Mobile Navigation for Limited Mobility Users

Bettina Harriehausen-Muhlabauter

Computer Science Department, University of Applied Sciences
Darmstadt, 69121, Germany
bettina.harriehausen@h-da.de

Abstract. Modern information technology can improve life in many places and situations in our society. This includes the improvement and simplification of life for people with special needs. We have developed a mobile navigation tool called Wheel Scout, which combines modern mobile information technology, such as smartphones and their programming, with existing navigational technology, meeting the goal to serve specific needs for people with special needs. In comparison toexisting tools, we concentrate on barrier-free routes rather than barrier-free buildings and our enhancements include: (a) the marking of barriers on a chosen route, (b) an intelligent computation of a detour in case the chosen route contains barriers, (c) the customization of the app by defining your personal profile(s), (d) the opportunity to include both static as well as temporary barriers, and (e) a high degree of interactivity which enhances the app steadily.

Keywords: Mobile navigation, Calculation of barrier-free walkways, Customization based on degree of impairment, Mobile insertion of static and temporary barriers, Enhancement through interactivity.

1 Introduction

Steep ramps, stairs, bouldering and other uneven footpath surfacing are often insurmountable barriers for the mobile handicapped or wheelchair users. On the basis of the geographical data of OpenStreetMap [4], we have developed a mobile navigation app which enables mobility impaired people to navigate from A to B on a barrier-free route. In addition to commonly known features of navigation systems, we are using the points of interest (POI) feature and their geo dataset to add barriers to the maps. Therefore, according to our definition, POIs are not tourist sights, gas stations or restaurants, but rather geographical positions that mark barriers for the mobile handicapped, such as steep ramps, narrow passages, staircases or uneven surfaces, or temporary barriers, such as fallen trees or construction sites.

2 Research and Related Work

During our initial research, we started by investigating commonly used GPS navigation systems with regard to their functionalities as well as their open interfaces

V.G. Duffy (Ed.): DHM 2014, LNCS 8529, pp. 535–545, 2014.

to add individual enhancements. We found that recent developments for the computing of a route have extended the choice of transportation to pedestrians but none of the popular systems (TomTom [1], Garmin [2], Google Maps [3] or OpenStreetMap [4] currently support features for wheelchair users or handicapped people.

Fig. 1. Choice of pedestrian routing in GOOGLE maps and OpenStreetMap

Fig. 2. Icons for barriers (stairs, narrow passage, incline, temporary construction site)

Besides looking at those popular navigation systems, we investigated several wheelchair navigation and support systems and found that they all differ heavily in their functionalities as well as their product status. Excellent tools, such as WheelMap [5], focus on barrier-free buildings rather than routes, and systems with a similar focus to ours, such as Rollstuhlrouting.de [6] or EasyWheel [7] haven't left their prototype-status yet or do not cover all of the features that we have focused on. After a general research regarding related work, we concentrated our research and the following design of the system on direct contact and interviews with wheelchair users. This led us to the definition of the following barriers, which can be added to the map by the users interactively: stairs, narrow passages, ramps, and various insurmountable surfacing. We have developed self-explanatory icons for all of those barriers. In addition to statics barriers, users can also add temporary barriers (marked by a red clock symbol), such as fallen trees or temporary construction sites. Presently, these temporary POIs are being automatically removed from the map after 2 days, unless another user renews its existence.

3 Design

We are using the traffic light metaphor to mark the selected paths according to the feature of being passable or not, i.e. whether they are barrier-free or not. In our app, "passable" is either defined by a standard profile or can individually be defined by the user, as certain barriers may cause difficulties for some users but not others; i.e. several members in our testing group were able to climb staircases with several stairs, whereas others would not be able to mount low steps. In case the path is barrier-free, it is marked as a green route from S (start) to Z (goal, German: Ziel). A path marked yellow contains barriers, which may cause difficulties for certain people but not others, i.e. the user can individually decide whether or not s/he wants to select that route.

