D3.2 Interface design prototypes and/or mock ups

Alce, Günter; Hermodsson, Klas; Lasorsa, Yohan; Liodenot, David; Michel, Thibaud; Razafimahazo, Mathieu; Lemordant, Jacques; Chippendale, Paul

2013

SEVENTH FRAMEWORK PROGRAMME

FP7-ICT-2011-1.5 Networked Media and Search Systems
b) End-to-end Immersive and Interactive Media Technologies

Specific Targeted Research Project

VENTURI

(FP7-288238)

immersiVe ENhancemenT of User-woRld Interactions

[D3.2 Interface design prototypes and/or mock ups]

Due date of deliverable: [31-01-2013]
Actual submission date: [29-01-2013]

Start date of project: 01-10-2011
Duration: 36 months

Summary of the document

Document Code:	D3.2 Interface design prototypes and/or mock ups – v0.4
Last modification: State:	29/01/2013 Quality checked
Participant Partner(s): Editor & Authors (alphabetically): Fragment:	Sony Mobile Communications, INRIA Editor: Jacques Lemordant Authors: Günter Alce (SONY), Klas Hermodsson (SONY), Yohan Lasorsa (INRIA), David Liodenot (INRIA), Thibaud Michel (INRIA), Mathieu Razafi-mahazo (INRIA), Paul Chippendale (FBK) Yes/No
Audience:	☒ public ☐ restricted ☐ internal
Abstract:	This document is deliverable D3.2 "Interface design prototypes and/or mock ups" and presents tools and prototypes to show new interface and interaction design.
Keywords:	Wizard of Oz, user interface, interaction design, navigation audio beacon, HMD, head tracker, headphone controls, QR code
References:	Refer to the corresponding section at the end of the deliverable

Document Control Page

Version number	V0.4
Date	29/01/2013
Modified by	Paul Chippendale
Comments	Ready for Quality checks
Status	☐ draft ☒ WP leader accepted ☒ Technical coordinator accepted ☒ Project coordinator accepted
Action requested	☐ to be revised by partners involved in the preparation of the deliverable ☐ for approval of the WP leader ☐ for approval of the technical coordinator ☐ for approval of the project coordinator Deadline for action: 29/01/2013

Change history

Version number	***Date***	***Changed by***	***Changes made***
0.1	28/12/2012	G. Alce	Preliminary version
0.2	15/01/2013	D. Liodenot	Add sections 4 to 6
0.3	27/01/2013	D. Liodenot	Add introduction and conclusion
0.4	29/01/2013	P. Chippendale	Quality check

Table of Contents

Executive Summary

Scope

This document provides the deliverable contents related to the T3.2 Interface and Interaction Design.

Audience

This deliverable is public.

Summary

In this report, the objective is to provide interface and interaction design, mainly based on new audio-visual technologies to improve user experience.

Structure

This deliverable is structured as follows: Section 1 is an introduction. Section 2 describes the Wizard of Oz tool, which enables the testing of novel design ideas. Section 3 considers IXE, a mobile prototype enabling indoor and outdoor navigation solely based on the Inertial Measurement Unit (IMU) and OSM network with audio instructions. In Section 4, we present prototypes to create OSM documents. Finally, in Section 5 we present PDRTrack, a localization test application designed to test the Pedestrian Dead-Reckoning (PDR) algorithms.

1 Introduction

In this report, we explore the opportunities provided by new audio-visual technologies (HMD, headphones with head-trackers, SmartWatch). We also deal with indoor and outdoor localisation and navigation based on the Inertial Measurement Unit (IMU) or Ant+ sensors. Different tools and prototypes using these technologies are presented to show new interface design, interaction design, auditory display design and navigation audio beacon.

2 Wizard of Oz

2.1 Research material

The project started with a descriptive approach and tried to describe the project by asking questions such as, where are we, what do we want to do, what techniques are available, and how can these techniques be used or combined. Therefore hypotheses were formulated inspired from VENTURI's three use cases: gaming, indoor assistant and outdoor tourism. The following six hypotheses were posed:

H1: Using AR visor for AR is better than using phone or tablet for AR

H2: New application paradigm which dynamically filters applications is better than the current application metaphor

H3: The AR visor gives an added value in everyday life to an extent that the user wants to keep them on

H4: Interaction redundancy results in low intrusion

H5: Adjustable user interface elements that can change colour and the placement of the user interface would be better than non adjustable user interface

H6: To use same type of interactions for virtual object interaction as with real objects is better than indirect screen/pointer interaction

The discussions after the listed hypotheses were about which concept frameworks are most suitable for designing and analysing these hypotheses. Example of conceptual frameworks which were considered are Distributed Cognition, Grounded Theory and Activity Theory. We decided to use Activity Theory and outlined a work flow process (Figure 2.1).

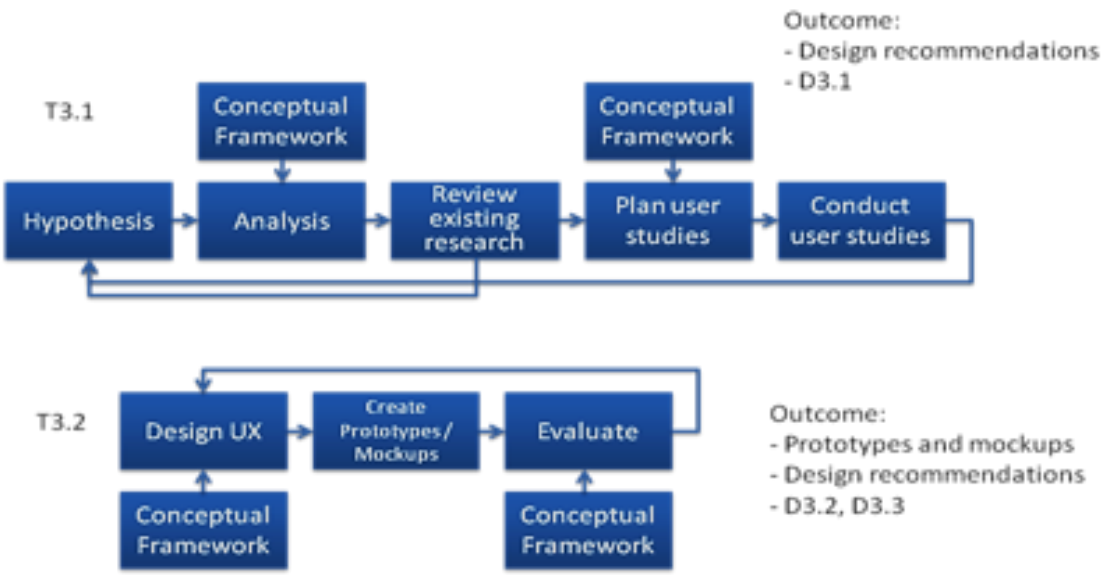


FIGURE 2.1 THE ITERATIVE DESIGN PROCESS

Each hypotheses was analysed, information was gathered and explained. Furthermore suggestions on how to collect evidence was listed and also different design inputs were suggested. For instance for the first hypothesis we started with listing positive and negative claims (Table 2.1) of using AR visor, phone and tablet.

TABLE 2.1 POSITIVE AND NEGATIVE CLAIMS OF AR GLASSES VS. HANDHELD DEVICE (POSITIVE IN ITALIC)

Claims	**AR visor**	**Phone**	**Tablet**
Field of view	*Full view*	Keyhole	Keyhole
Always connected	Have to put on	Slip out of pocket	In a bag
Power usage	Battery	Battery	Battery
View	*See through*	Camera image	Camera image
Holding the phone in front of you	*N/A*	Physically tiring	Physically tiring
Social acceptance	*Unobtrusive*	Intrusive	Intrusive

From research material, we identified that to be able to test the design ideas a tool was needed. We decided to develop a tool which could be used to simulate non-complete systems. Thus the Wizard of Oz tool was developed. The research material mentioned above is presented in the appendix.

The Wizard of Oz tool will be useful during the whole project; for UCY2 we will use it to investigate how the user should be guided in an indoor area. Several ways of navigation possibilities will be tested, including: audio guiding, visual guiding and a combination of audio and visual navigation. Another interesting area which will be investigated is how the user will be notified when a target product is found e.g. sound, vibration and graphical animations. Furthermore, different notifications will be simulated to see how intrusive they are, but also to investigate how users want to read or remove notifications in different situations.

2.2 Wizard of Oz tool

2.2.1 Introduction

When conducting user studies of a system which is in its early development, it is essential to be able to simulate missing 'parts' of the system to collect valuable feedback from potential users. A Wizard of Oz (WOZ) tool is one way of doing it.

WOZ testing is a well known method and is used for testing a non-complete system on users, where a human wizard acts as the system to fill in the missing gap. The method was initially developed by J.F. Kelley in 1983 to develop a natural language application [1]. The task of the wizard varies between different implementations [2]. In one end we have the supervisor who observes the system and acts as a safeguard and if the need arises intervenes. On the other side of the spectrum we have the controller. The controller takes the role of a missing part of the system or the entire system and emulates it. The role of the wizard can also change depending on the phase of the development. When more and more functionality is added the role of the wizard goes from being a controller to a supervisor.

WOZ testing is a powerful tool for uncovering design ideas in limited evolved technologies, especially for systems performing in physical environments, since the designers are less constrained by technical specifications [4] [3]. It is very important that a WOZ interface is supporting the wizard in performing their assigned tasks by observing the user, observing the environment, simulating sensors and/or controlling content [3]. The tool should be useful to the dedicated wizard operator and could, if possible, have automated functionality. S. Dow

et al. opinion is that WOZ testing is "leading to more frequent testing of design ideas and, hopefully, to better end-user experiences".

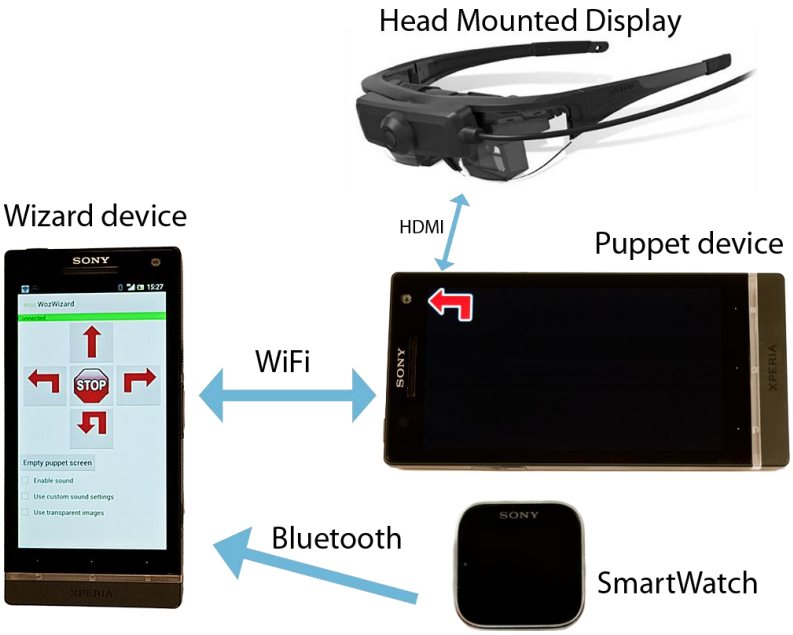
2.2.2 Setup

The WOZ tool consists of two android devices communicating with each other over WiFi (Figure 2.2). One device acts as the wizard controlled by the test leader and the other one is the puppet used by test person. The wizard application can control the user interface on the puppet device. The main features which was identified as necessary includes, present media such as image, video and sound, easy navigation and location based triggering, capability to log test results and visual feedback and integrate Sony SmartWatch [5] for interaction possibilities. Wizard device can be used as a remote control and is as already mentioned useful to test new user interface ideas before the actual application is developed.

The wizard and puppet applications communicate over WiFi using a proprietary protocol.

Android versions supported:

- Wizard: 3.2 and up
- Puppet: 2.3.3 and up



. FIGURE 2.2 SHOWS WIZARD OF OZ TOOL SETUP

2.2.3 Connection

The WOZ tool communicates via IP over WiFi. Both TCP and UDP are used in the communication. Network communication is handled through two separate services, one using TCP and the other UDP. UDP is only used to transfer the camera feed from the puppet device to the Wizard device. The other service, using TCP, is used for all other communication. The reason UDP is used for the camera feed is that speed is more important than making sure all packages are received.

All communication except for the camera feed uses a protocol generated by protocol buffers over a TCP connection [6]. Since we use wireless communication it is hard to guarantee a stable connection. Instead of trying to guarantee a stable connection, focus was put on being able to recover in case of an unwanted drop in connection. Another negative side effect of a trembling connection is that the wizard cannot really be sure that the connection is up at a given time. To counteract this, the wizard device will show its connection status on top of the screen (Figure 2.4). To clarify whether or not the connection is established the field will change color. If the

devices are connected the field will be green and red otherwise. To establish whether or not a connection is up the wizard device will send a ping every two seconds. If no answer has been received before the next ping, the wizard will consider itself as disconnected. The puppet device does not send out its own pings, only answering the ones it get. If it has not received any pings in a given timeframe it will consider itself disconnected and starts to initiate a new connection.

2.2.4 Wizard

The wizard contains most of the functionalities and can be used to control what is shown on the puppet device. Different views were developed for different situations, example of views which the wizard can enter includes navigation view, camera view etc. The views are presented in more detail in section "Features". Additional example of functionality running on the wizard is the SmartWatch which is connected with the wizard device but worn by the test person. The wizard device triggers events on the puppet when it has received events from the SmartWatch.

2.2.5 Puppet

The user interface of the puppet is designed to not have too much point and click interaction, other than the SmartWatch which is in-directly interaction through the wizard as mentioned in the previous section. The only form of point and click interaction is a menu that contains two elements, one button to close the program and one button to enter the IP address of the wizard device. Other than that it is a black background that is replaced by what the wizard wants the puppet to see (Figure 2.2). The reason for using black background is that when using HMDs with optical see-through black is transparent. There is also support to show navigation arrows on a video-see through device (Figure 2.3).

FIGURE 2.3 AN IMAGE DISPLAYED OVER THE CAMERA PREVIEW ON THE PUPPET DEVICE (VIDEO-SEE THROUGH)

The puppet device can also, if supported by the device, read out text strings, using the Android text to speech library.

2.2.6 Features

This section will describe the implemented features accessible by the wizard and also show the view of each feature.

Filebrowser

In the filebrowser view, it is possible to list all files which can be of interest to show or play to the test person on the puppet device (Figure 2.4).

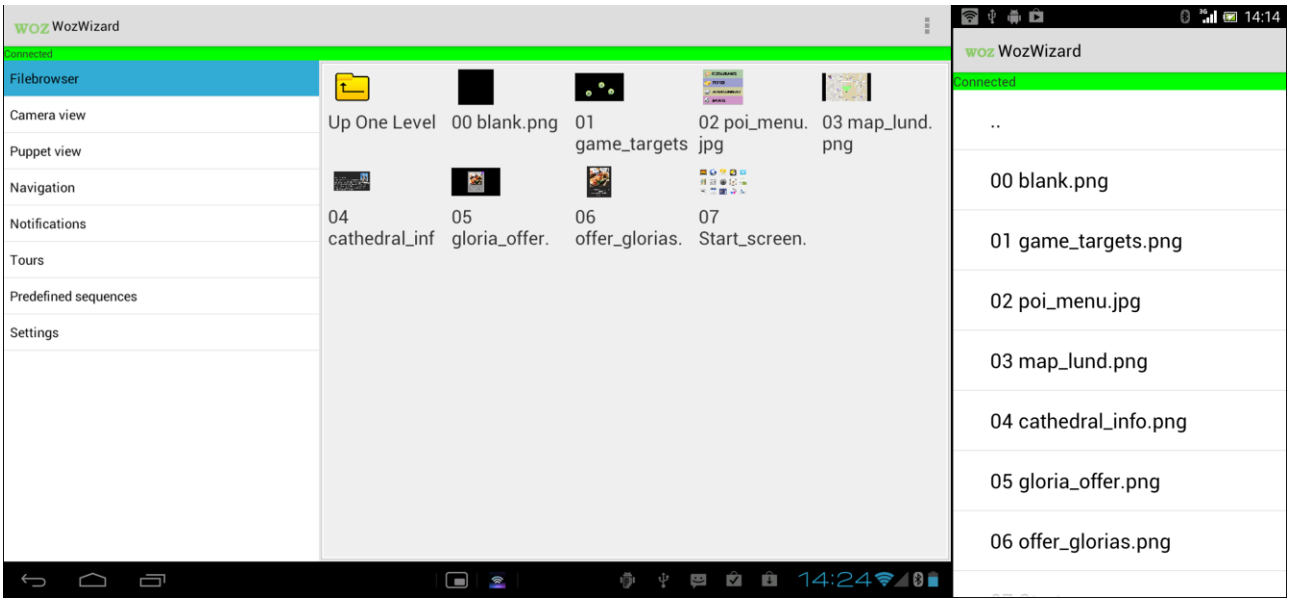


FIGURE 2.4 SCREENSHOT FROM THE FILEBROWSER VIEW (LEFT SIDE: SONY TABLET S, RIGHT SIDE: SAME VIEW ON A SONY XPERIA S)

Camera

From this camera view (Figure 2.5) the wizard can start the camera on the puppet device and also request the puppet device to send the camera feed. It is also possible to set the puppet to take pictures automatically within a time interval which also is set from the wizard. It should be noted that it is not necessary to display the camera view on the puppet device it can be started in the background without the test person knowing it.

