline based routing algorithms to create our final route. An exemplary result of this can be seen in Figure 4d. We modify the Dijkstra algorithm (Algorithm 1) and extend its cost function $\delta_{p_u p_v l_i}$:

$$\delta_{p_u p_v l_i} := \begin{cases} \mathcal{W}_S \cdot \|p_u - p_v\|_2 , & (\delta_{p_u l_i} = 0) \wedge (l_i \in \mathcal{S}) \\ \mathcal{W}_R(l_i) \cdot \|p_u - p_v\|_2 , & (\delta_{p_u l_i} = 0) \wedge (l_i \in \mathcal{R}) \\ \delta_{p_u l_i} , & otherwise \end{cases} ,$$

[11] Short explanation: (1-3) initialize cumulative $\{d\}$istance towards each node, $\{prev\}$ious node for the currently shortest connection, distance sorted priority $\{q\}$ueue (startpoint $p_{r=0}$), (4-5) while queue not empty, take closest node, (6) check all reachable shoreline or *OSM* route segments, (7) get closest facade point for p_u along l_i, (8-11) if distance to reachable node smaller than current, store and add new node to priority queue. For a more in-depth Dijkstra explanation, please refer to [8].

where $\delta_{p_u l_i}$ and $\mathcal{W}_R(l_i)$ are defined as:

$$\delta_{p_u l_i} := (1 + |\mathcal{C}_{p_u p_v}| \cdot \mathcal{W}_R) \cdot d(p_u, l_i) \quad ,$$

$$\mathcal{W}_R(l_i) := \begin{cases} \mathcal{W}_C , & informalCrossing(l_i) \\ \mathcal{W}_{PS} \cdot \mathcal{W}_R , & PedestrianSignal(l_i) \\ \mathcal{W}_{APS} \cdot \mathcal{W}_R , & APS(l_i) \\ \mathcal{W}_{APS_p} \cdot \mathcal{W}_R , & PilotTone(l_i) \\ \mathcal{W}_R , & otherwise \end{cases} ,$$

with S being all real and virtual shorelines, R representing all other route segment types found in our *Edge Expanded Graph*, $\mathcal{W}_* \in \mathbb{R}_+$ being our selected corresponding segment multipliers (weights or penalties) for all the different used route segment types (where smaller is *better*), $|\mathcal{C}_{p_u p_v}|$ as the number of streets crossed when connecting p_u with p_v, and finally $d(p_u, l_i)$ the distance from our current node to the tested one as defined by the weights in the graph. Furthermore, our selected $\mathcal{W}_*$ are chosen to satisfy:

$$\mathcal{W}_C > \mathcal{W}_{PS} > \mathcal{W}_{APS} > \mathcal{W}_{APS_p} > \mathcal{W}_R > \mathcal{W}_S > 1 \quad .$$

3.5. Directions

Our final modified algorithm exports yet another *GeoJson* file that contains, in the order they appear in the final route, information about every route segment, e.g., its length and relative direction, whether it is following a facade, crossing a driveway or street, if there is a (accessible) pedestrian traffic signal or only a zebra-crossing. For example, the numbered speech bubbles in Figure 5 now provide the following announcements: (0) *"Please turn north until you reach a building."*, (1) *"Follow the facade to the left for 8m."*, (2) *"Continue for 18m at 1 o'clock to cross a driveway."*, (3) *"Follow the facade for 16m."*, ..., (12) *"You have reached your destination."*.

	$\bar{d}$	$\bar{r}_w$	$\bar{ps}$	$\bar{c}$
R_{OSRM}	**458**	12.8	.157	.064
$R_{Walkway}$	466	**16.4**	.669	.128
R_{APS}	464	**16.4**	**.792**	**.063**

Table 1: Results on 1000 randomly generated routes for the original *OSRM* algorithm as well as our walkway and (accessible) pedestrian signals *(APS)* prioritizations. We compare the averages for: route distance in m ($\bar{d}$), percentage on pedestrian walkways ($\bar{r}_w$), number of pedestrian signals used ($\bar{ps}$) as well as the number of informal crossings ($\bar{c}$).

4. Evaluation

We evaluate our route generation algorithms in 3 separate steps: a purely randomized evaluation, public transit stations as seed points, and a small *Wizard of Oz* experiment with a blind participant. After map data pre-processing, which is performed once, route generation requires only a few seconds per route. Due to memory limitations, this is done on a routing server, but could potentially be implemented on a smartphone or other mobile device as well.