Fig. 3. Marking of routes (green: barrier-free, yellow: route includes surmountable barriers, red: route includes barriers)

Fig. 4. Display of a barrier (staircase with steps of 28cm height) along a selected route (red) and computed barrier-free detour (green)

In case the app detects a barrier along the shortest route, that route will be marked red and a barrier-free (green) alternative will be computed automatically.

Fig. 5. Wheel Scout GUI

Fig. 6. Icons (from top left to bottom right): compute a route, show map, show profile, add a profile, active wheelchair, electric wheelchair, wheelchair with companion, navigation

In all cases, the user can always select to view the barrier along the route and will be shown the details for the barrier, which includes the icon on the type of barrier as first information, but in addition also text which specifies the barrier, such as the number of stairs and the height of the stairs. In case a photo was added for the barrier before, the user can also choose to view that photo. All this information helps to decide whether or not the barrier can be surmounted or not.

The look & feel of the app is designed so it can be used without major explanations and each GUI is held very simple without overloaded functionalities. The icons we developed for the barriers (see ch. 2) and the functionalities are self-explanatory:

4 Functionalities

4.1 Customization

For users who choose not to define their individual profile, we have predefined three standard profiles (Fig.7a), one for active wheelchair users, the second one for users of electric wheelchairs, and the third one for wheelchair users who have a companion, who helps them to move.

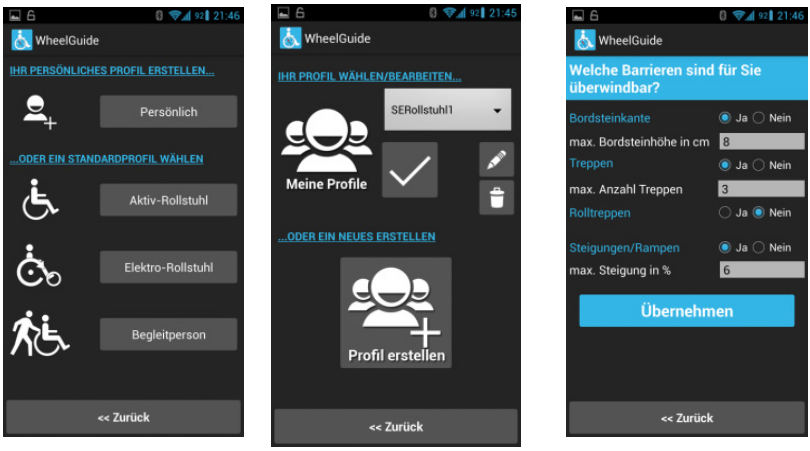

Fig. 7. (a) Selection of a profile, (b) Definition of an addition profile, (c) Customization of individual profile, here: Which barriers are surmountable for you? Curbstone with its height, stairs with the number of steps, escalators, ramps and their degree of incline

Beyond these standard profiles, each user can define one or more individual profile (Fig.7b). Our users' feedback has shown that not only do the individual profiles vary on a daily basis, but the range of what is regarded as a barrier can vary tremendously among wheelchair users of the same type of wheelchair. Some users were able to climb staircases with a relative high number of steps, others would already consider low steps unsurmountable. Our app guides the user through the definition of an individual profile (Fig.7c).

4.2 The Marking of Barriers

We distinguish between two types of barriers: permanent and temporary barriers. Examples for permanent barriers are staircases and ramps, temporary barriers can be temporary construction sites which make a passage unpassable, fallen trees or surface conditions due to temporary weather conditions, e.g. icy roads. Once a temporary barrier is included into the map, it will be automatically removed after two days, unless it is re-entered by the same or another user.

Permanent Barriers. There are three possibilities for permanent barriers to appear in the app:

1. they are predefined by OSM,
2. we include them into the OSM maps before distribution by using the POI feature and current GPS position to include a "point of interest", i.e. one of our barriers, or
3. the second option is performed by one of our users interactively, i.e. the app and its correctness of barriers will grow by using it.