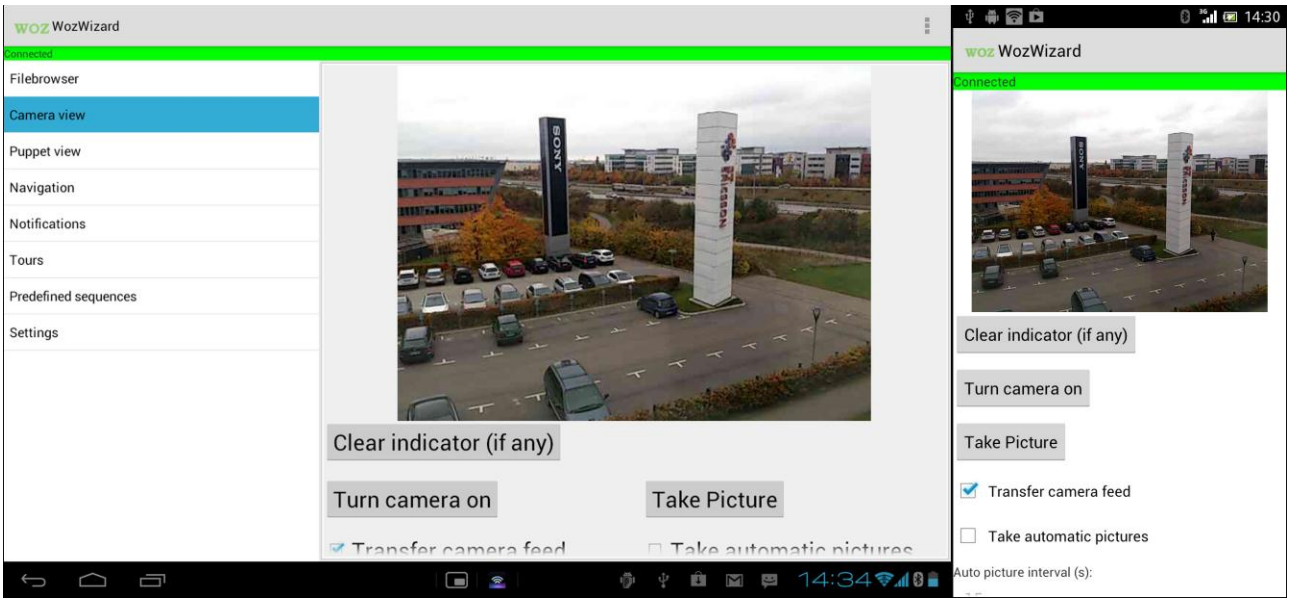


FIGURE 2.5 SCREENSHOT FROM THE CAMERA VIEW (LEFT SIDE: SONY TABLET S, RIGHT SIDE: SAME VIEW ON A SONY XPERIA S)

Puppet

This feature allows the wizard to see what is displayed on the puppet device.

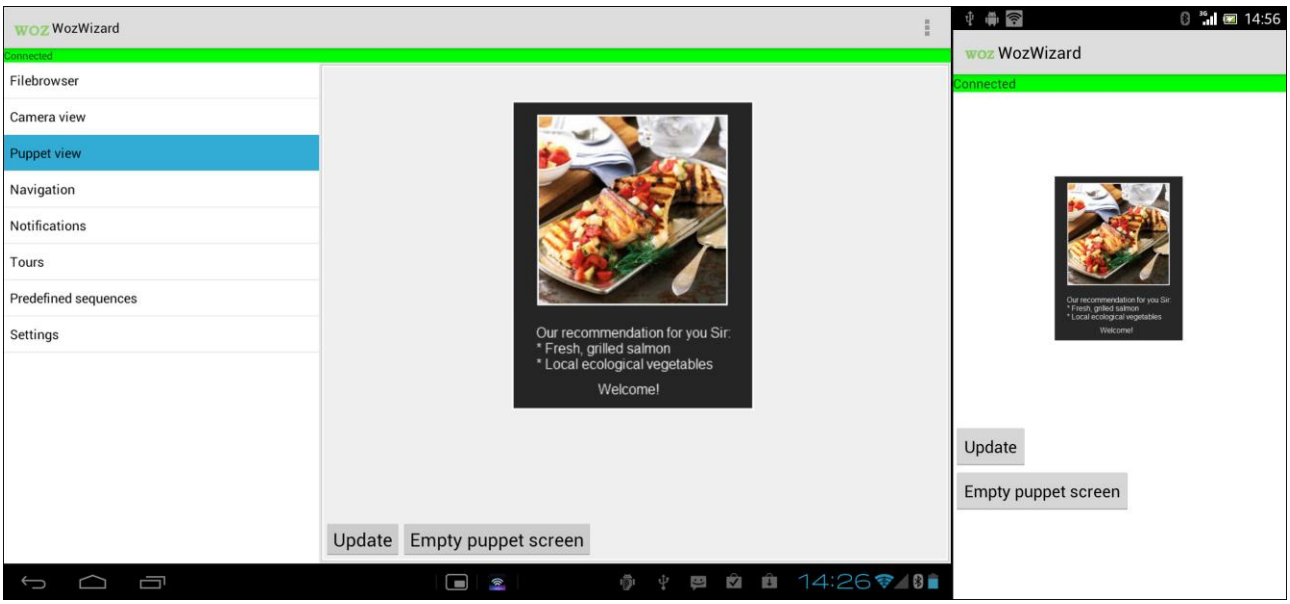


FIGURE 2.6 SCREENSHOT FROM THE PUPPET VIEW (LEFT SIDE: SONY TABLET S, RIGHT SIDE: SAME VIEW ON A SONY XPERIA S)

Navigation

The purpose with this feature is to fake navigation. It can be used both for indoor and outdoor navigation. It is also possible for the wizard to enable/disable audio navigation together with the visual navigation (Figure 2.7). Additionally it is also possible to customize the audio navigation since Android Text-to-Speech engine is used.

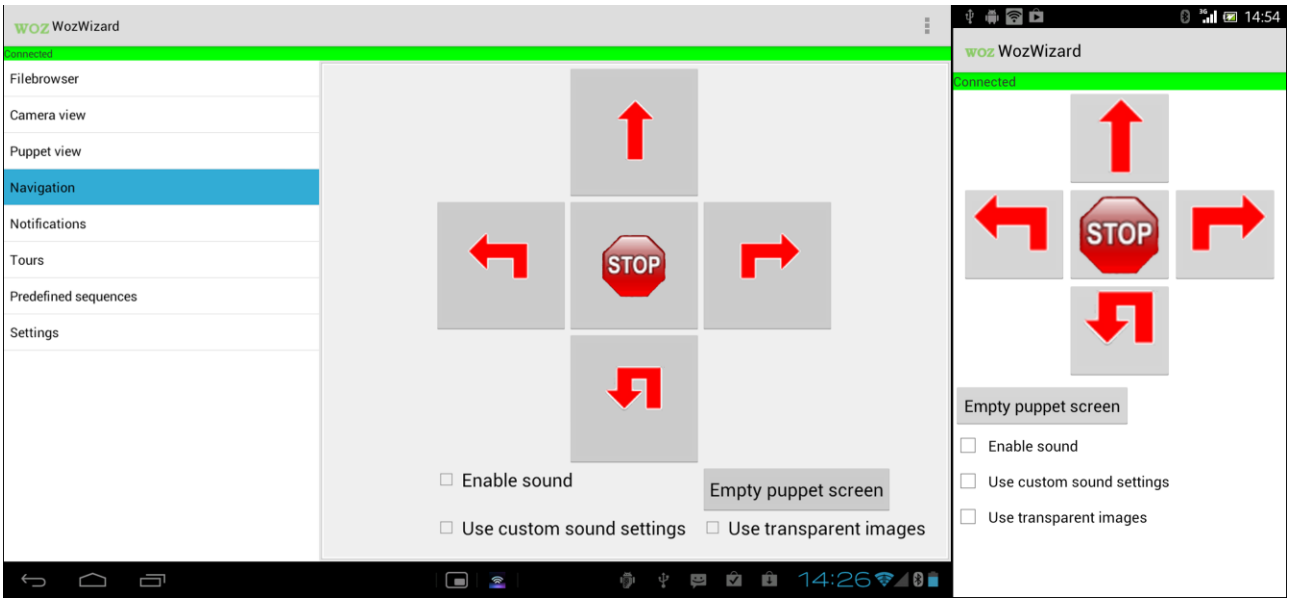


FIGURE 2.7 SCREENSHOT FROM THE NAVIGATION VIEW (LEFT SIDE: SONY TABLET S, RIGHT SIDE: SAME VIEW ON A SONY XPERIA S)

Notification

The wizard can choose to send default notifications but the wizard also have the possibility to create and send different simulated notifications on the fly such as Facebook, Twitter, G-Talk, Alarm, SMS etc. Further it is also possible to get the notification messages to be read out on the puppet device.

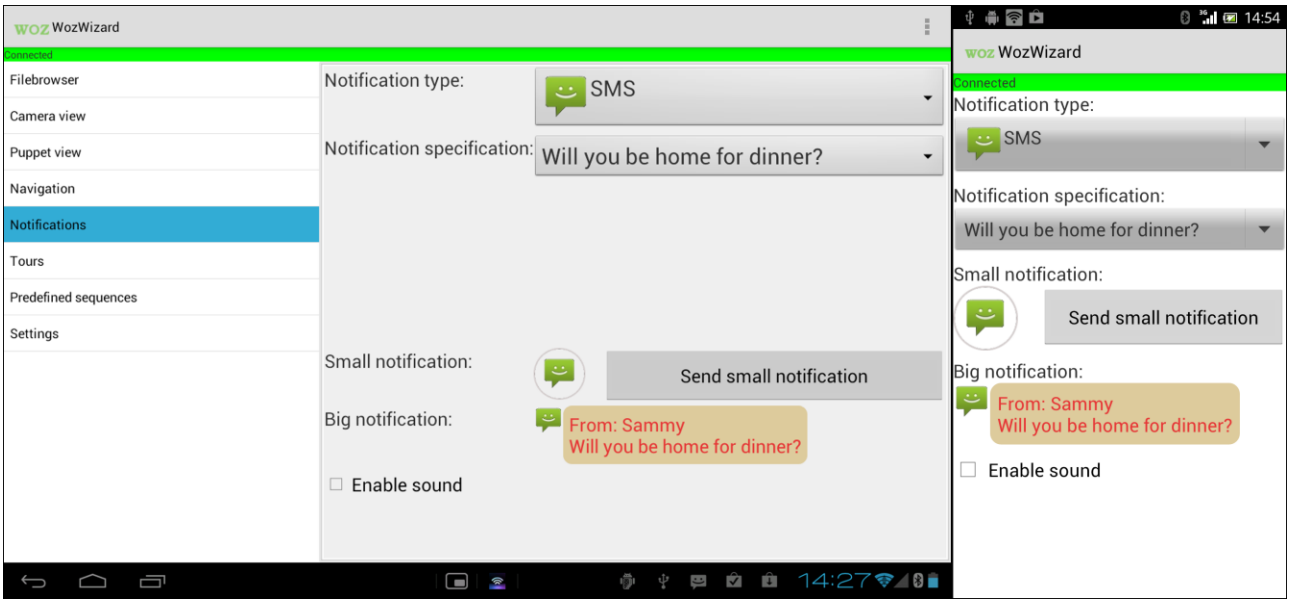


FIGURE 2.8 SCREENSHOT FROM THE NOTIFICATION VIEW (LEFT SIDE: SONY TABLET S, RIGHT SIDE: SAME VIEW ON A SONY XPERIA S)

Tours

With the tour feature the wizard can trigger different actions on different locations. The idea is to give the wizard the opportunity to create a tour with the tool. The wizard can walk around and choose to set different actions e.g. show an image in different locations presented on a map (Figure 2.9). Further the wizard can start the tour which will be running in the background and still have the possibility to use the other features.

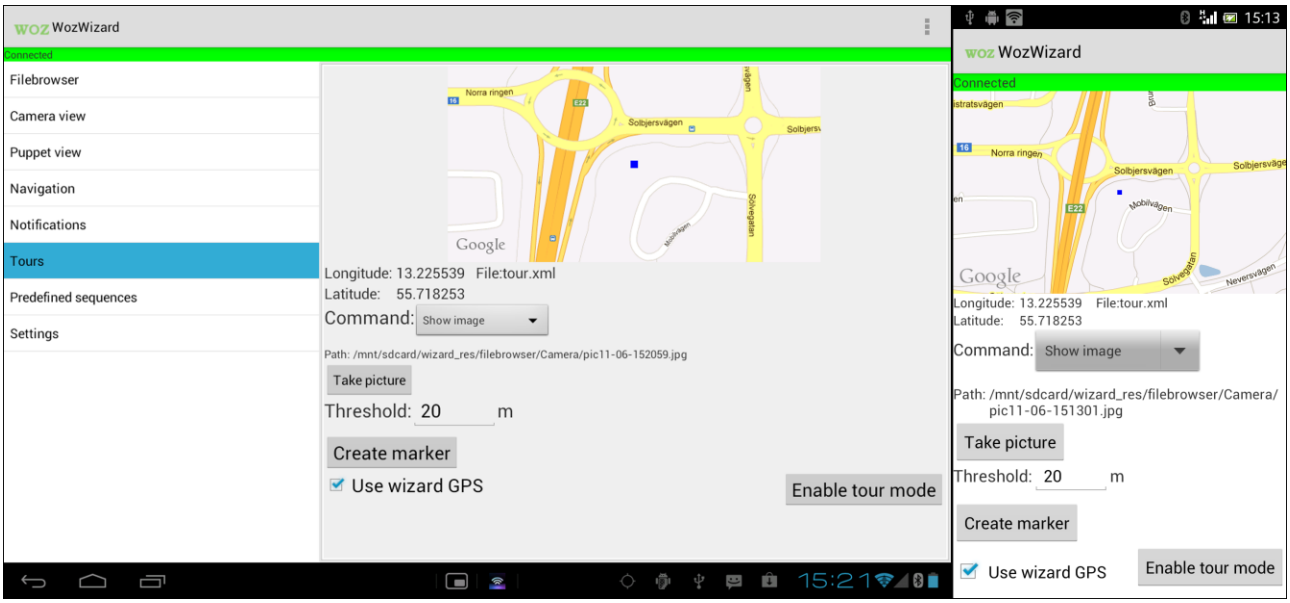


FIGURE 2.9 SCREENSHOT FROM THE TOUR VIEW (LEFT SIDE: SONY TABLET S, RIGHT SIDE: SAME VIEW ON A SONY XPERIA S)

Predefined sequence

This feature helps to predefine a user study. The idea is to list all commands and simply click through without needing to switch views.

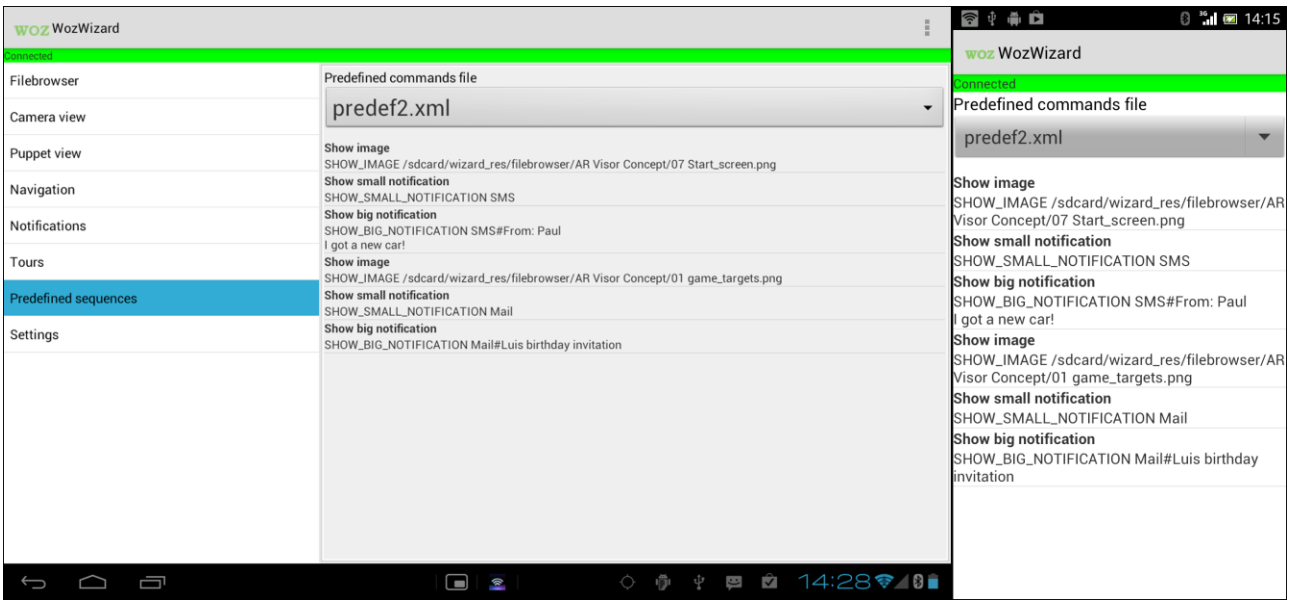


FIGURE 2.10 SCREENSHOT FROM THE PREDEFINED VIEW (LEFT SIDE: SONY TABLET S, RIGHT SIDE: SAME VIEW ON A SONY XPERIA S)

Log

Both the wizard and puppet applications have support for logging the activities. The logs are saved in the SD-card. All entries have a timestamp and if the position is known latitude and longitude are also saved. Examples of events that create entries are if a connection has been established, if a command has been received or sent and in some places error messages.