4.1. Random

To get a first idea of the route transformations that our algorithms create, we perform an evaluation of 1000 randomly selected, rather short routes (less than 1km). These were created in a largely rural area including multiple large cities, covering roughly 4000 km^2. We compare in terms of average distance, percentage of walkways, average number of pedestrian signals used per each individual route, and average number of road crossings per route without pedestrian signals, i.e., informal crossings, as determined by our previous security and safety considerations.

Walkway and Accessible Pedestrian Signals Prioritization Table 1 shows that our modifications do not increase the average length by any significant amount, so the generated routes are comparable in absolute distance, which is a nice property, as it means secure routes must not necessarily be much longer. The pedestrian walkway prioritization cannot significantly increase the percentage of walkways taken, as especially in rural areas, *OSM* often simply does not contain enough in its data. Furthermore, the pedestrian walkway prioritization already includes a lot more pedestrian signals, as such connections are more likely to be used when connecting walkways on both sides of a street, but the pedestrian signals prioritization increases this even further, as intended. Interestingly, the walkway prioritization doubles the average number of informal crossings, as it requires more roadside changes, but the pedestrian signals modification nicely compensates for that.

	$\bar{d}$	$\bar{r}_w$	$\bar{s}_r$	$\bar{s}_v$	$\bar{c}$
$R_{Shorelines}$	**139**	00.0	**24.5**	12.3	0.056
R_{Final}	162	**17.6**	20.9	**11.0**	**0.043**

Table 2: Results on 400 randomly generated routes for the (virtual) shoreline prioritization and our final algorithm. We compare the averages for: route distance in m ($\bar{d}$), percentage on pedestrian walkways ($\bar{r}_w$), percentage of real shorelines, i.e., facades, ($\bar{s}_r$) and virtual shorelines, ($\bar{s}_v$), as well as the number of informal crossings ($\bar{c}$).

Shorelines We also compare our shoreline based algorithm to our final version, which merges all preceding ones, as shown in Table 2. Here, 400 random routes in an urban setting were used, while rural areas were explicitly left out, as one cannot safely assume there to be buildings directly adjacent to the pedestrian sidewalk. Our final algorithm has a greater effect on the average distance, as it includes many more fine-grained details and actively avoids informal crossings as long as possible. The shoreline prioritized version contains zero pedestrian walkways, as it always directly connects building facades or falls back to road surfaces, while the final algorithm integrates all three respectively. As expected, the final algorithm decreases the relative shoreline amount through inclusion of pedestrian walkways and decreases the number of informal crossings.

4.2. Public Transit Stations

Random routes do not properly represent the actual, everyday movements of blind and partially sighted people. Often, especially in urban areas, routes contain the own home or workplace, and specific points of interest, e.g., businesses or public places, as start and endpoints. More importantly, especially in urban areas, larger distances are usually not covered by foot or car, but by public transport systems, such as buses and tram lines. Thus, we choose to take public transit stations as *seed* points (they are part of the *OSM* data) and create semi-random routes from these seeds. Figure 1 shows a visualized excerpt of an exemplary urban area, where 187 tram stations were used as seeding points. 10 routes were generated each, within a radius of 700m and end endpoints selected to be close to a building's facade or a pedestrian walkway, also allowing them to end in parks or other locations reachable by foot as well.

Walkway and Accessible Pedestrian Signals Prioritization Table 3 shows that for this more realistic scenario, much like for the random evaluation, the average length is not increased significantly. More importantly, the percentage of walkways taken per route almost doubles compared to the original *OSRM* algorithm, as does the number of used

	$\bar{d}$	$\bar{r}_w$	$\bar{ps}$	$\bar{ps}_a$	$\bar{ps}_p$	$\bar{c}$
R_{OSRM}	**621**	26,4	2.327	0.205	0.019	0.702
$R_{Walkway}$	654	**46,4**	**4.819**	**0.338**	0.070	1.501
R_{APS}	655	45,5	4.709	0.320	**0.814**	**0.615**

Table 3: Results on 1870 public transit station based routes for the original *OSRM*. We compare the averages for: route distance ($\bar{d}$), route percentage on pedestrian walkways ($\bar{r}_w$), number of pedestrian signals ($\bar{ps}$), *APS* with haptic or aural signals ($\bar{ps}_a$), *APS* that contain a pilot tone ($\bar{ps}_p$), and informal crossings ($\bar{c}$).

pedestrian signals. Furthermore, the pedestrian signals optimized algorithm drastically increases the number of *APS* used, while also decreasing the number of informal crossings, which results in much safer street crossings.