Temporary Barriers. There are two possibilities for temporary barriers to appear in the app:

1. they are interactively included by users when they see or experience a temporary barrier, or
2. they are included by the road traffic licensing departments of cities which decide to use the app as a means to receive and distribute data about temporary barriers.[1]

4.3 Interactivity

The marking of barriers, and thus the correctness and completeness of the app, depends to a high degree on the interactivity of the users. When experiencing or seeing a permanent or temporary barrier that is not included in the app yet, the user can include it by using the GUI shown in Fig.9, which asks to report and include a barrier (German: Barriere melden), to specify the type of barrier (German: Barrierenart), here: ramp, to specify more detailed information on the barrier (German: Steigung in %), here: the degree of incline, and to optionally take a picture of the barrier, so future users can decide upon the picture whether or not the barrier will cause difficulties. In case of a ramp, the user may decide, and later see, the image in Fig.8.

After the new barrier is added to the app, its position is marked on the map with the corresponding icon. Upon selection (touch or roll-over) the additional textual information is given, as well as a thumbnail image of the barrier, in case a picture was previously added. In that case, the image can be enlarged.

[1] Note: In Germany, temporary barriers, such as temporary construction sites, have to be reported to the city's road traffic licensing department before the barrier is set up. But of course, barriers such as fallen trees cannot be reported beforehand.

Fig. 8. Reporting and adding a new barrier-information

Fig. 9. Image of a ramp taken by a user and added to the POI-barrier

Fig. 10. Information that is displayed for each barrier upon request (here: staircase with the number and height of stairs, as well as image)

4.4 Computation of Individual, Barrier-Free Routes

The basis of our route computation is provided by the free navigation tool OpenStreetMap (OSM), which provides map data for locations worldwide. Independent of Wheel Guides functionalities, users can use OSM to compute an individually chosen route, depending on their means of transportation (e.g. by car, by bicycle, by foot). As Wheel Guide users will never choose highways or big country roads without footpaths along the side as their routes, those are blocked in our algorithm. We use OSM's information on barriers as well as our added barriers to computer the shortest possible path for each user depending on his profile. In case an unsurmountable barrier is detected on the shortest path, that path will be marked red, the barriers will be displayed and a barrier-free detour will be computed and shown as a green route. In case barriers are detected which may cause difficulty, but the user should decide whether or not they can be handled, the routes will be displayed in yellow color and the barriers are displayed with all information that is available in our database.

5 Implementation and Technology

5.1 HTML5

In a first step, we started developing the app for Android platforms, but soon found that this was an unacceptable limitation, as the mobile market is much too diverse and dominated by at least 4 "key players": Android phones, Apple's iPhones with iOS, RIM's Blackberry and Windows Phones. And apart from the mobile market, we also wanted to offer our navigation tool to users without smartphones and to those in general who want to compute their routes from their homes. Therefore, we have chosen to develop an HTML5 website which makes it possible to inform users without smartphones about barriers and to plan routes from home. In addition, it saved us development time, which we would have had to invest if we had chosen to develop the app for all available platforms natively. By having chosen HTML5, all users can access our app, even if their smartphone is not natively supported, as they can use the browser alternative instead. This way, the web version of our app can be used via browsers on hardware which we do not explicitly support.

5.2 Open Street Map

We have chosen OpenStreetMap (OSM) [4] as the underlying map, as its interfaces enable us to add data to the existing maps, i.e. our barriers as "points of interest", and we can access its data, which enables us to extract the information that we need to compute ideal routes for our individual wheelchair user. The data that we extract from OSM includes ways and their nodes, as well as all barriers, such as steps, inclines, curbstones and the nature of the ground, such as soil conditions. All these data are needed to check which ways and nodes exist and which ones are passable according to the user's profile. The ways and nodes include feature-value pairs, called tags. Examples of such tags are: `surface=cobblestone`, `surface=grass`, `smoothness=intermediate`, `incline=*`, and `width=*`, including the gradient of the incline in percent or the width of a passage in meters[2]. The tags are accessed via the Overpass API [8], which returns custom selected parts of the OSM map data. It acts as a database over the web: the client sends a query to the API and gets back the data set that corresponds to the query [9].