3 IXE: Interactive eXtensible Engine

3.1 Introduction

IXE is a mobile application allowing indoor and outdoor navigation only based on the Inertial Measurement Unit (IMU) described in section 5 and not on GPS. This application enables users to test navigation on any well-formatted OSM network with audio instructions. The embedded router in IXE provides the shortest path between your current location and a chosen destination in OSM format. As a consequence the generated route can be directly edited in any OSM authoring software such as JOSM or in the mobile OSM route editor described in section 4. Figure 3.1 introduces IXE independent modules from algorithm level (IMU and routing) to cross-modal user interactions (audio, button and so on) and interfaces. This mobile application was tested by visually impaired people whose feedbacks are detailed in D3.1 [26].

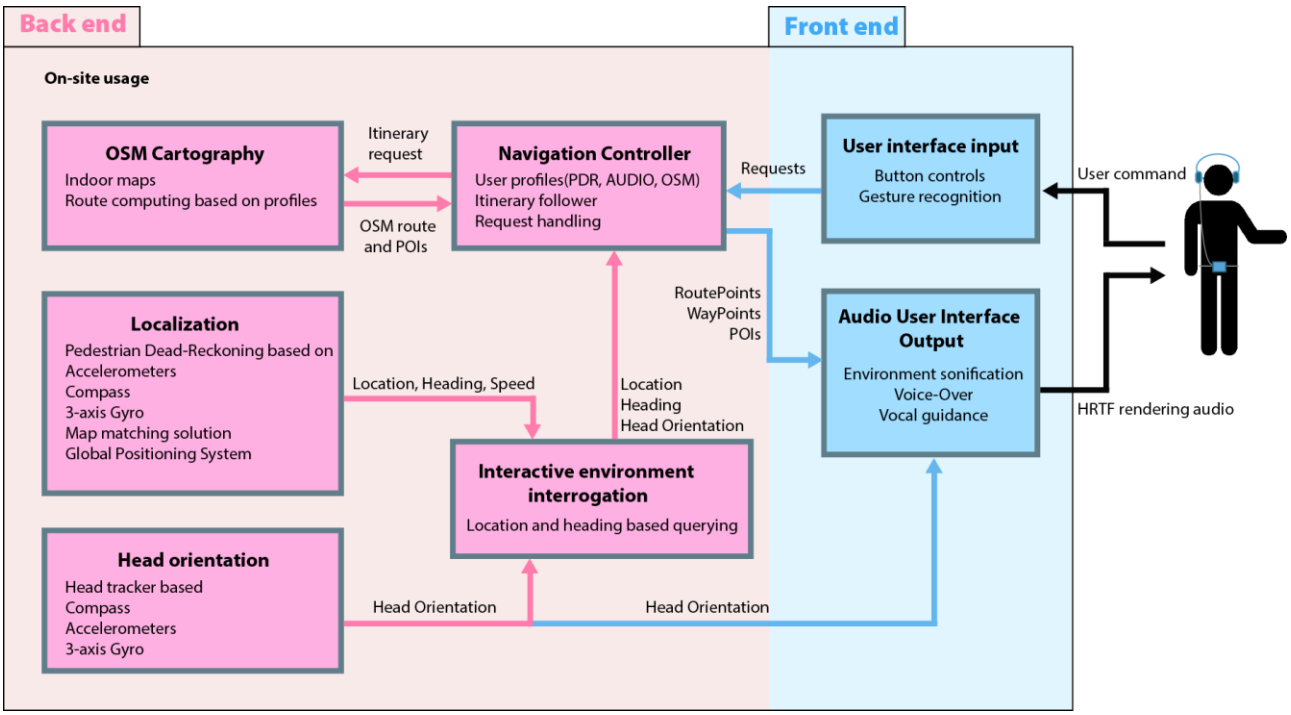


FIGURE 3.1 IXE ARCHITECTURE

3.2 Application features

As shown on figure 3.1 IXE is built above several software components:

- OSM cartography:
  - Manages map resources
  - Computes shortest path
- Localization
  - Provides current location using PDR regardless of whether the user is indoor or outdoor.
  - Calibration process view
- Head orientation: provides body orientation with respect to the north. Actual IXE version takes into account device orientation and not head orientation. In future work a head tracker based on compass, accelerometers and 3-axis gyroscope could be a very interesting way to investigate in.

- Interactive environment interrogation: at anytime user can have access to information about nearby environment by pointing the device in the wished direction.

- Navigation: Turn-by-turn instructions are provided for the computed route and synthetized by a text-2-speech engine

In addition to these components which provides 'must have' features, some convenient features has been added to the application in order to improve user experience:

- QRCode reader: OSM network and routes can be directly downloaded and then imported in the application by scanning a QRCode without exiting the application

- Mobile OSM route editor: allows an author to edit on the fly a computed route. This feature is very relevant for visually impaired people who want to add custom audio instruction for a specific route

- User profiles: user settings can be saved by the application.
  There are three classes of settings:
  - Routing settings: user can choose if the router has to take into account stairs and/or lifts and if instructions must be automatically generated
  - Audio settings: user can choose audio beacon behaviour, never enabled, when user is stopped, always enabled
  - PDR settings: physiological and pedometer parameters (user height, calibration value, IMU position)

3.3 User interfaces and interactions

IXE user interfaces are simple and respect iOS human interface guidelines [31]. Focus on the primary task is the most important guideline to respect when an application is designed for visually impaired people. Each presented view to the user must be clear and interactions limited as much as possible to a single or two tasks. For visually impaired people it is very complicated to use a touch screen, that's why IXE supports VoiceOver and defines accessibility attributes for each UI element. An application is accessible for visually impaired people when all user interface elements with which users can interact are accessible. Figure 3.2 shows the main menu of the application where it is possible to choose a route among those previously saved. User has just to scroll down or scroll up the view to select a route, when its finger touches a cell VoiceOver announces the route name.

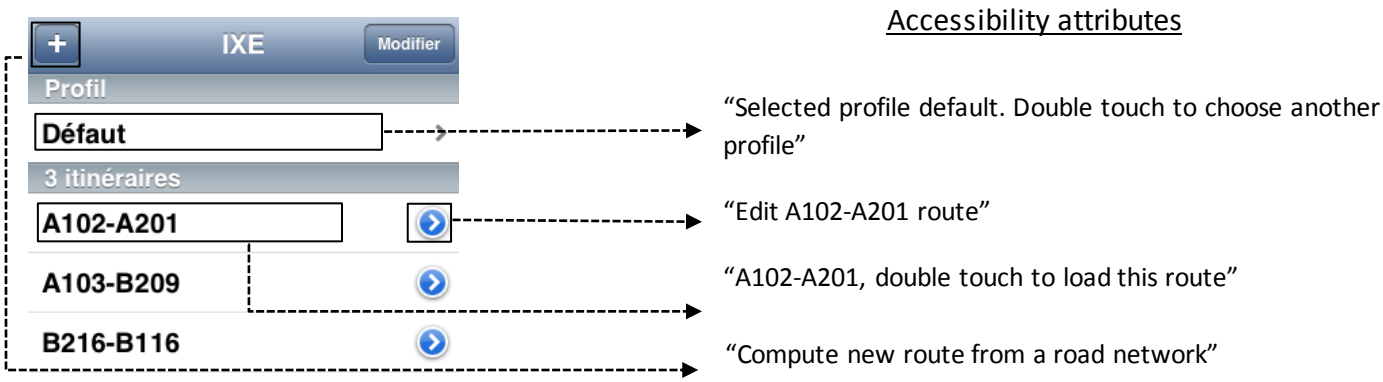


FIGURE 3.2 MAIN MENU SUPPORTING VOICEOVER

When user double touches the screen a new view (Figure 3.3) appears and invites the user to start the navigation. It is possible to enable simulation from this view by touching the auto button.

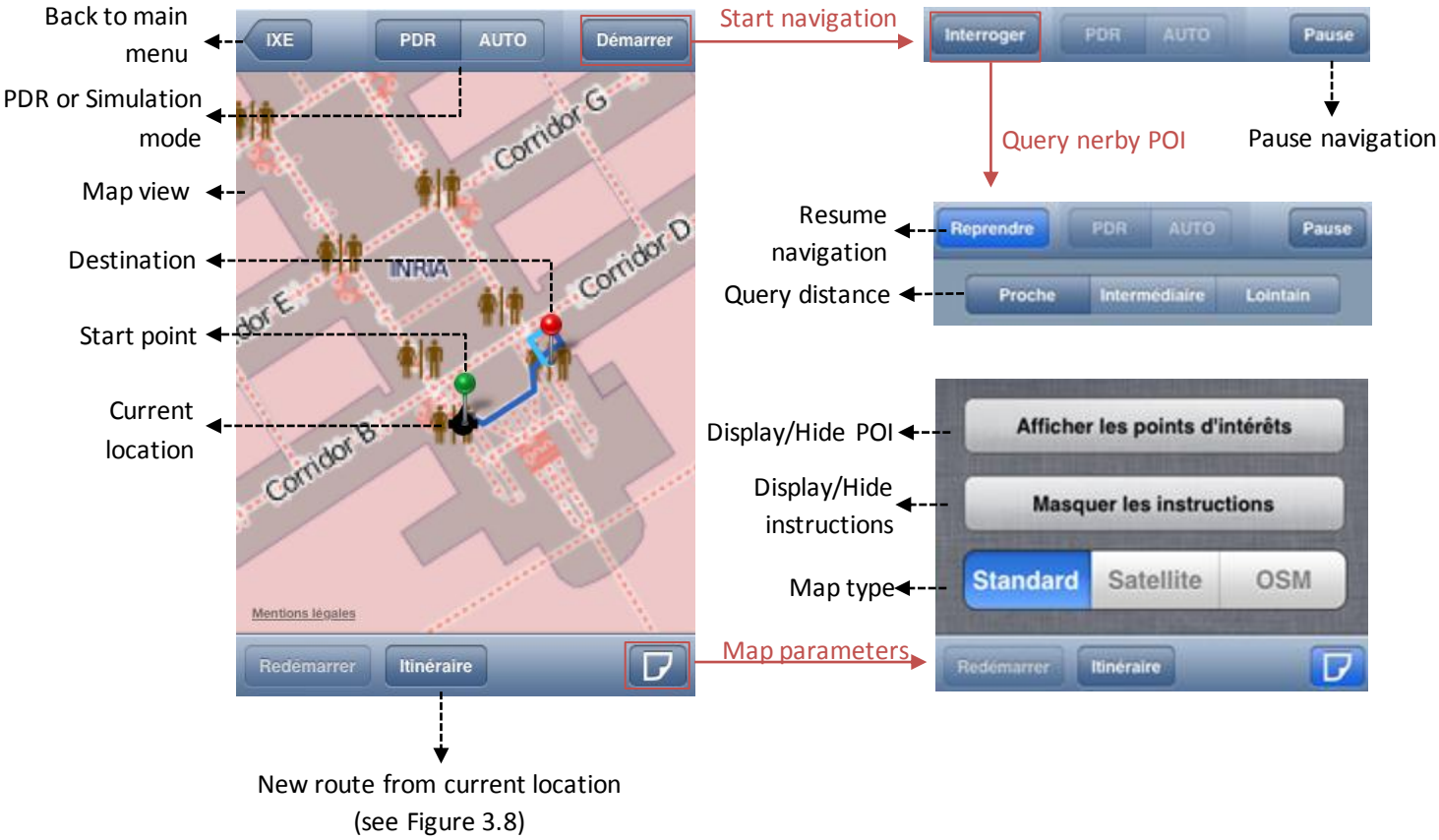


FIGURE 3.3 NAVIGATION VIEW

Figure 3.3 introduces navigation interface with a map and button to start, pause or stop localization process. This interface is not suitable for visually people because of small buttons and the map, which is useless, but it is important to provide such interface for valid users. To meet the needs of visually impaired people all interaction on this view can be control by headphones buttons as shown on figure 3.4.

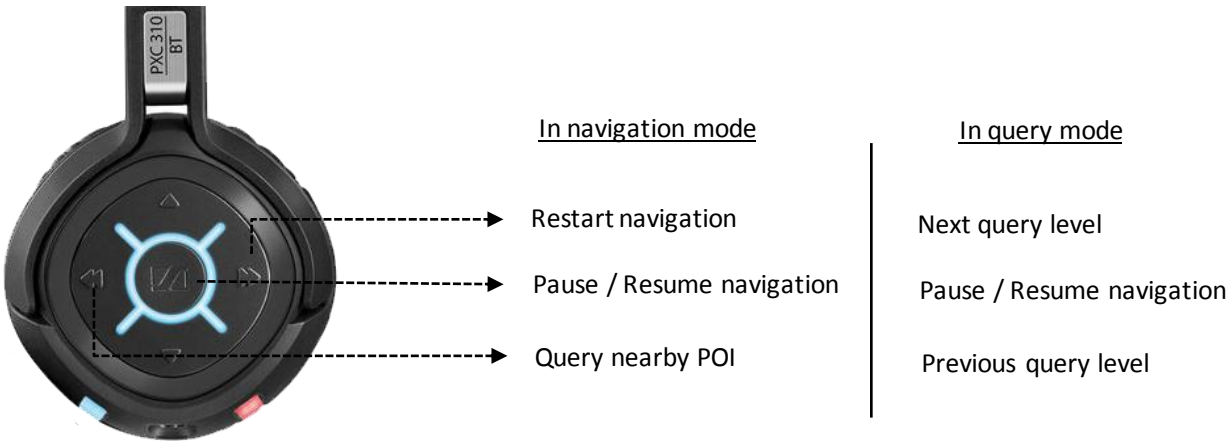


FIGURE 3.4 HEADPHONE CONTROLS

It is possible to query nearby POI according to three predetermined distances (5m, 10m and 30m) as shown on figure below:

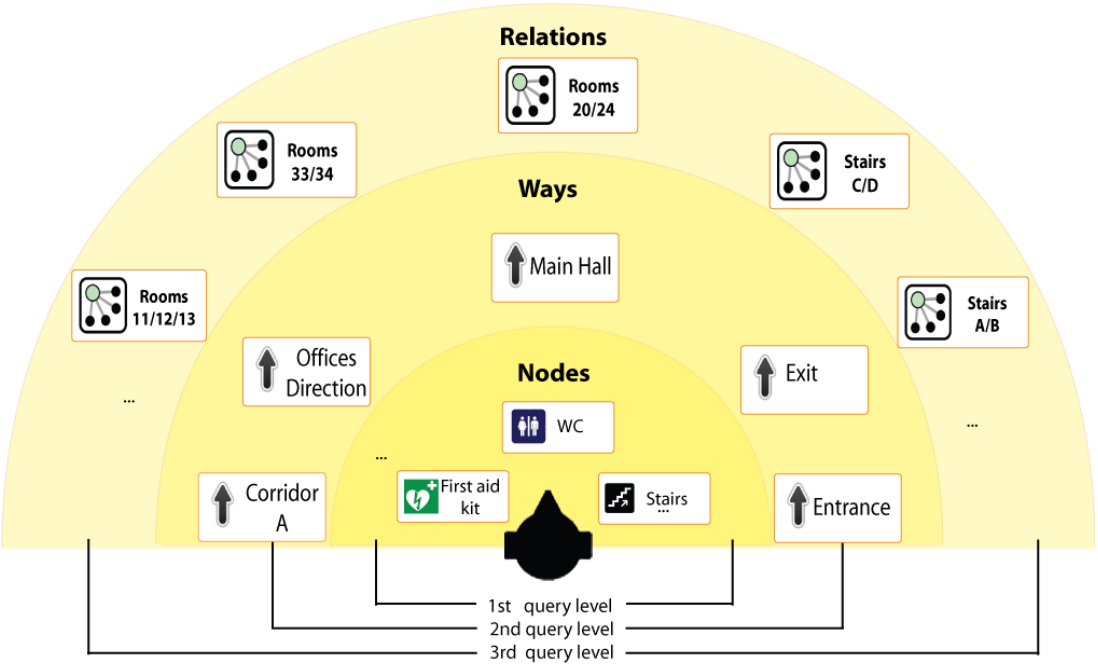


FIGURE 3.5 - QUERY LEVELS

Interfaces for user profile (Figure 3.6 and 3.7) respect high contrast guideline (white character on black background). This view is divided into 4 parts: routing parameters then audio parameters and physical parameters and finally pedometer parameters.