Shorelines We once more compare our final algorithm to the purely shoreline based one, as shown in Table 4. There is, again, only a very minor effect on the average route distance, while there's a great increase in the percentage of integrated pedestrian walkways, a slight reduction in real and virtual shorelines, as well as a decrease in informal crossings. An example of such a final generated route (and other possible intermediate modifications of our algorithm) can be seen in Figure 4.

4.3. Wizard of Oz

We evaluate our system's usefulness in a small *Wizard of Oz* experiment [7], as there is currently no *GNSS* system available to us that delivers the required precision. We use two 400m long routes, starting or ending at a public transit station, containing multiple street crossings, driveways, and a zebra crossing. Furthermore, the routes would mostly consist of real and virtual shorelines, to create the desired very fine-grained routing, but chosen to be in a location unknown to the participant. A closely following supervisor would interactively read generated directions to the participant (similar to 3.5) and ensure the participant's safety.

Knowing the location, length, and direction of the shoreline, as well the next segment, is a much appreciated feature and greatly helps in the creation of a *mental map* of one's immediate surroundings. More importantly, assistance when a shoreline ends, i.e., which direction to continue in and how far, was valued even more. On long virtual shorelines, the participant wished to have a lighthouse system, guiding him towards the next real shoreline, to prevent veering off the best path. Also knowing a crossing's type, i.e., whether it's a driveway or one has to cross a road, increased the participant's confidence and perceived safety.

Overall, this specific participant regarded the information provided by the system very highly and would definitely like to use it in the future, especially in unknown areas. Furthermore, the participant would also like to get more information about the layout of an encountered street crossing. Finally, the participant suggested that the output should be highly configurable, especially the verbosity level and type of direction announcements used.

	$\bar{d}$	$\bar{r}_w$	$\bar{s}_r$	$\bar{s}_v$	$\bar{c}$
$R_{Shorelines}$	**178**	00.0	**31.4**	7.9	.056
R_{final}	198	**22.0**	26.0	**8.2**	**.035**

Table 4: Results on 1870 public transit station based routes for the (virtual) shoreline prioritization and our final algorithm. We compare the averages for: route distance in m ($\bar{d}$), percentage on pedestrian walkways ($\bar{r}_w$), percentage of real shorelines, i.e., facades, ($\bar{s}_r$) and virtual shorelines, ($\bar{s}_v$), as well as the number of informal crossings ($\bar{c}$).

5. Conclusions and Future Work

We show that our created system is able to generate routes for blind and partially sighted people using available geospatial information. To the best of our knowledge, this is the first work to generate routes for blind and partially sighted people on such a detailed level, inspired by *natural feature* based shorelines. Our evaluation shows that our routing algorithm can create safer routes by the defined criteria: avoid informal crossings, prefer accessible pedestrian signals, integrate shorelines where possible, and thus allow a more confident and self-reliant mobility for blind and partially sighted people, especially in unknown areas.

At the time of this writing, our system cannot be used as is, as current generation *Global Navigation Satellite Systems (GNSS)* do not achieve the required accuracy. Only with the recent advent of a newer generation *GNSS*, e.g., Galileo, such fine-grained routing will become possible.

Future research includes generating improved routes, using more available geospatial information, in order to also rely on more different types of actually used shorelines types, e.g., walkway borders in the park or roadside curbs. The creation of the suggested lighthouse assistance system would be very beneficial for further evaluations. Also, personal preferences and abilities must be taken into account when developing such a system. Finally, the generated routes should be tested in a proper, much larger user study, using a real working prototype.

Acknowledgements

This work has been partially funded by the *Bundesministerium für Bildung und Forschung (BMBF)* under grant no. 16SV7609.