An XML file is being returned which is loaded by the server and used in the next processing step in which the single nodes and ways from the file are translated into Java objects which contain the ID of the objects and their information about barriers as a key-value-pattern. An example for an entry for a staircase could be highway[3] = steps and step_count = 5 for 5 steps. When a user sends a route computation query,

[2] The complete list of tags we use from OSM can be looked up here: `http://wiki.openstreetmap.org/wiki/DE:Rollstuhlfahrer-Routing#Tags_f.C3.Bcrs_Routing`

[3] The OSM attribute *highway* is used for any way that is developed, as opposed to dirt tracks or trails.

these data are compared with the user's profile by the route computation algorithm in order to check which ways and nodes are passable or not.

Fig. 11. Marking of the position based on the IP address only

5.3 GPS

The entire computation of the routes, the detours and the barriers is based on GPS data which is provided by HTML5's geolocation functions, which include functions for error handling, the availability of the location functions, and the query of the position. When the latter is called from the mobile device, the current GPS data is sent. In case the app is used via a browser, the position is estimated by means of the device's IP address and the possibly existing WLAN signal. In cases where only the IP address is known, the accuracy of the position is not very exact and can only be shown for a general region, e.g. metropolitan area of a city, meaning an accuracy of approximately 25-150 km.

In cases where we have a WLAN signal known by Google, an accuracy of approximately 0.5 m is theoretically possible. Our experience and tests have shown that the accuracy in a large city is usually between 30 and 75 m. But because we only need to access these possibilities when no GPS signal is available, this scenario is limited to stationary computers in residential homes. In these cases a vague positioning doesn't cause problems, as the user doesn't need to find or locate his position on a map, as he knows where he lives, but it's rather used to simplify the use of the map.

When using the web-version of Wheel Guide, the browser will ask for permission, each time the position is being computed. With the mobile version this step happens automatically, as the app will be permanently authorized to access the GPS function upon download.

5.4 Computation of Routes and Barrier Handling

The most prominent feature of our app is the handling of barriers and the resulting computation and drawing of a barrier-free route. In order to check whether the chosen route contains barriers, we compare the list of tags and values which exist for each node and way with the individual profile of the user (shown on the left of Fig. 13). This will tell us whether the route is passable and barrier-free (drawn as a green route) or not.

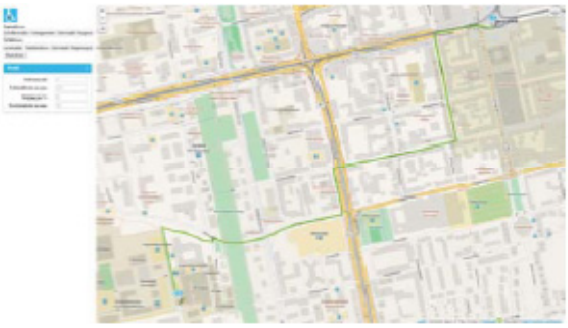

Fig. 12. Computing and drawing a route based on a profile which allows 1 step (German: Stufenanzahl), widths of 735 cm (German: Rollstuhlbreite in mm), an incline gradient of 6% (German: Steigung in %), and curb heights of 30 mm (German: Bordsteinhöhe in mm

We are using the Dijkstra-algorithm [10] to compute the optimal route for our users. This algorithm computes the shortest path for a given start-node and one (or more) target-nodes, in our case the goal or location the user wants to travel to.

In OSM and our database an intersection is called node and a street or path is called way.

First, the start- and target-nodes are determined by selecting the nodes from our database which are closest to the selected start- and target-nodes. Starting from these (database) nodes, the algorithm will check for the next nearest nodes. These will be connected by ways. In case the connecting way is impassable for wheelchair users due to barriers, this node will be ignored. For all nodes that can be reached on a barrier-free way, the distance between the nodes will be computed and stored for further computation.