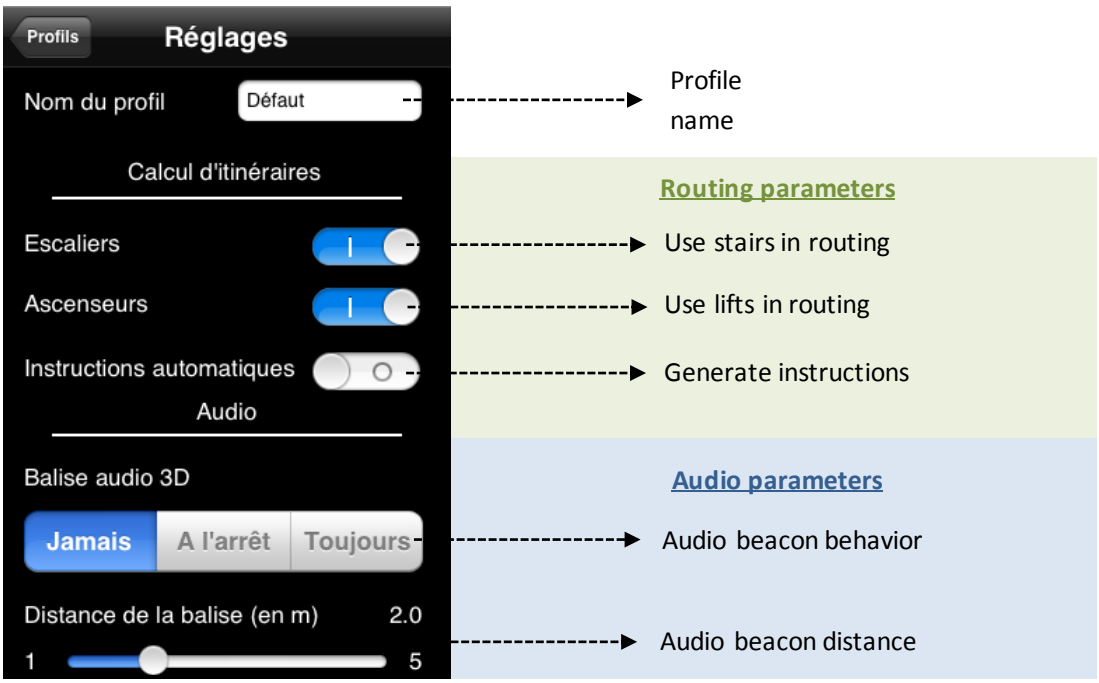


FIGURE 3.6 ROUTING AND AUDIO PARAMETERS

FIGURE 3.7 PEDOMETER AND CALIBRATION PARAMETERS

IXE allows computing a route from an OSM network (figure 3.8). The user can load an OSM file and sets start point and destination. Then, he can modify routing parameters to define if the route computed can cross stairs and/or elevators. Finally, the user defines if the application has to generate audio instructions.

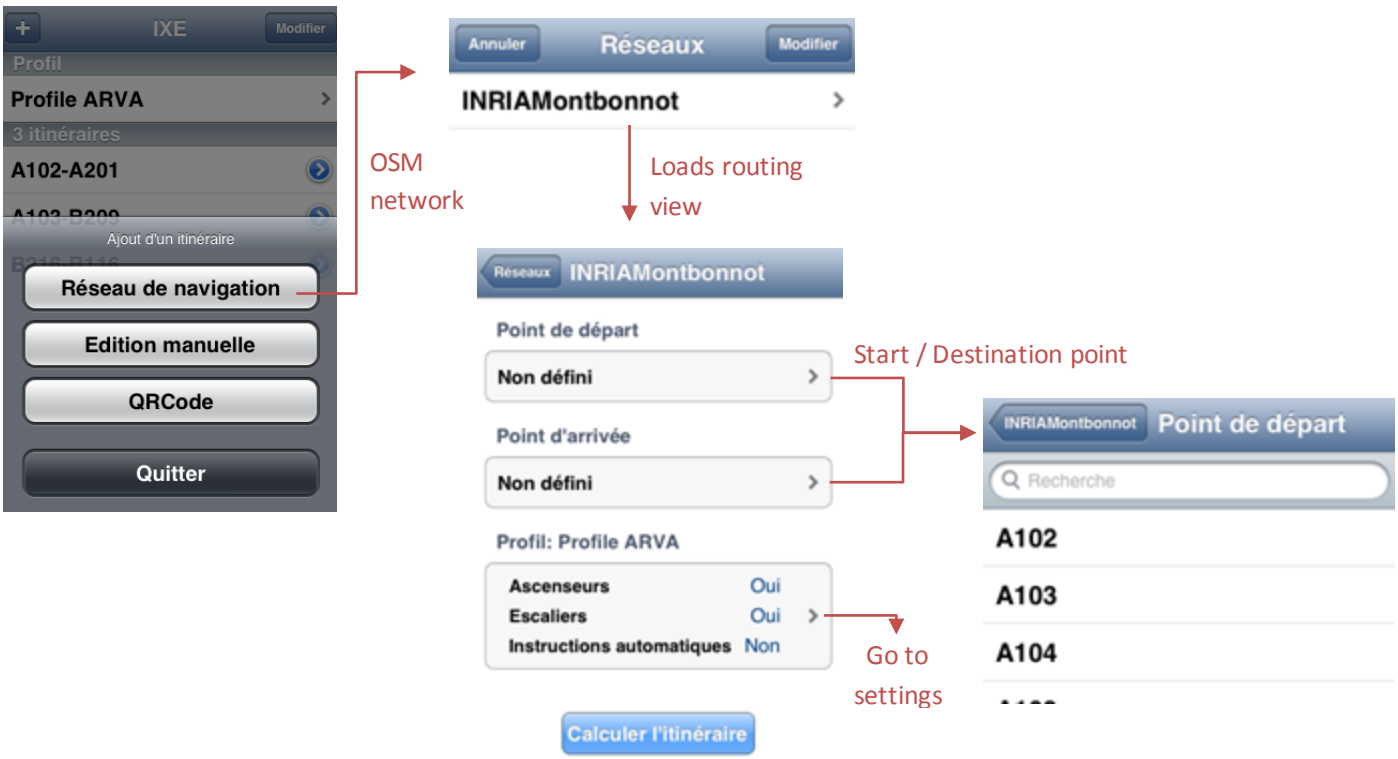


FIGURE 3.8 ROUTING IN IXE

The last feature is the embedded QRCode reader, which allow scanning and then downloading an OSM network or a route. A relevant use case could be the scan of a QRCode on business card referring to a predetermined route, from the INRIA office reception to a room for example.

FIGURE 3.9 QRCODE READER

3.4 Guiding people using a navigation audio beacon

Using HRTF (Head-Related Transfer Function) rendering, it is possible to create a virtual audio beacon that is precisely positioned in the 3D space. This audio beacon can be used as a navigation guide to follow, while still providing other navigation cues at the same time. Using a spatialized audio beacon instead of synthesized speech reduces the user cognition while providing him confirmation feedback at all times. It is especially useful for users with eyesight disabilities.

3.4.1 Pre-rendering the navigation audio beacon

Because mobile phones have limited processing capabilities, we need to use a set of pre-rendered HRTF samples to create the audio beacon, as HRTF rendering is a processor intensive task.

The principle is to divide the circular space around the listener into a definite number of points, and pre-calculate the HRTF 3D spatialization of the audio beacon for each of these points (figure 3.10).

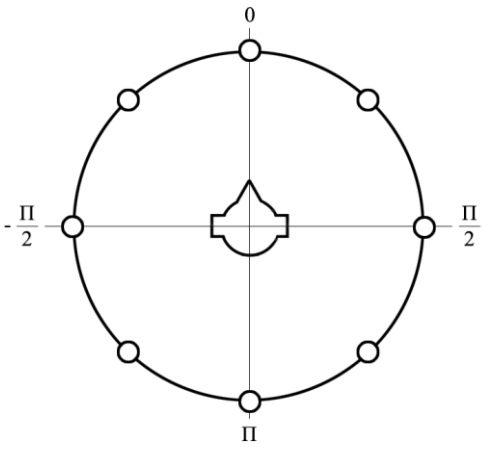


FIGURE 3.10 PRE-RENDERING THE NAVIGATION AUDIO BEACON USING 8 POSITIONS AROUND THE USER

With an objective of guiding people, at least 8 points are needed (4 cardinal points + 4 diagonals) for providing significant information. This granularity can be augmented for a more precise guidance, at the cost of a greater memory usage.

After having pre-rendered all the audio samples, we can use them to play the navigation audio beacon. Two methods can be implemented, and we have tested both to experiment their result:

- If we consider a short duration navigation audio beacon sound (1-2s), we can simply play them one after the other in loop while choosing the sample the closer to the direction to take before starting a new repetition.

- We can also start all samples at the same time, and then adjust in real-time the mix of these sources to obtain by interpolation the desired guiding direction. A different weight $P_{\gamma i}$ is affected to each sample corresponding to its target mix volume (figure 3.11), with variation depending of the orientation angle β relative to the head of the listener (figure 3.12).

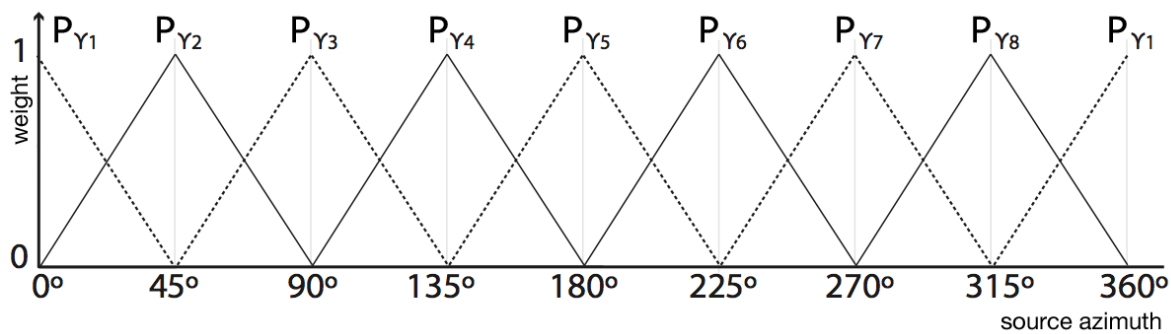


FIGURE 3.11 WEIGHTS OF THE SAMPLES DEPENDING OF THE SOURCE AZIMUTH

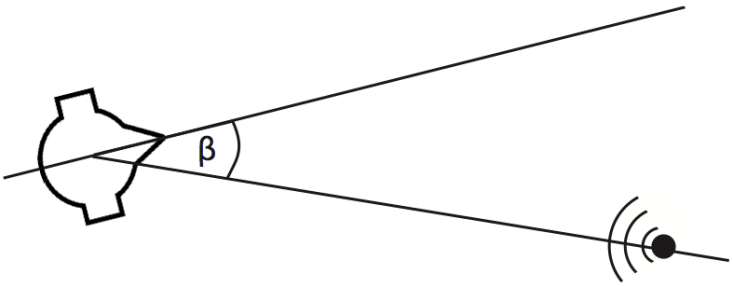


FIGURE 3.12 SOURCE ANGLE ORIENTATION *B* ANGLE RELATIVE TO THE USER HEAD

The first case can be expressed directly using our XML audio language described in D5.1, below is an example implementation of it. This template can also be adapted for the second case by changing only the synchronization parameters and adding application code to perform the mix adjustments.

```
<maudl>
   <sounds>
      <sound id="pointer" loopCount="-1" pick="manual" render3D="no" play="nav.pointer.start"
      stop="nav.pointer.stop">
         <soundsource src="pointer_p0.wav" setNext="pointer.p0"/>
         <soundsource src="pointer_p1.wav" setNext="pointer.p1"/>
         <soundsource src="pointer_p2.wav" setNext="pointer.p2"/>
         <soundsource src="pointer_p3.wav" setNext="pointer.p3"/>
         <soundsource src="pointer_p4.wav" setNext="pointer.p4"/>
         <soundsource src="pointer_p5.wav" setNext="pointer.p5"/>
         <soundsource src="pointer_p6.wav" setNext="pointer.p6"/>
         <soundsource src="pointer_p7.wav" setNext="pointer.p7"/>
      </sond>
   </sounds>
</maudl>
```

Here we have used a single *sound* container named *pointer* in which we add the 8 samples corresponding to the 8 pre-rendered pointer positions, as sound sources. It is configured to play in loop (*loopCount* attribute set to -1) with a manual selection of the current sample (*pick* attribute) and 3D rendering deactivated, as it is pre-calculated. During the user movements, the position manager will select the best sample to play corresponding to the direction to take by sending an event in the form *pointer.p** to the sound manager.

As for the second case, experiments showed various difficulties regarding its implementation: perfectly synchronizing the 8 sources to play in parallel is a difficult task: it would need to develop a specific sound library with sample-precise synchronization. In addition, various simple tests with users showed that dues to the different imprecision factors (synchronization of the samples, mixing delay) the results were confusing for the users leading to guiding errors and misinterpretation of the audio beacon direction. This is why we chose to stick with the first method that is also simpler to implement on many platforms.

3.4.2 Additional improvements

In order to give the user the maximum amount of information with the navigation audio beacon and minimize the error chances while improving his precision, we chose to improve the pre-rendering precision by augmenting the granularity to 16 pre-rendering positions, resulting in smoother indications. We also added a new level of information: the spatialized sound used by the audio beacon varies depending of the indicated direction, allowing the user to better correct its moving direction (figure 3.13). We have split in 5 zones the 16 different possible di-

rections and added a direction hint by changing the kind of sound of the beacon. This allows the user to discriminate quickly and efficiently the direction to take.

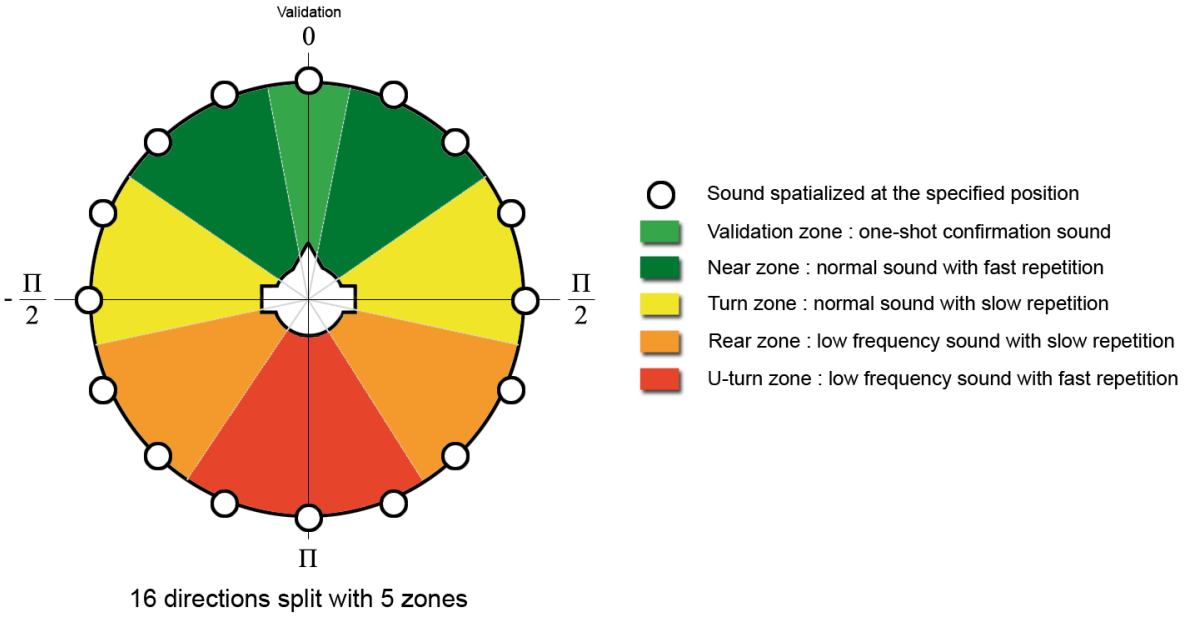


FIGURE 3.13 NAVIGATION AUDIO BEACON WITH ZONE VARIATIONS

Based on the tests we made with various kinds of users, it appears that some people feel more confortable at first with an audio beacon based using voice instructions, especially the ones having difficulties to hear the 3D sound spatialization or too hasty to overcome the slight learning curve. We have then designed a second type of navigation audio beacon based on the same concepts but using voice instructions for the 16 directions instead spatialized sounds. With this pointer, the ability to receive additional voice instructions from the environment is lost, and the instantaneousness of the cognition due to the longer duration of the instructions and the need to interpret them. This second version of the audio beacon is then not recommended for most users, as it only benefits people with hearing impairments past the learning curve of the first version.

3.4.3 Using a head tracker to improve sound spatialization

In the heart of the current navigation systems, the direction indicated by the audio beacon considers the listener head to have the same orientation as the listener body. This supposition works generally well in most cases, but it is more natural for the user to only turn his head without turning its body when quickly searching for the direction to take.

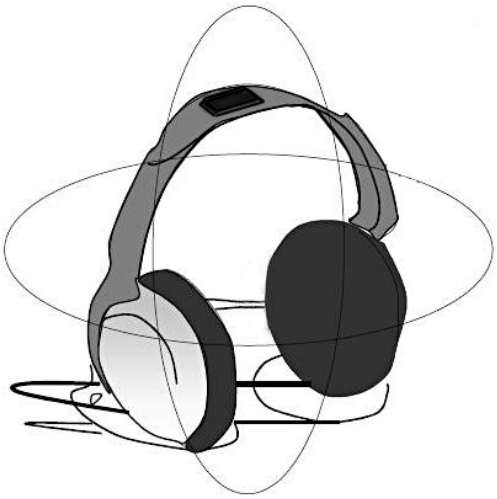


FIGURE 3.14 EXAMPLE OF A HEAD TRACKING DEVICE BASED ON A GYROSCOPE

In order to complete our spatialization system, we have made experimentation using a *head tracking* device (figure 3.14). This hardware module can connect to a mobile phone and embeds various sensors, in particular a gyroscope and magnetometer. When positioned on the user head, this module allows the navigation system to track the head rotation relative to the user body. Using this rotation, we can fix the sound spatialization to reflect the exact orientation of the user head, instead of his body: the virtual sound space is then stabilized and the precision of the navigation audio beacon further improved.