References

[1] D. Ahmetovic, C. Bernareggi, A. Gerino, and S. Mascetti. ZebraRecognizer: Efficient and Precise Localization of Pedestrian Crossings. In *22nd International Conference on Pattern Recognition (ICPR)*, pages 2566—2571, 2014. 2

[2] S. Andreev, J. Dibbelt, M. Nöllenburg, T. Pajor, and D. Wagner. Towards Realistic Pedestrian Route Planning. In *15th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2015)*, OpenAccess Series in Informatics (OASIcs), pages 1–15, 2015. 2

[3] J. R. Blum, M. Bouchard, and J. R. Cooperstock. Spatialized Audio Environmental Awareness for Blind Users With a Smartphone. *Mobile Networks and Applications*, 18(3):295–309, 2013. 2

[4] H. Butler, M. Daly, A. Doyle, S. Gillies, and S. Hagen. The GeoJSON Format. RFC, Internet Engineering Task Force (IETF), August 2016. 3

[5] T. Caldwell. On Finding Minimum Routes in a Network with Turn Penalties. *Commun. ACM*, 4(2):107–108, Feb. 1961. 3

[6] Á. Csapó, G. Wersényi, H. Nagy, and T. Stockman. A Survey of Assistive Technologies and Applications for Blind Users on Mobile Platforms: A Review and Foundation for Research. *Journal on Multimodal User Interfaces*, 9(4):275–286, 2015. 2

[7] N. Dahlbäck, A. Jönsson, and L. Ahrenberg. Wizard Of Oz Studies–Why and How. *Knowledge-based systems*, 6(4):258–266, 1993. 1, 8

[8] E. W. Dijkstra. A Note on Two Problems in Connexion with Graphs. *Numer. Math.*, 1(1):269–271, Dec. 1959. 1, 6

[9] M. Haklay and P. Weber. Openstreetmap: User-generated Street Maps. *IEEE Pervasive Computing*, 7(4):12–18, 2008. 3

[10] K. Hara, V. Le, and J. Froehlich. Combining Crowdsourcing and Google Street View to Identify Street-Level Accessibility Problems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 631—640, 2013. 2

[11] S. Hintz and A. Baumgartner. Automatic Extraction of Urban Road Networks from Multi-View Aerial Imagery. *ISPRS J. Photogrammetry Remote Sensing*, 58:83–98, 2003. 2

[12] V. Ivanchenko, J. Coughlan, and H. Shen. Detecting and Locating Crosswalks Using a Camera Phone. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2008. 2

[13] J. Jiang, G. Han, and J. Chen. Modelling Turning Restrictions in Traffic Networks for Vehicle Navigation System. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 34(4):106–110, 2002. 3

[14] D. Koester, B. Lunt, and R. Stiefelhagen. Zebra Crossing Detection from Aerial Imagery Across Countries. In *International Conference on Computers Helping People with Special Needs (ICCHP)*, Linz, Austria, July 2016. 2

[15] D. Luxen and C. Vetter. Real-time Routing with OpenStreetMap Data. In *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '11, pages 513–516, New York, NY, USA, 2011. ACM. 3

[16] B. Matthews, D. Hibberd, and O. Carsten. Road and Street Crossings for Blind and Partially Sighted People: The Importance of Being Certain. Technical report, Guide Dogs for the Blind Association, 2014. 1, 4

[17] A. Noulas, S. Scellato, C. Mascolo, and M. Pontil. An Empirical Study of Geographic User Activity Patterns in Foursquare. *ICWSM*, 11:70–573, 2011. 2

[18] S. A. Panëels, D. Varenne, J. R. Blum, and J. R. Cooperstock. The Walking Straight Mobile Application: Helping the Visually Impaired Avoid Veering. Georgia Institute of Technology, 2013. 2

[19] L. Quam. Road Tracking and Anomaly Detection in Aerial Imagery. In *ARPA Spring 1978 Image Understanding Workshop*, 1978. 2

[20] B. Schmitz and T. Ertl. Individualized Route Planning and Guidance Based on Map Content Transformations. In *Computers Helping People with Special Needs. 14th International Conference, ICCHP 2014, Proceedings*, pages 120–127, 2014. 2

[21] J. L. Souman, I. Frissen, M. N. Sreenivasa, and M. O. Ernst. Walking Straight Into Circles. *Current Biology*, 19(18):1538–1542, 2009. 2

[22] T. Völkel and G. Weber. RouteCheckr: Personalized Multicriteria Routing for Mobility Impaired Pedestrians. In *Proceedings of the 10th international ACM SIGACCESS conference on Computers and accessibility*, pages 185–192. ACM, 2008. 2

[23] S. Winter. Modeling Costs of Turns in Route Planning. *GeoInformatica*, 6(4):345–361, Dec 2002. 3

[24] D. Zielstra and A. Zipf. Quantitative Studies on the Data Quality of OpenStreetMap in Germany. In *Proceedings of GIScience*, 2010. 3