Once the distances between the start node and all reachable nodes are stored in the reachable nodes, the start node will be marked off as „visited“ and will be ignored until the goal node is reached. The following steps will then be processed in this order:

1. Look for the node with the shortest distance and which hasn’t been visited yet.
2. Compute the distances to all reachable nodes and record the distance in case no value has been entered yet or the distance is shorter than the previous value.
3. Return to step 1. until the node with the lowest value and which hasn’t been visited yet, is the target-node.

This process will be repeated twice in order to compute a green and a yellow or red path. After first trying to find a green path, the process will be repeated for a yellow or red route. The yellow route will include the user’s profile data for yellow routes, i.e. barriers, which are surmountable for some users or under certain circumstances. In case barriers are detected along the chosen route, they will be buffered in the data for that route in order to be retrieved and included in the app later upon request of the user. At that stage, the user can decide individually, whether the barriers are surmountable for him or not. When computing and drawing a red route we do not include profile data, as those routes are not surmountable under any circumstances and the app will always draw the shortest route.

```
public void pathDraw() {

    StrictMode.ThreadPolicy policy = new
    StrictMode.ThreadPolicy.Builder().permitAll().build();
    StrictMode.setThreadPolicy(policy);


    road = new Road(pRouteGet());
    PathOverlay pathOverlay = RoadManager.buildRoadOverlay(road,
    mapView.getContext());

    pathCalculate();

    switch (pType.charAt(0)) {
        case 'd':
            pathOverlay.setColor(Color.GREEN);
            break;
        case 'Y':
            pathOverlay.setColor(Color.YELLOW);
            break;
        case 'R':
            pathOverlay.setColor(Color.RED);
            break;
    }
    Paint pPaint = pathOverlay.getPaint();
    pPaint.setStrokeWidth((float) (pPaint.getStrokeWidth() + 2.0));
    pPaint.setAntiAlias(true);
    ;
    pathOverlay.setPaint(pPaint);


    mapView.getOverlays().add(pathOverlay);
    mapView.invalidate();
```

Fig. 13. Pathdraw function – drawing routes in different colors according to the computed passability for the individual user

Depending on the result of this computation process, the function *pathdraw* [see Fig. 13] will draw the computed route in the appropriate color:

6 Testing

During the entire process of the design and development, we worked closely together with a group of wheelchair users. Their constant feedback enabled us to find bugs and optimize the app appropriately.

Fig. 14. Wheelchair users testing the Wheel Guide app

7 Conclusions and Next Steps

The initial feedback from users after the app was launched showed us that we were on the right track. The community of Wheelchair users responded very positively and together we have defined more functionalities, which we have started to build into Wheel Scout. These include: barrier-free toilets along the routes and specific locations (e.g. pharmacies, bakeries, banks), which the user may individually choose to be included and shown along the chosen route.

References

1. TomTom, TomTom Navigation app für Android (2014), http://www.tomtom.com/de_de/products/car-navigation/tomtom-navigation-for-android/ (last access June 02, 2014)
2. Garmin, Garmin Straßennavigation (2014), http://www.garmin.com/de-DE/explore/ontheroad (last access June 02, 2014)
3. Google Maps, Google Maps für Android, http://www.google.de/mobile/maps/ (last access June 06, 2014)
4. OpenStreetMap, http://www.openstreetmap.org/ (last access August 6, 2013)
5. Wheelmap, Wheelmap project, http://wheelmap.org/ (last access August 6, 2013)
6. Rollstuhlrouting, Rollstuhlrouting project (2013), http://www.rollstuhlrouting.de/ (last access August 6, 2013)
7. Menkens, C., et al.: EasyWheel - A Mobile Social Navigation and Support System for Wheelchair Users. In: Proceedings 2011 Eighth International Conference on Information Technology: New Generations, Las Vegas, Nevada USA, April 11-13 (2011)
8. Overpass API, http://overpass-api.de/ (last access December 6, 2013)
9. Overpass API, http://wiki.openstreetmap.org/wiki/Overpass_API (last access December 6, 2013)
10. Dijkstra Algorithm last access (2013), http://en.wikipedia.org/wiki/Dijkstra%27s_algorithm (December 6, 2013)