Though, this system has limitations: in order to extract the exact rotation of the head relative to the body, a complex calibration process has to be performed, otherwise the whole system would generate more orientation errors than improvements.

4 OSM authoring

The IXE application presented in section 3 provides a complete indoor/outdoor navigation solution based on OSM. In order to make quick modifications or sketch complete navigation route on the go, we also designed a mobile OSM route editor that is integrated into the IXE application. Please not that the route editor was developed initially in French, as it was the language of the IXE application mock-up at this time, so the user interface screenshots may contain French labels. A second application using a kick-scooter allows creating OSM map with a good accuracy.

4.1 Mobile OSM Route Editor

Because it is always useful to make quick editions on the go like changing audio instructions or adding POIs, we decided to create an embedded editor into the IXE application. Using the same pedometer module that constitutes the base of the PDR localization system we use for estimation of the user location, we can also create new navigation routes from scratch on this mobile editor. When you already have a plan of a place you want to navigate, it is also possible to use the touch edition mode to place route points on the map directly.

4.1.1 Requirements

In order to use all the features of the embedded editor, there are some requirements, depending of the features you want to use:

- Pedometer editing (see figure 4.1): if you want to create or edit routes using the pedometer mode, you first need to have selected and calibrated your walk profile in the IXE application, so the computed distances are accurate.
- Route editing: since the mobile editor can only open OSM route documents, you must first generate a route using IXE if you only have an OSM navigation network document.
- Starting location: when you want to create a new route from scratch, you must know your starting location. By default, your GPS location is used, but you should change it since GPS is not precise enough for this kind of usage, especially if you are inside a building.
- Audio instruction generation: the editor can automatically generate OSM audio instruction tags for basic turn-by-turn directions, but it needs a completed route to do so.
- Background image: you can add a background image layer to an OSM route, for example to trace a route manually using a building map image. This image must be configured beforehand directly in the OSM document.

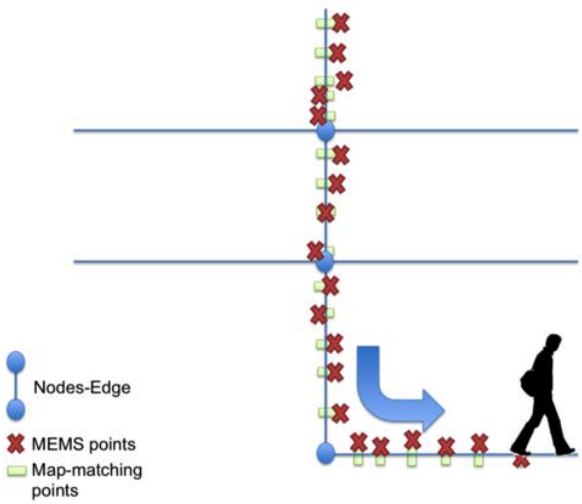


FIGURE 4.1 USING THE PEDOMETER TO CREATE AN OSM ROUTE

4.1.2 User interface

The mobile OSM editor user interface can be decomposed into 5 views:

- The main editor view, a WYSIWYG view where you can see the route, POIs and audio instructions and edit them directly.
- The document properties / editor preferences view, where you edit the basic document properties, generate basic turn-by-turn audio directions and change editor settings.
- The route node position editor, to change manually the position and level of a selected route node.
- The POI editor view, allowing editing all the properties of a selected POI.
- The audio instruction editor view that can be used to test the generated audio or edit the properties of the selected audio instruction.

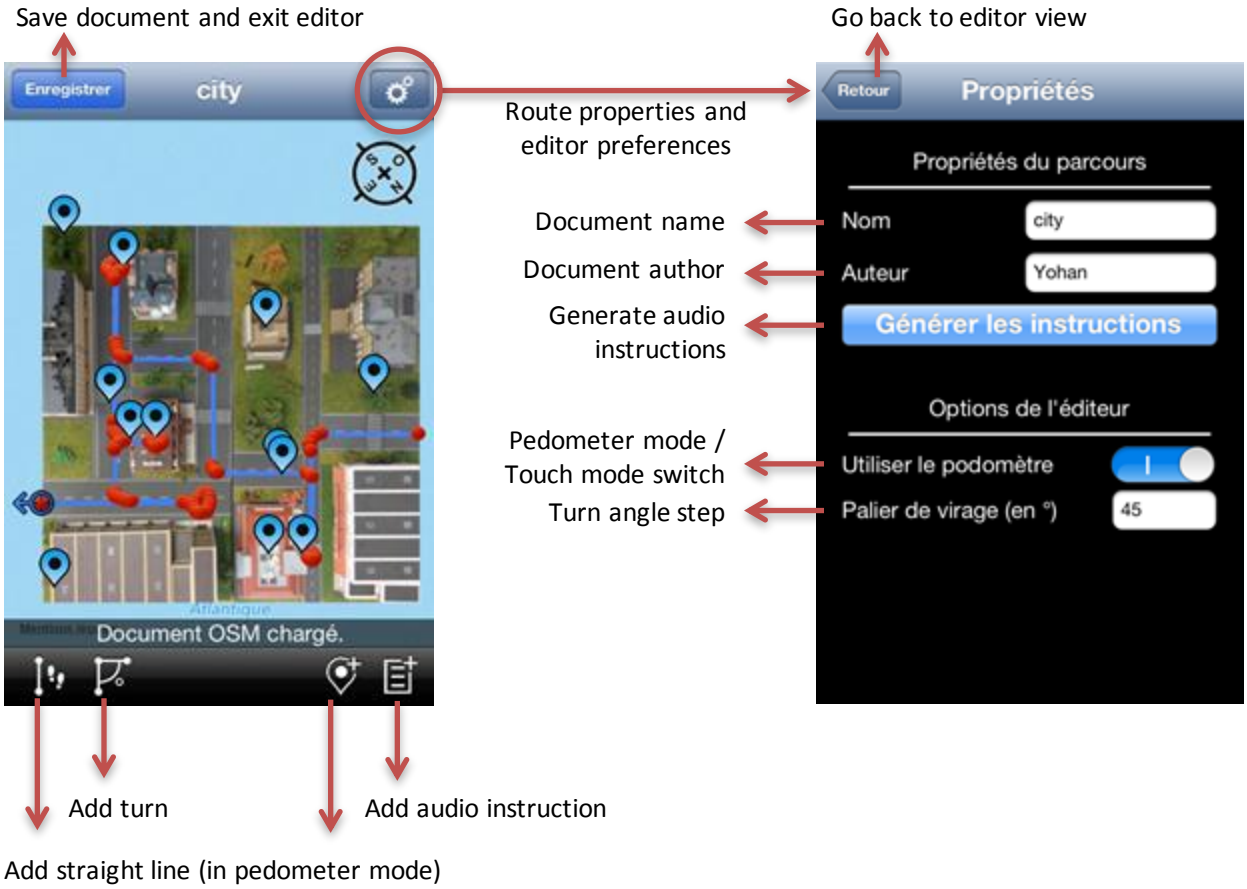
We will now a look at how these views are linked to each other and how we designed the user interactions for an intuitive mobile editing experience.

Main editor interface

When you load an existing OSM route document, you are directly presented the main editor interface where you see the route with the POIs and audio instructions as a whole (see figure below, left screenshot), as well as a compass indicating your current orientation and a toolbar to make edits.

Each map feature has a distinct representation for a quick and easy visualization:

- Route nodes are represented by a red circle
- Route edges are represented by a blue line (the colour goes darker/lighter for differentiation of the levels)
- POIs are represented by a blue pin
- Audio instructions are represented by a yellow note
- The current position and orientation are represented by a blue target with an arrow indicating the current direction on the map



On the top of the main editor view, there is 2 buttons on the left and right side: the first one to save and quit the editor, the second one that opens the route properties / editor preferences view. Since the editor preferences are saved within the OSM document and thus are related to the document properties, it made sense to group these two sections together.

On the bottom of the view is located the toolbar, with a status just above providing contextual information. You can immediately know which route edit mode enabled by looking at the first toolbar icon, as shown on the figure below:

Pedometer edit mode

Touch edit mode

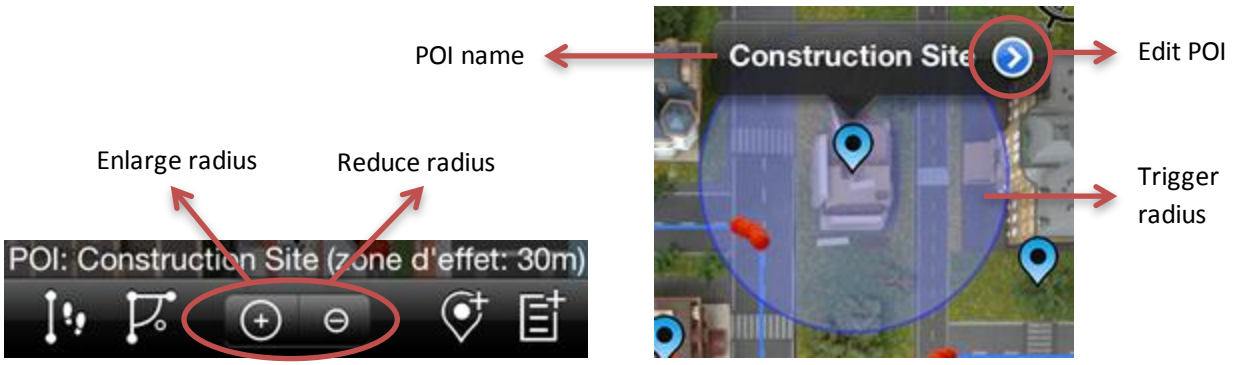
4 icons compose the main toolbar, from left to right:

- The first one allows adding a straight line to the route, starting from the current position (and orientation when in pedometer mode). Depending one mode (instructions are provided via the status bar), you just walk to the desired location and touch the icon again or touch the screen to the desired location to create a new route node an edge in a straight line.

- The second icon allows changing the current orientation on the map, by simply turning the device (you holding the phone) facing the desired direction. This is essentially useful in pedometer edit mode, but can still be used as a direction helper when in touch edit mode.

- The third icon allows adding a new POI, by directly touching the desired POI location on the map.
- The fourth icon allows adding a new audio instruction, the same way as a POI.

When a POI or an audio instruction is selected by touching it on the map, additional toolbar buttons appears:

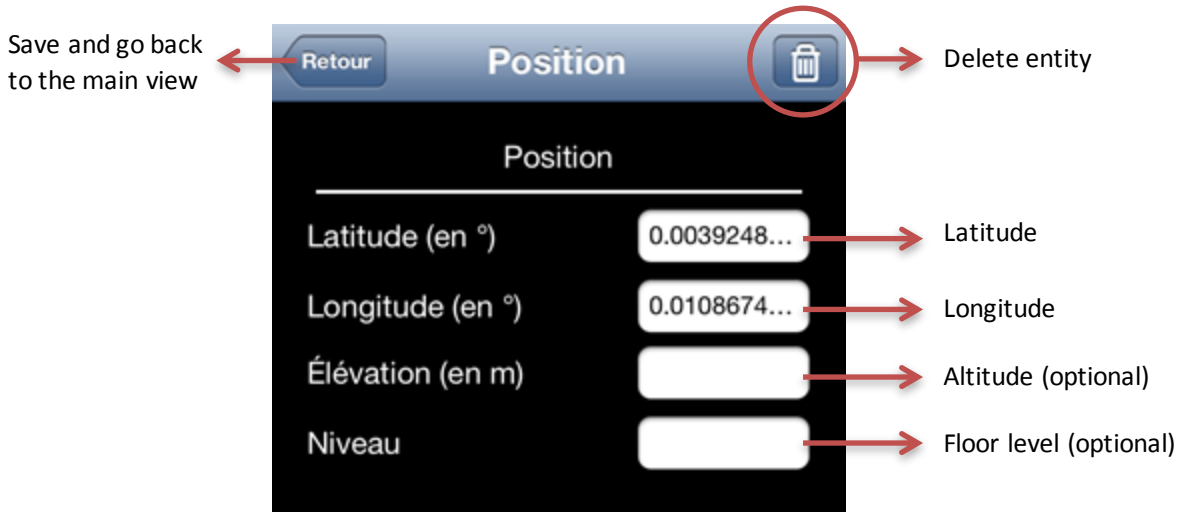


These buttons allows enlarging or reducing the triggering zone of the selected POI or audio instruction. The actual triggering zone is also expressed in meters in the status bar, and shown on a map using a transparent blue circle.

When a map entity is selected, a black popover appears with showing the feature type or its name if it is available, with a blue button on the right. When touching this button, the editor view related to the entity type is opened. A selected map entity can also be moved on the map, simply by dragging it to the desired location after touching it.

Route node editor

The route node editor allows editing the various properties of a node, such as the latitude/longitude location, the altitude and its floor level (when inside a building).



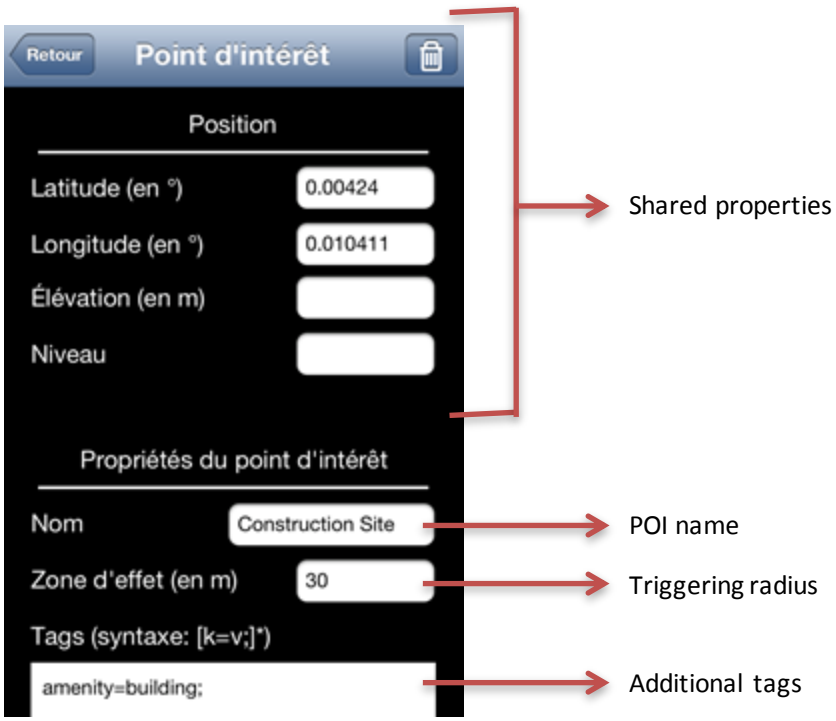
It can also be deleted from the map, by tapping the recycle bin icon: user confirmation is then asked before performing the deletion.

Asking for user confirmation before deletion

This basic view and properties is also the base of the POI and audio instruction editors, since these map entities are also OSM nodes, with more advanced properties.

POI editor

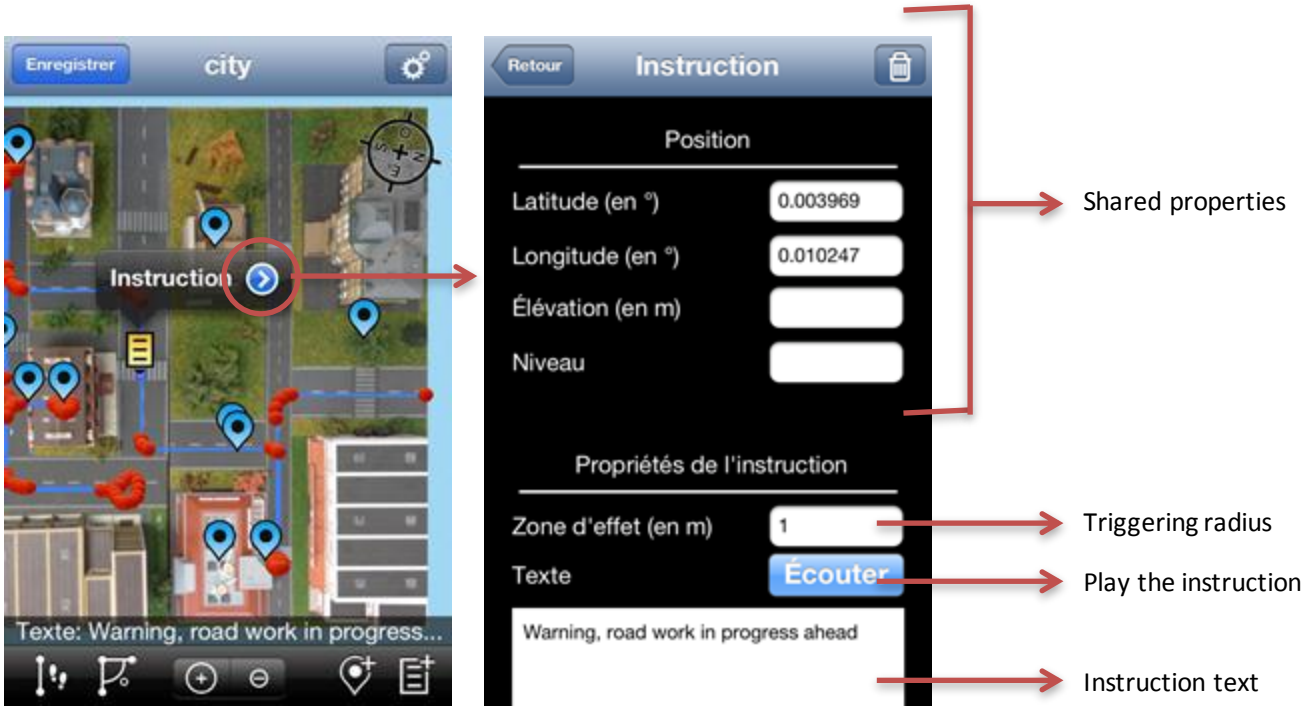
In addition to the same properties presented in the previous section, the POI editor view shows new edition fields:



You edit the POI name, set the triggering radius value directly in meters, as well as specifying multiple OSM tags using the `key=value;` syntax.

Audio instruction editor

The audio instruction editor is similar to the POI editor, with specific controls related to the audio instruction itself: you can enter the audio instruction text to be synthesized by the text-to-speech engine, and try playing the instruction by tapping the *listen* button.



4.2 Android kick-scooter

Android kick-scooter is an Android application using a kick scooter in order to map indoor and outdoor places easily in OpenStreetMap XML format. The scooter uses ANT+ [32] sensor (with gyroscope and counter wheel) to make precise plan of a travel. It automatically takes into account the corners (from gyroscope data) and can create custom POI (Points of Interest) in order to enable the user to obtain a relatively complete route. Finally, OSM files are collected, cleaned and recalibrated using an OpenStreetMap editor like JOSM. Figure 4.2 bellow shows users with the Android Kick-scooter at Sugimotocho train station (Japan).

FIGURE 4.2 STUDENTS USING THE ANDROID KICK-SCOOTER AT SUGIMOTOCHO, OSAKA CITY UNIVERSITY

4.2.1 Technologies

Kick-scooter is a mobile Android application (version 2.3 & higher supported) connected to ANT+ sensor on the scooter. ANT+ is a 2.4GHz practical wireless networking protocol and embedded system solution specifically designed for wireless sensor networks (WSN). Ant+ is primarily designed for collection and transfer of sensor data, to manageable units of various types. The mobile application receives sensor data (counter wheel and gyroscope) and computes a route with distance and directions.

4.2.2 Features

Android kick-scooter application allows the following features:

- Computes a route taking into account changes of directions
- Saves points of interest with voice or text
- Saves sensor data in XML files
- Saves computed route in OSM files
- Sends files by email
- Uses wheels of different diameters
- Manages multiple magnets to increase the accuracy of measurements
  (By default, it is equivalent to +/- the circumference of the wheel)

4.2.3 Experimentation

The application allows creating OSM map of public indoor places. Users can take data discreetly and create the full map when they come back home using an OSM editor. This solution is a way to create map more quickly with a good accuracy. The application was used to create maps of Grenoble station and Osaka station. Right image of figure 4.3 shows the OSM document of Sugimotocho station map created with Android kick-scooter.

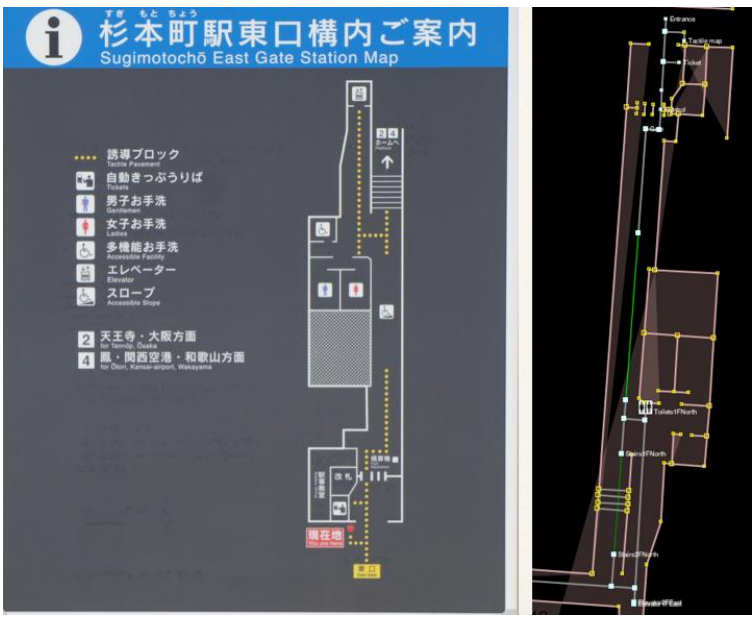


FIGURE 4.3 OSM DOCUMENT OF SUGIMOTOCHO STATION

5 PDRTrack: localization test application

5.1 Introduction

A mobile application called PDRTrack has been designed to test and find the best parameters for the Pedestrian Dead-Reckoning (PDR) algorithms. This application provides the current user location for each step and displays it on a map. Navigation and routing are not part of PDRTrack, to start the localization the user has only to choose a start point and a start orientation. Similar applications exist for infrastructure-less indoor navigation such as FootPath [25] from Aachen but localization in this application is constraint to a computed route. From PDRTrack side since localization is not based on a predetermined route but on a full navigation network it is possible to freely move in the building. For example, user can enter in a room and then go back to a corridor, take the stairs and finally turn back. From tests done with visually impaired people [26], a software requirement has been added: the localization must not be constraint to a specific route because user can leave the route at anytime. Another main innovation of PDRTrack is the dataset recording and reading mode, which enables the user to save all sensors values during his walk and replay them. This feature is particularly useful to test the same walk with different algorithm parameters without walking again.

This part will detail this mobile application from requirements to user interfaces.

5.2 Requirements

5.2.1 OpenStreetMap document

The PDR module needs an OSM [28] document description (indoor, outdoor or both) in input as detailed in part 4.1 of deliverable D4.3 [27]. As often as possible the PDR will try to match the current location on a way defined in the OSM document. There are two map-matching algorithms, one for correct the location and the other one for the orientation, richer the map is better corrections will be. This document won't detail these algorithms, please refer to D4.3 for further information. Here below an OSM example with a way that can be used by map-matching algorithms. This way includes standard tags and values from the approved map features [29], additional tags needed for the navigation are introduced in D4.3.

```
<?xml version='1.0' encoding='UTF-8'?>
<osm version='0.6' generator='JOSM'>

  <!-- Nodes hold by the corridor -->
  <node id='-43' visible='true' lat='40.01724758683568' lon='2.9948837556279706' />
  <node id='-18' visible='true' lat='40.02583770246074' lon='3.0069869656592285' />
  <node id='-16' visible='true' lat='40.025826707140524' lon='2.9948704796222487' />
  <node id='-15' visible='true' lat='40.0123408216216' lon='2.994891347992221' />

   <!-- Corridor definition -->
  <way id='-17' visible='true'>
    <nd ref='-15' />
    <nd ref='-43' />
    <nd ref='-16' />
    <nd ref='-18' />
    <tag k='highway' v='footway' />
    <tag k='indoor' v='yes' />
    <tag k='level' v='2' />
    <tag k='name' v='Corridor' />
  </way>
</osm>
```

The OSM document can be produced by any authoring application such as ones described in section 4 and directly imported in PDRTrack.

5.2.2 Pedometer calibration

Before using the localization part the pedometer must be calibrated with the physical and physiological characteristics of the user. This calibration process is strongly recommended for a precise localization. Indeed, depending on the user height and his way of walking each stride length won't be the same. The pedometer doesn't provide a mean value but estimates the distance from a parameterized model of walking with the calibration value. Depending on the device position the acceleration peak enabling the detection of a step is different. Indeed, if the device is hand-held the vertical acceleration will be more stable than if the device is chest-mounted, that's the reason why the vertical acceleration threshold is a pedometer parameter. The device orientation is automatically detected, so landscape, upside down, face down or face up mode doesn't affect step detection algorithm.

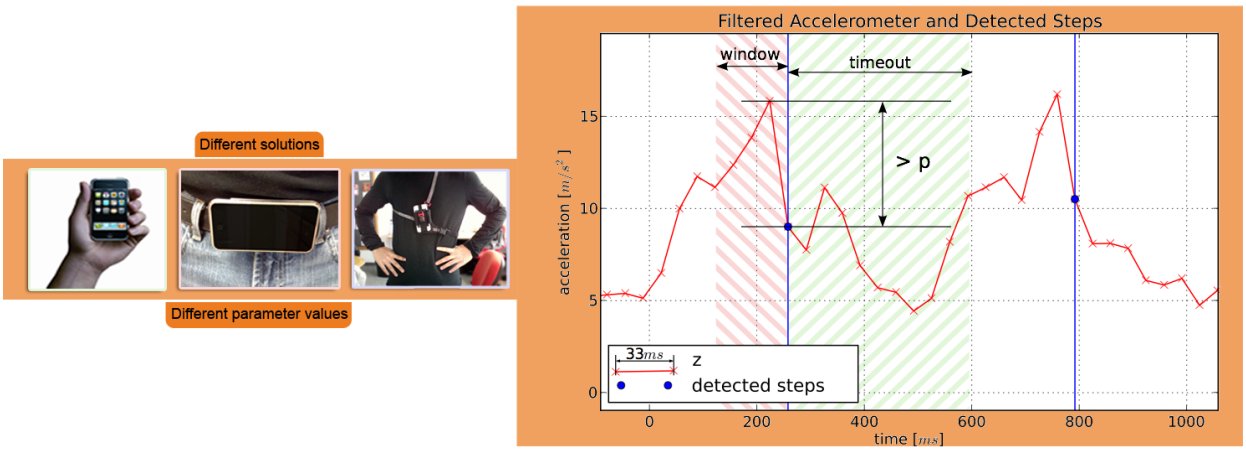
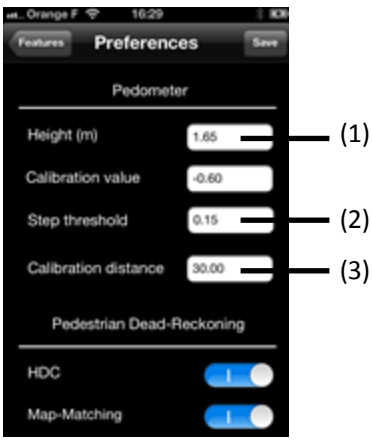
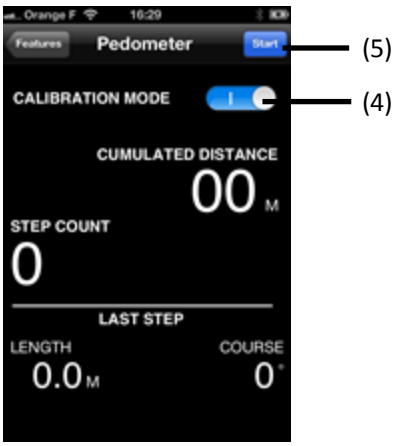


FIGURE 5.1 CHEST/BELT MOUNT OR HAND-HELD PDR

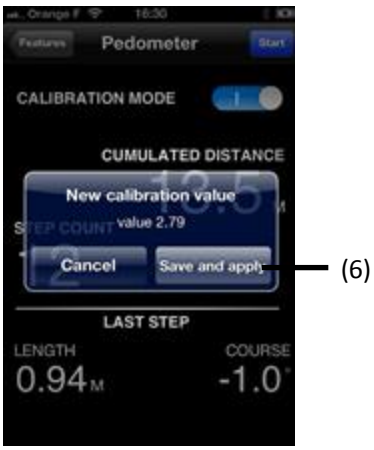
The calibration process is in 6 steps:



(1) Enter your height in meter
(2) Choose the acceleration threshold
(3) Choose the calibration distance



(4) Enable calibration mode
(5) Start the pedometer and walk the distance entered in (3)



(6) After stopping the pedometer save and apply the new calibration value

5.3 User Interfaces

PDRTrack user interfaces are divided into two parts:

- Left part
    1. OSM network: list of all OSM files imported in the application
    2. Dataset: list of all files containing sensors values recording
    3. Pedometer: access to calibration parameter
    4. Drift elimination: access to heading drift elimination algorithm parameters
    5. Map-Matching: access to map-matching algorithm parameters

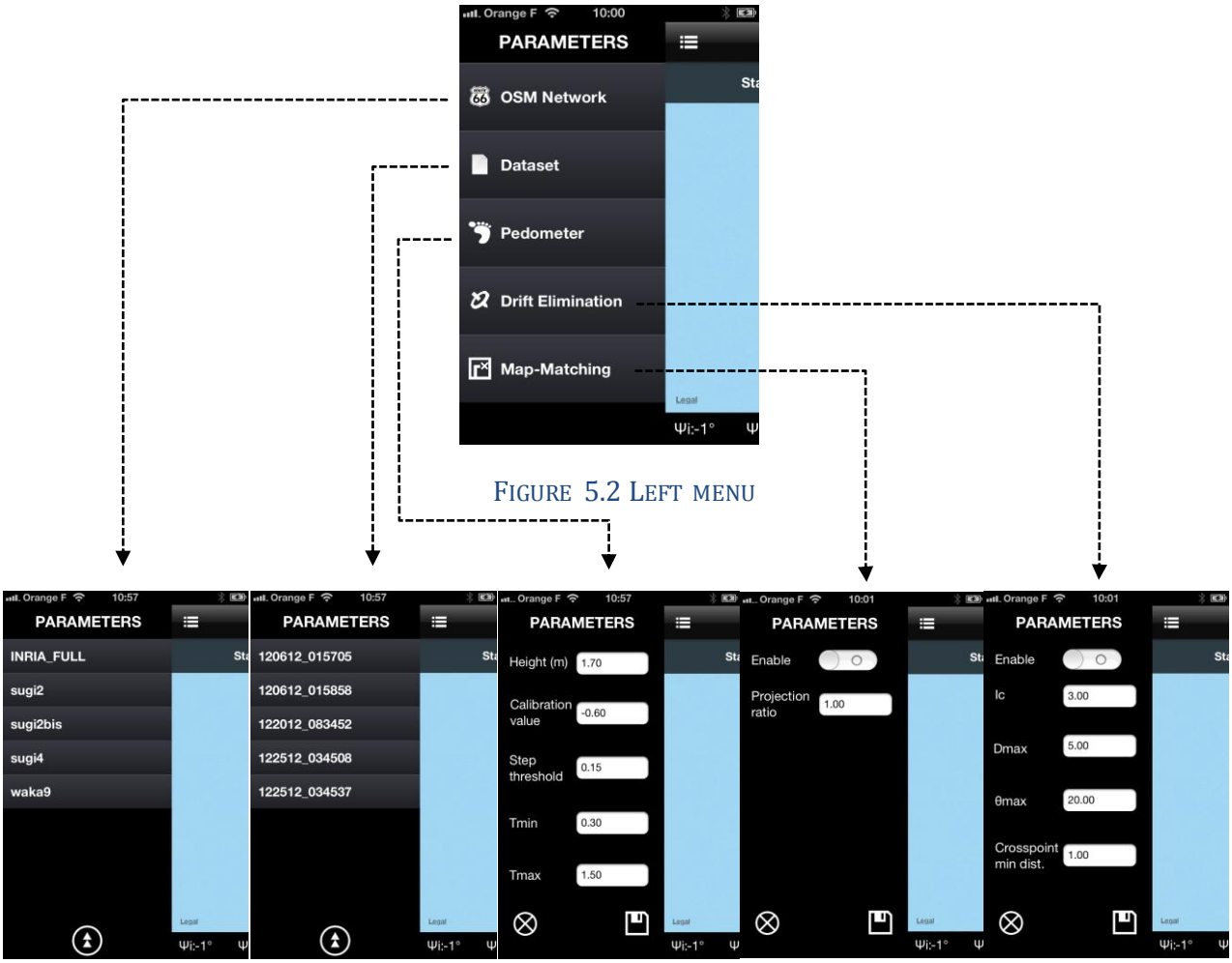


FIGURE 5.2 LEFT MENU

FIGURE 5.3 LEFT SUBMENUS

When the PDR is in recording mode the produced dataset has the current date for name according to this format "MMddyy_hhmmss".

- Right part
    - Localization view: map with overlaid user position
    - Start point and orientation chooser view: enables user to choose from which position to start the localization
    - Pedometer view: provides pedometer information such as step length, cumulated distance and number of steps
    - Dataset recording/reading mode: a switch button enables recording or reading mode

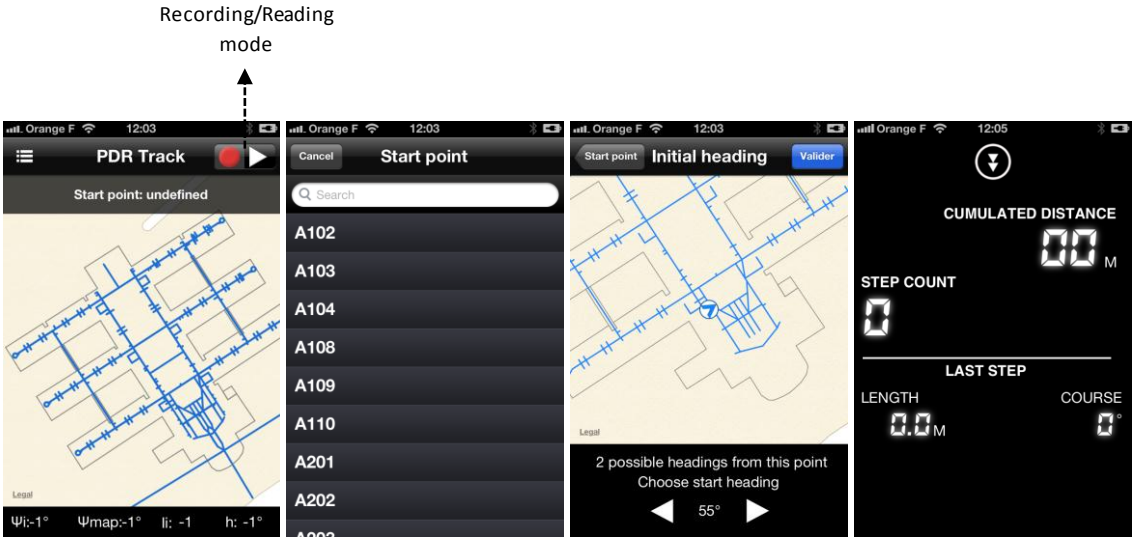


FIGURE 5.4 MAIN PDRTRACK VIEWS

During the localization the trace describing the current trajectory of the user is directly drawn on a map, in red as shown on Figure 5.5. A well-known issue with native map view for indoor visualization is the limited zoom provided by the MapKit. Depending on which map type that was in use the zoom can reach the level 21 (satellite tiles), level 19 otherwise. This zoom level is not enough for navigating inside a building, that's the reason why visualization based on SVG and HTML 5 is currently in progress.

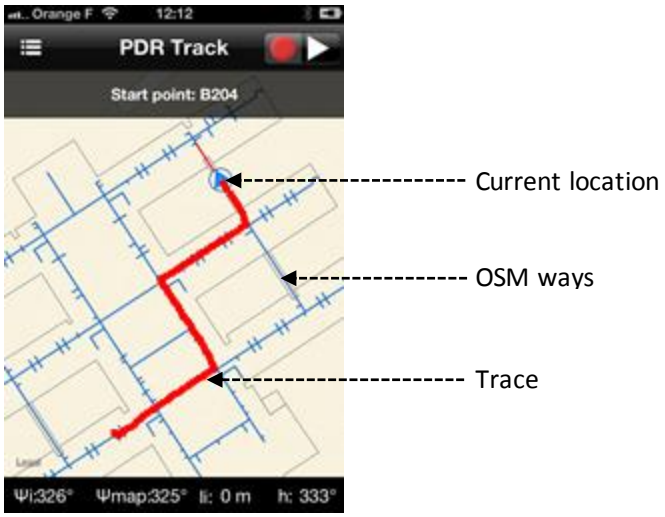


FIGURE 5.5 REAL TIME TRACE

Thanks to indoors localization reliability, user should be able to zoom as much as he wants on the map. Latest mobile phones can easily handle CSS transformation on SVG maps in a WebView, that's why we have choose to use HTML5 features to be more portable on others OS.

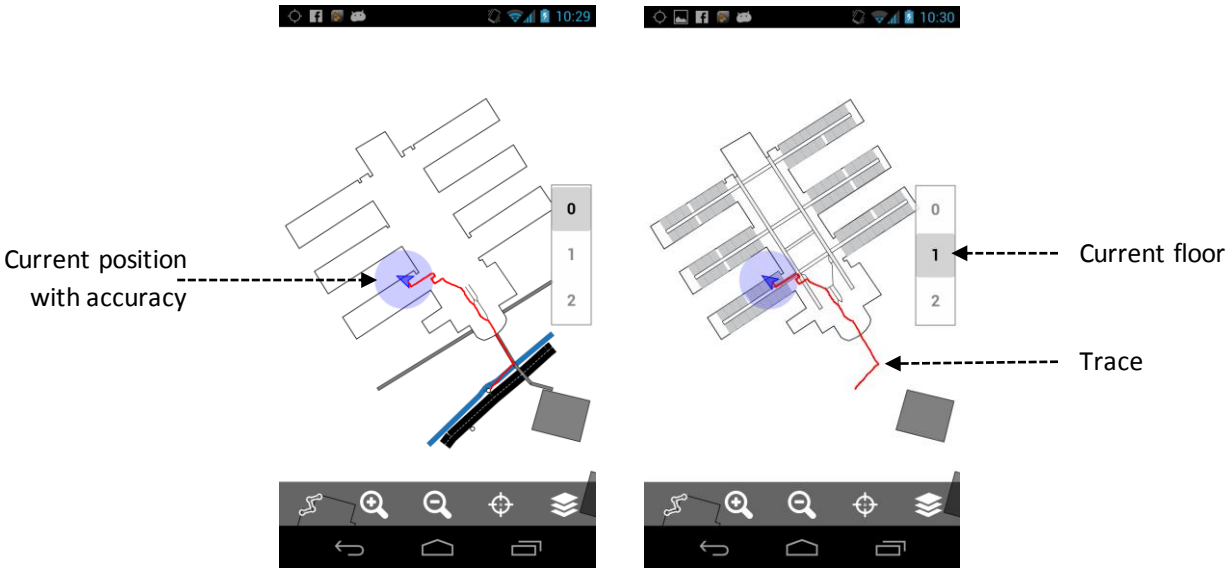


FIGURE 5.6 REAL TIME TRACE ON SVG MAP MULTI FLOOR

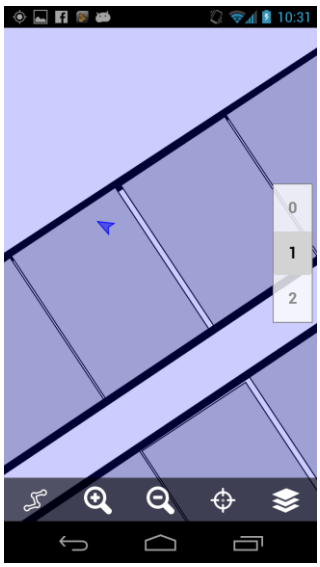


FIGURE 5.7 ZOOM TO LEVEL 24 NOT PIXELATED

Some basics features have been integrated to navigate through the map (Figure 5.6):

- Zoom-in / Zoom-out not pixelated (Figure 5.7)
- Translate with fingers (mobile web browser) or mouse (desktop web browser)
- Follow current position

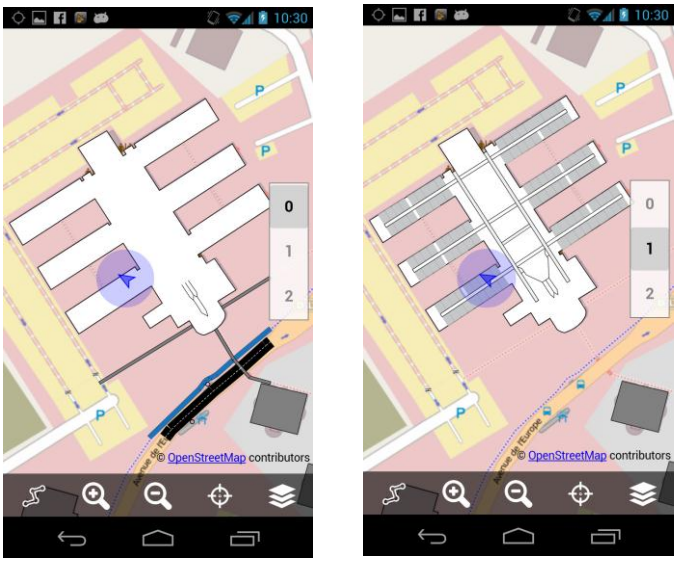


FIGURE 5.8 SVG MAP MULTI FLOOR WITH OSM SUR LAYER

To enhance visualization system, these following functionalities have been added:

- Multiple SVG (buildings) visualization
- Manage multiple floors (Figure 5.6)
- Show OSM, Google, Google Satellite sub layers (disabled when zoom is too high) (Figure 5.8)
- Visibility controlling according to zooming
- Embedded geographic coordinates
- Trace (Figure 5.6)

In order to improve authoring, in SVG files, all these features are handle by a new namespace, with new attributes. For example the following:

```
<svg xmlns="http://www.w3.org/2000/svg"  xmlns:svgmap="http://tyrex.inria.fr/svgmap"
        svgmap:visibleMinZoom="17" svgmap:minFloor="0" svgmap:maxFloor="2" svgmap:title="Inria"
        svgmap:startLat="45.21882318" svgmap:startLon="5.8061204" svgmap:endLat="45.21730293"
        svgmap:endLon="5.807867424">
  <g id="rooms" svgmap:floorRange="1;1">
     …
  </g>
  …
</svg>
```

This system allows designers to create their own maps easily with Inkscape/Illustrator… without any developer knowledge.

PDRTrack was tested in France at INRIA Rhône-Alpes and in Japan at Sugimotocho and Wakayamadaigaku stations. These tests have allowed improving map-matching algorithm and therefore the localization accuracy. A public video introducing PDRTrack is available on YouTube [30].

6 Results and Conclusions

This report describes tools and prototypes to show new interface and interaction design provided by new audio-visual technologies. The Wizard of Oz tool (section 2) is essential to simulate part of the system to collect feedback from users. IXE mobile application (section 3) deals with indoor and outdoor navigation based on the Inertial Measurement Unit and presents user interfaces and interactions for guiding people by using this technology. The OSM authoring part (section 4) describes solutions to create OSM maps using PDR or a kick scooter. Finally, PDRTrack (section 5) is a localization test application and provides user interfaces and especially a solution for visualization of building with many levels, based on SVG and HTML 5.

7 References

[1] J.F. Kelley, An empirical methodology for writing user-friendly natural language computer applications. In CHI '83 Proceedings of the SIGCHI Conference of Human Factors in Computing Systems, pages 193–196, 1983.

[2] S. Dow, B. MacIntyre, J. Lee, C. Oezbek, J.D. Bolter, and M. Gandy. Wizard of oz support throughout an iterative design process. Pervasive Computing, IEEE, 4(4):18 – 26, oct.-dec. 2005.

[3] Steven Dow, Jaemin Lee, Christopher Oezbek, Blair MacIntyre, Jay David Bolter, and Maribeth Gandy. Wizard of oz interfaces for mixed reality applications. In CHI '05 Extended Abstracts on Human Factors in Computing Systems, CHI EA '05, pages 1339–1342, New York, NY, USA, 2005. ACM.

[4] Y. Li, J.I. Hong, and J.A. Landay. Design challenges and principles for wizard of oz testing of location-enhanced applications. IEEE Pervasive Computing, 6:70–75, April 2007.

[5] Sony Mobile Communications. Sony SmartWatch. http://www.sonymobile.com/gb/products/accessories/smartwatch/. [Online] [Accessed 27 Dec 2012].

[6] Google Developers. Protocol buffers developer guide. Accessed 25 Oct 2012.

[7] J. P. Rolland, H. Fuchs, "Optical Versus Video See-Through Head-Mounted Displays in Medical Visualization", Journal of Presence, Vol. 9, No.3, June 2000, pp. 287-309.

[8] C. S. Montero, J. Alexander, M. T. Marshall, S. Subramanian, "Would You Do That? – Understanding Social Acceptance of Gestural Interfaces", MobileHCI 10, September 7-10, 2010, Lisbon, Portugal, ACM

[9] Y. Rogers, H. Sharp, and J. Preece, "Interaction Design - beyond human-computer interaction", 2011, pp. 51-53, 59.

[10] B. Shneiderman, C. Plaisant, "Designing the User Interface – Strategies for effective human-computer interaction", Fifth edition, 2010, pp. 304 – 308.

[11] R. Catrambone, J. Stasko, J. Xiao, "Anthropomorphic Agents as a User Interface Paradigm: Experimental Findings and a Framework for Research"

[12] M. Haake, "Embodied Pedagogical Agents – From Visual Impact to Pedagogical Implications", Doctoral Thesis, Dept. of Design Sciences, Lund University, Sweden.

[13] B.,Svensson, M. Wozniak, Augmented Reality Visor Concept, Master Thesis, *Lund University,* 2011, pp. 35.

[14] Augmented City 3D, http://www.youtube.com/watch?v=3TL80ScTLlM&feature=html5_3d, [Online] [Accessed 27 Dec 2012].

[15] Steve Mann's papers, http://n1nlf-1.eecg.toronto.edu/research.htm, [Online] [Accessed 27 Dec 2012].

[16] S. Mann, "'WearCam' (The Wearable Camera): Personal Imaging Systems for long-term use in wearable tetherless computer-mediated reality and personal Photo/Videographic Memory Prosthesis"

[17] Y. Ishiguro, J. Rekimoto, "Peripheral vision annotation: noninterference information presentation method for mobile augmented reality", Proceedings of the 2nd Augmented Human International Conference Article No. 8, ACM, New York, USA, 2011.

[18] Eyez 720p video streaming, http://www.engadget.com/2011/12/07/eyez-720p-video-streaming-recording-glasses-hands-on-video/ [Online] [Accessed 27 Dec 2012].

[19] Demonstrating a next generation head-up display (HUD), http://www.pioneer.eu/eur/newsroom/news/news/next-generation-head-up-display-HUD/page.html , [Online] [Accessed 27 Dec 2012].

[20] Pioneer Augmented Reality Head-Up Display Navigation System Arriving 2012, http://www.geeky-gadgets.com/pioneer-augmented-reality-head-up-display-navigation-system-arriving-2012-video-24-10-2011/, [Online] [Accessed 27 Dec 2012].

[21]Eye tracking research, http://www.tobii.com/en/eye-tracking-research/global/, [Online] [Accessed 27 Dec 2012].

[22] J. L. Gabbard, J. Zedlitz, J. E. Swan II, W. W. Winchester III, "More Than Meets the Eye: An Engineering Study to Empirically Examine the Blending of Real and Virtual Color Spaces", IEEE Virtual Reality 2010

[23]Colour blind UI check tool, http://www.vischeck.com/vischeck/vischeckImage.php, [Online] [Accessed 27 Dec 2012].

[24] More than pictures under glass, http://worrydream.com/ABriefRantOnTheFutureOfInteractionDesign/, [Online] [Accessed 27 Dec 2012].

[25] Paul Smith, "Accurate Map-based Indoor Navigation Using Smartphones", IPIN 2011, http://www.comsys.rwth-aachen.de/fileadmin/papers/2011/2011-IPIN-bitsch-footpath-long.pdf

[26] VENTURI consortium, "D3.1: Report on user expectations and cross modal interaction", January 2013.

[27] VENTURI consortium, "D4.3: WP4 outcome definitions and API specifications for inter-task / inter-WP communications ", January 2013.

[28] OpenStreetMap official website, http://openstreetmap.org

[29] OpenStreetMap features http://wiki.openstreetmap.org/wiki/Map_Features

[30] PDRTrack overview, http://www.youtube.com/watch?v=MisAjkCi0m0

[31] iOS Human Interface Guidelines, http://developer.apple.com/library/ios/#documentation/UserExperience/Conceptual/MobileHIG/Introduction/Introduction.html

[32] Ant+, http://www.thisisant.com

8 Appendix for Research Material

8.1 H1: Using AR visor for AR is better than using phone or tablet for AR

Explanation

Mobile augmented reality (AR) applications are becoming increasingly popular but the experience is not very immersive and neither is the interaction since the interaction area is limited to palm size screen. Using the phone or tablet for augmenting the world is like watching the world through a keyhole. Further the augmentation on a phone or tablet is done on a picture/image of reality and not on the reality as you perceive it which is a degradation of the world to the quality and speed of the camera sensor. However AR visors needs to fuse information with objects which demands spatial calculations, tracking etc. Further cognitive, perceptual, ergonomic and social issues must all be considered when developing AR visors.

With AR visor the AR application(s) can be running so the augmented information is always available. However with a phone or tablet the user must actively initiate the use of the AR application and point the device in the desired direction for there to be any augmented information available. This would feel awkward when standing in a public spot and holding up a device in front of them for extended periods of time. It is both socially awkward and physically tiring.

Positive and negative claims of using AR visor, phone and tablet (*positive* in italic):

Claims	**AR visor**	**Phone**	**Tablet**
Field of view	*Full view*	Keyhole	Keyhole
Always connected	Have to put on	Slip out of pocket	In a bag
Power usage	Battery	Battery	Battery
View	*See through*	Camera image	Camera image
Holding the phone in front of you	*N/A*	Physically tiring	Physically tiring
Social acceptance	*Unobtrusive*	Intrusive	Intrusive

Questions to be answered through user test

- How useful is the image size using AR through a phone or tablet compared with AR visors?

  Experience: Well known that some phones have too small screen for some use cases. End user buys larger screen size phone to better meet these use cases.

  Test method: Test with existing AR applications in a phone and compare with visors or larger screen (VR).

- How much is the degradation of the augmented reality through a phone or tablet compared to "see through" AR visors?

Experience: Both phone and tablet has screen dependencies such as camera speed, camera quality, time to zoom, time to focus, time for processing, screen resolution, camera resolution, reflections, brightness etc.

Test method: Obvious? Compare specifications using data sheets.

Literature: Optical versus video see-through head-mounted displays in medical visualization [7]

- How socially acceptable is it to hold up the phone in front of you all the time?

  Literature: The results from the two Master theses; *Augmented Reality Visor Concept* and *Social Interaction in Augmented Reality*.

  Literature: Would you do that?: understanding social acceptance of gestural interfaces; [8]

  Test method: Perform focus group and user studies. Compare between using AR applications for phone (Junaio, Layar or Wikitude) and visor.

- Would indirect interaction through phone or tablet mean less immersion compared with AR visor?

  To click on screen to interact with an object instead of actually grabbing the air or pointing at it will give less immersion. Conversely the more direct interaction will give the user a better spatial sense, familiarity and enjoyment.

- Compare two interaction types: manipulating and exploring

  Test method: Perform usability test, and compare phone/tablet vs AR visor.

- Would indirect interaction through phone or tablet mean inaccuracy due to scaling and small involuntary hand movement?

- Will the user miss AR experience opportunities?

  Comment: This is obvious, a user interface which is most often out of sight or in your pocket is a lot of lost opportunities to discover new things or possible contextual data that can be offered by the mobile device. *Assuming the users wear the glasses.*

- Will the user perceive that he or she can execute a task faster and more accurately by using the visors than to take a mobile device from a pocket?

  Test method: Measure the time it takes to perform different actions with Fitts's law

- Is an optical see through system better for AR than a video see through system?

  Literature: Optical versus video see-through head-mounted displays in medical visualization [7]

8.2 H2: New application paradigm which dynamically filters application is better than the current application metaphor

Explanation

The current mobile device user interface and application paradigm are often centred on one, in focus, application which is displayed full screen. This application is often dedicated to supporting the user in carrying out specific task or tasks. We are assuming that this type of application paradigm is not the optimal paradigm for an AR visor (optical see through head worn system).

When using the current application metaphor the user needs to search for an application, start it and use it. Different applications are needed in different situations. A new application paradigm could be a non anthropomorphous service that filters applications by context; this means that the application search function should not be needed as much as today. However getting context aware guessing "wrong" has a higher penalty than the positive effects of getting them "right". Dividing apps up into things like "notifications", "actions" and "intents" and then displaying them free of their applications would lessen the impact of bad guesses. It would be interesting to investigate if WIMP is still a valid application metaphor. The menu could e.g. be shown on different surfaces on the palm; table, wall, body etc. but still remain as the familiar WIMP.

Direct Manipulation

Direct manipulation proposes that digital objects be designed at the interface so that they can be interacted with in ways that are analogous to how physical objects in the physical world are manipulated. In so doing, direct manipulation interfaces are assumed to enable users to feel that they are directly controlling the digital objects represented by the computer [9]. Three core principles are:

- Continuous representation of the objects and actions of interest;
- Rapid reversible incremental action with immediate feedback about the object of interest;
- Physical actions and button pressing instead of issuing commands with complex syntax

Who should be in control of the interface?

Different interaction types vary in terms of how much control a user has and how much the computer has. Whereas users are primarily in control for command-based and direct manipulation interfaces they are less so in context-aware environments and agent-based systems. User controlled interaction is based on the premise that people enjoy mastery and being in control. In contrast, context-aware control assumes that having the environment monitor, recognize, and detect deviations in a person's behaviour can enable timely, helpful and even critical information to be provided when considered appropriate.

To what extend do you need to be in control in your everyday and working life? Can you let computing technology monitor and decide what you need or do you prefer to tell it what you want to do? How would you feel if your assistant told you to drive more slowly because it had started to rain or drive faster or you will be late to your meeting? [9]

Sub hypothesis

- Direct manipulation UI is better than natural UI (e.g. Natural language UI,) see [10], such as Apple's Siri or Microsoft's office assistant Clippy).
    - User study of a central application that can assist with correct information and/or use correct application.

Literature: Anthropomorphic Agents as a User Interface Paradigm: Experimental Findings and a Framework for Research [11]

Literature: Embodies pedagogical agents [12]

- Context based filter which makes certain applications available is better than non filtered
    - User intended actions needs to be considered. The context based filtering assistant/application can't know what the user wants to do all the time. Possibility for using all application is needed.
    - In Samsung TVs today the applications are geographically dependent.
- How should user interact with local items?

  Test method: User study, by pointing, grabbing on items?
- How should user interact with items far away?

  Test method: User study by letting the user hold his hand in front of an interactive object the content is then mirrored into the hand. See *AR Visor Concept* page 35 [13].
- How should user interact with incoming events?
    - Events that come from a local place e.g. information about some offer from a shop or system information?
    - Events that come from something or someone far away?

Design input

- Use multimodal interaction techniques such as vibration, sound, text, picture, animation etc this way several of the human senses will be involved processing the information. Dominant senses can however in some situations wipe out other senses.
- A standardized framework that makes it possible for applications to plug-in and be used seamless is needed.
- Add: "Gaze tracking together object recognition allows adaption to the users varying interests without being intrusive."
- Important to find a balance between intrusiveness (what to present and how) and serendipity
    - Concept video Augmented City (watchable with 3D glasses) [14]
- The notification centre in Android (and recently in iOS) may be a good design for collecting notifications that does not have a connection to the local context
- The desktop metaphor with widgets may be a good inspiration
    - Only essential info is shown on a small area and can be placed next to other widgets for at a glance status
- The world browser

  - A constant inflow of items which augments and populates the world around us depending on our geo-location and the geo-location of items around us. Most likely there will be specific information architecture in order to solve such a population without the need for the mobile device to constantly scour the web with searches and to itself aggregate and filter the result.

- Self contained applications
  - Similar to how applications work today in that they take up some display real estate with a canvas where you can enter text or draw an image. Even if this app is similar to the ones today in that it can be an application installed it must still be adapted so that it can function in the real world: be interacted with using other input than mouse and keyboard or touch events. Can be placed o projected on different surfaces or locations where it needs to adapt to its surrounding.

- Tools to operate on real or virtual objects
  - Similar to how we can carry around tools such as a pen or a hammer etc. we could carry around virtual tools which are for applying changes or actions on real and virtual objects around us. These tools need to be directed at objects and act upon them instead of like applications today load a document and then within its own environment act on it. It will require a strong interconnectedness between different virtual tools and the objects that they can operate on.

8.3 H3: The AR visor gives an added value in everyday life to an extent that the user wants to keep them on

Explanation

The visor form factor (optical see through head worn system) allows for a never out of sight user experience. When something is constantly in view it is crucial that it does not irritate, interrupt or make the user feel discomfort. We believe that the user must be given added value by wearing the system while at the same time avoiding side effects which could make the user want to take them off.

The idea is to immerse the user with the mixed reality environment to an extent that the user wants to keep the AR visors on. The user should perceive a larger value of using the AR visor for everyday situations. Some important areas to achieve this are ergonomics, content, safety, privacy and technical requirements. Industrial design, form factor and light weight are some ergonomically issues. What type of content and how much content are also other factors to not overload the user with information. Further the user should feel safe using the visors factors as radiation, thermal (visors getting too warm) and nausea needs to be investigated. Indirect safety; when driving the car it is important that the user does not get “blue screen” covering user's sight. Other technical specifications that are important to keep the visors on are field of view, resolution, power consumption, weight, grounded glass etc.

Sub hypothesis

- The user perceives a larger value of using the AR visor for achieving certain goals, rather than resorting to other solutions.
    - Try to find out: In what situations do visors value or experience? In what situations would people use visors?
- Keeping the AR visors on makes the users perceive that he/she is always up to date (Local offers, social network, emergency communication.)

  Literature: Steve Mann papers, [15] e.g. his WearCam paper, [16]

  Test method: Through user studies get user opinions, user’s attitudes to wear glasses the entire time.
- The user feels more satisfied in his life because of the sense of increased control and more available options in his life
- The use of the visor's can be controlled by the user so that it never reaches a level of intrusion or irritation which forces the user to remove them in order to be able to complete a task.

  Test method: Evaluate by user studies and applying for instance NASA RTLX
- The AR visors feels safe for users and do not harm the user’s eyes, cause nausea, thermal issues or radiation issues.

  Test method: By questionnaire study.
- The AR visors are ergonomically good designed concerning industrial design, form factor, weight, etc.
    - *"Attractive things work better"*, Donald A. Norman

  Test method: By questionnaire study and/or interviews.

- The AR visors can have eye correction lenses (replace specs of a kind)
    - The AR visors might look odd worn over the normal specs
    - Perhaps contact lenses is the only possible resolution for people that need vision correction
- There have been several reports that HMD often create nausea. Further investigations are needed to dig into studies and see what the top causes for this nausea is. Is it accommodation – convergence conflict problems or is it poor registration (objects floating around) etc.?
    - ***Test method:*** User study?
    - Synchronization loss in video-pipe of the displays
    - Stereoscopic 3D effect vs. clone image feed
- Covering the centre of FOV means high intrusion
    - A user interface which is covering a central and large part of the FOV can easily create irritation and intrusiveness in case graphics is placed there without the direct command of the user. When objects in the surrounding are augmented with information the situation must be avoided where everywhere you look you will activate information popup. We assume that you may have indications of where there is information to be retrieved but then a select or more info action which can reveal more information. There must in that case be a balance between information given and the easiness of getting more without accidentally activating this more info function just by glancing at things.

      Literature: Peripheral Vision Annotation: Non-interference Information Presentation Method for Mobile Augmented Reality paper by Y. Yoshiguro and J. Rekimoto [17]

      Literature: There should be design guidelines on HUD type UI that work and does not work from military and computer gaming areas
- Overlay graphics that appear to be locked into a real world location must move with the environment
    - If a user is looking at a wall where there is a virtual painting hanging then moving the users head to the sides should have the painting hanging at the same place and not lag behind or appear to be floating and not pinned down. If many objects are constantly floating this may be like being drunk and things are spinning around resulting in poor UX.
    - If the overlay information can be included at the same focus distance as the object, the user will not be forced to frequently refocus his/her vision.

Design input

- The AR visor should be able to sustain rain
- It can be good to understand why a user does not want to use the AR visor and try to support the solution chosen by the user or to include characteristics that are important for the user. For instance if the user does not find it valuable to use the personal assistant for giving flight details, it should still support him in his/her attempt to find/double check the information, for instance by having the possibility to record the details given as he/she asks a person. Same goes for navigational advice. This tension between different solutions becomes evident if the activity diamond is applied.

- Attractive glasses from Eyez [18]
- Steve Mann's WearCam paper [16] has a list of extreme stress tests that very well cover what everyday needs could be:
    - The Casino Test: "Casino operators are perhaps the most camera–shy and the savviest in detecting cameras. Therefore, if the apparatus can pass undetected in the casino environment, under the scrutiny of pit bosses and croupiers, it can probably be used in other establishments such as grocery stores, where photography is also usually prohibited."
    - The Sports Test: "A sporting event, such as volley- ball, requires good hand–eye coordination" and "being able to play a competitive game of volleyball through the apparatus should be a reason- able test of its visual transparency and quality."
    - The Rain (Swim) Test: "An extreme scenario might be that if one were to fall off of a small sailing vessel or the like, into the water, that the apparatus and wearer would both survive."
- Should be see-through and have easy turn off
    - Maybe consider hardware on/off switch like in noise-cancelling headphones
    - Maybe there should be a quickly accessible mute button which does not turn the system off but suspends any graphics overlay which may be good in situations when the user must not be disturbed
- Should have context depending shut off if failure happens
- Indirect safety; when driving the car it is important that the user doesn't get “blue screen” covering user's sight.
    - Will driving with visors be banned similar to talking on the phone? Might be possible to glean information from Car Connectivity Consortium and its Driver Distraction rules.
        - Pioneer Head-Up display. hologram based AR Navigation system Pioneer [19] and video [20]

8.4 H4: Interaction redundancy results in low intrusion

Explanation

Since not all user contexts can be correctly sensed, an augmented reality mobile device should allow for different interaction modalities (e.g. voice, touch, gestures, and gaze) as well as different levels of discreteness (e.g. small discrete movements of the hand versus full body gestures). One example is when the hands of the user are busy which makes voice a possibility while touch input is not. Another example is when the user is in on a train next to a sleeping person thus one would not like to be forced to speak out aloud.

Input modalities and discreteness levels should be considered both from the perspective of the user and observers/bystanders.

The goal of investigating this area is to determine the minimum number of modalities and discreteness levels which an AR Visor should support in order to be useful in everyday life situations.

- Pocket interaction can be good if gestures are not ok, but otherwise gestures can be used.
- State of the art gaze-tracking might be a differentiator as an input modality and a enabler for new interaction paradigm
    - Eye-tracking research from Tobii [21]
- Redundancy
- Unforeseen contexts may stop the user to use certain modalities e.g. loud environment and voice communication, bumpy bus ride and small touch areas

Design input

- It should always be possible to perform simple operations when hands are busy. Examples could include:
    - Dismiss an application asking for user input or requesting the user's attention
- A user should never be forced to speak out loud in order to complete an action
- A user should never be forced to use full body or big movement gestures to complete an action
- It should be possible to stand and talk to someone and perform simple interactions without it being noticeable.
- A user should be able to receive a notification while talking to another person without feeling distracted

8.5 H5: Adjustable user interface elements that can change color and the placement of the user interface would be better than non adjustable user interface

Explanation

It is well documented that natural lighting conditions and real-world backgrounds affect the usability of optical see-through AR displays in outdoor environments. In many cases, outdoor environmental conditions can dramatically alter users' color perception of user interface (UI) elements, by for example, washing out text or icon colors [22]. The UI elements need to be adjustable to fit into the user's context. The colours should be able to change depending on the light conditions and background colour. However the colour should not change from white to pink but from white to light grey. Further the UI needs to change when out and running compared to sitting on the train. Even the placement of the UI should be adjustable depending on interesting objects in field of display.

Example: Pulling out information from the ToDo list-app about vicinity of the bike repair shop while biking with a broken horn and unobtrusively pointing on its location with a label "Repair bike here ->"

Example: Your keys holds to e.g. via Bluetooth, if the device (mediator/phone/visors) cannot find the keys around in your bag, you will get an unobtrusive notification, about forgotten keys.

Example: Imagine user place NFC tags across his social habitat (work space, home, car, on the objects he/she interacts), whenever phone catches those tags, some additional info about current context will be delivered to visors. E.g. phone is sitting on the work desk (next to NFC tag placed there), user gets notified about that consider a relevancy level

Tool: Colour blind UI check tool [23]

Literature: More Than Meets the Eye: An Engineering Study to Empirically Examine the Blending of Real and Virtual Colour Spaces [22]

Positive and negative claims of adjustable UI elements:

Claims	Positive	Negative
Adjustable colour	User can read and see the UI elements independent of the background	Intrusive if colour changes draws attention to the user
Adjustable placement	Not hindering interesting objects	Performance problem Irritating or difficult to read if UI element moves around too much
Fixed elements	Easy for the user to find some important information (battery, number of messages etc.)	Might hinder interesting objects.
System initiated	User doesn't have to do anything	Changing to bad colour
User initiated	User have control	If the user needs to change often irritating

Sub hypothesis

- Sound volume that is automatically adjusted to surrounding audio levels is more pleasant and results in less intrusion (no need to constantly manually adjust volume up or down) than a constant audio volume only manually adjustable by the end user

  Test method: Through user studies?

- Adjusting size of UI elements and activation areas according to the movement situation of the user/device will increase the accuracy of input and readability. Situations where this should be of special benefit are when walking or a bumpy car ride where the constant movement makes reading and input difficult for small text and input areas.

  Test method: Through user studies?

- Fixed placement of the UI elements can be acceptable in the specific use cases like cycling or skiing

Design input

- Brightness and colour of overlaid UI elements must adjust to brightness and colour of the user's environment in order to allow for good readability and to avoid eye strain

8.6 H6: To use same type of interactions for virtual object interaction as with real objects is better than indirect screen/pointer interaction

Explanation

When a user is wearing an AR visor (optical see through head worn system) there is a possibility to blend the real world and the virtual world. We can have virtual objects be displayed and positioned so that they appear to be in the real world. One example would be to draw a flower pot and flower sitting on the table. When the user moves around the sensors in the system takes that into account and redraws the image so that perspective and placement is correct. It is also possible to use a shared service which makes virtual objects be persistent between uses and between persons

Objects in real life are there for all our senses and we have through our lifetime learned how to handle them. In case of a tool like a hammer, we pick it up by the handle and can use it by swinging it towards a nail. All our real world objects have volume, weight and location. We know that we can put it in a container and bring it with us. These are just some of the properties of a real world objects of course.

The hypothesis is that if we can represent and handle virtual objects in a similar way as for real world objects then we will introduce a natural and intuitive way which will be easier and more efficient than a mouse pointer or touch screen indirect selection and activation paradigm for virtual objects.

Design input

- A virtual object should have a pervasive spatial location. If you place it on a table and walk out of the room, it should still be there if you return to the room a couple of minutes later.
- A virtual object can be picked up and brought along somewhere on the user's body (e.g. in a pocket, or in the hand of a user)
- The virtual object may have an owner which prohibits other users from picking up or acting on it
- The virtual object may be limitless in quantity which means that if it is picked up it will still remain in place while another instance of the virtual object is the one picked up. One example is a pen stand with one pen in it. The virtual pen object may be configured so there will always be a pen in the pen stand. If a user then picks up the pen it will be duplicated so that one pen is in the hand of the user picking it up but there will also be one pen left in the pen stand.
- There should be a simple way for a user to determine whether an object is virtual or not. One way would be to press a mute button on the visors which will make all virtual objects disappear. Another way would be for each object to be accompanied with a small floating text label which can be easily detected if looking for it.
- A virtual object shall be able to be retrieved or removed from the world by its owner without being at the same location.
- A virtual object shall be able to be operated or acted upon by both users with the capability of direct tangible interaction as well as indirect methods (pointer, touch screen or other indirect input).
- Indirect screen/pointer interaction for virtual object interaction is better than to use same type of interactions as with real objects
- Video link: More than pictures under glass [24]